Expressive power of parametrized quantum circuits

Yuxuan Du,[1,*] Min-Hsiu Hsieh,[2,†] Tongliang Liu,[1,‡] and Dacheng Tao[1,§]
[1]*UBTECH Sydney Artificial Intelligence Centre and the School of Information Technologies, Faculty of Engineering and Information Technologies, The University of Sydney, Sydney 2006, Australia*
[2]*Centre for Quantum Software and Information, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney 2007, Australia*

(Received 4 March 2020; accepted 18 May 2020; published 22 July 2020)

Parametrized quantum circuits (PQCs) have been broadly used as a hybrid quantum-classical machine learning scheme to accomplish generative tasks. However, whether PQCs have better expressive power than classical generative neural networks, such as restricted or deep Boltzmann machines, remains an open issue. In this paper, we prove that PQCs with a simple structure already outperform any classical neural network for generative tasks, unless the polynomial hierarchy collapses. Our proof builds on known results from tensor networks and quantum circuits (in particular, instantaneous quantum polynomial circuits). In addition, PQCs equipped with ancillary qubits for postselection may possess expressive power stronger than that of those without postselection. We employ them as an application for Bayesian learning, since it is possible to learn prior probabilities rather than assuming they are known. We expect that it will find many more applications in semisupervised learning where prior distributions are normally assumed to be unknown. Lastly, we conduct several numerical experiments using the Rigetti Forest platform to demonstrate the performance of the proposed Bayesian quantum circuit.

DOI: 10.1103/PhysRevResearch.2.033125

I. INTRODUCTION

There is a ubiquitous belief called "quantum supremacy" that quantum computers will outperform classical computers [1]. One characterization of quantum supremacy relates to the expressive power of quantum computing, since the probability distribution generated by quantum devices may not, classically, be sampled efficiently and accurately. Two leading proposals toward this goal are Boson sampling [2] and instantaneous quantum polynomial time (IQP) circuits [3].

The system noise in current implementations is known to be the major roadblock. Widespread explorations have been conducted to verify whether noisy intermediate-scale quantum (NISQ) [4] devices can also outperform classical computers for specific computation tasks. It has been proven that, with system noise, quantum supremacy will disappear in Boson sampling [5] but will remain in IQP [6]. In addition to demonstrating the existence of quantum supremacy, the issue of finding practical applications for NISQ devices with quantum advantages needs to be further studied.

Quantum machine learning problems have been popularized because of their ability to efficiently process tremendous amounts of data. They are also exploited as alternative testbeds to confirm quantum advantages [7–13]. By employing NISQ devices, potential quantum advantages may still be retained, benefiting from the fact that most statistical machine learning algorithms are robust to system noise; i.e., the noise contained in the input data and models has a negligible influence on the final results [14,15].

Expressive power is a central topic in classical machine learning and it has generated great interest in quantum machine learning. It is deeply tied to two major topics in machine learning: discriminative modeling and generative modeling, which aim to learn patterns and the probability distribution of input data [16], respectively. Expressive power in discriminative learning relates strongly to classification performance, e.g., by employing the kernel method [17], the kernel support vector machine can efficiently classify nonlinear data. In generative modeling, the expressive power of two highly successful models, the restricted Boltzmann machine (RBM) and the deep Boltzmann machine (DBM) [18,19], to represent quantum many-body states has been extensively investigated [20–24]. Consequently, the RBM and the DBM have been broadly applied to physics research, e.g., identifying phase transition, solving many-body wave functions, and accelerating Monte Carlo simulations [25–29].

Parametrized quantum circuits (PQCs) are promising NISQ devices that have demonstrated their potential to be applied to practical applications with quantum advantages. By employing variational hybrid quantum-classical algorithms, PQCs have been applied to accomplish both the generative [30–32] and the discriminative [33–36] tasks. A PQC is composed of a set of parametrized single-qubit and controlled single-qubit gates with noise, and the parameters are iteratively optimized by a classical optimizer. In general, the

[*]yudu5543@uni.sydney.edu.au
[†]min-hsiu.hsieh@uts.edu.au
[‡]tongliang.liu@sydney.edu.au
[§]dacheng.tao@sydney.edu.au

proposed PQCs can be divided into two types: multiple-layer PQCs (MPQCs) and tensor network PQCs (TPQCs). An MPQC consists of multiple blocks of quantum circuits in which the arrangement of quantum gates in each block is identical [30,31,37]. Mathematically, we denote the input quantum state as $|0\rangle^{\otimes N}$ with N qubits, the total number of blocks as L, and the ith block as $U(\boldsymbol{\theta}^i)$, where the number of parameters is proportional to the number of qubits $|\boldsymbol{\theta}| \propto N$ and N is logarithmically proportional to the dimension of the generated data. The generated quantum state of an MPQC, $|\Phi\rangle$, is defined as $|\Phi\rangle = \prod_{i=1}^{L} U(\boldsymbol{\theta}^i)\,|0\rangle^{\otimes N}$. The TPQCs treat each block as a local tensor. The arrangement of the blocks follows a specified tensor network, such as matrix product states and a tree tensor network [35]. Mathematically, the ith block $U(\boldsymbol{\theta}^i)$ is composed of M_i local tensor blocks, with $M_i \propto N/2^i$, denoted as $U(\boldsymbol{\theta}^i) = \bigotimes_{j=1}^{M_i} U(\boldsymbol{\theta}^i_j)$. The generated state from a TPQC is defined as $|\Phi\rangle = \prod_{i=1}^{L} \bigotimes_{j=1}^{M_i} U(\boldsymbol{\theta}^i_j)\,|0\rangle^{\otimes N}$. Refer to Sec. II for more details.

Although PQCs have provided strong evidence of quantum advantage [38,39], two important questions remain unexplored: (i) What is the expressive power of PQCs? (ii) Is there any quantum advantage of PQCs that can be used to solve practical problems? A comparison of expressive power between PQCs and classical neural networks is desirable and may benefit both physics and machine learning areas, since PQCs are capable of solving many kinds of machine learning tasks and classical machine learning methods have also been extensively applied to physics research.

To analyze their relationships, we first prove that MPQCs can be formulated by the tensor network language. This shows that MPQCs, TPQCs, and classical neural networks have a close connection with tensor networks, such as matrix product states (MPSs) [40]. We then exploit entanglement entropy, as a metric that evaluates the expressive power of tensor network states, to characterize the expressive power of PQCs and neural networks. We provide a rigorous proof that, given the number of trainable parameters that polynomially scale with the number of qubits or visible neurons N, the bond dimensions represented by MPQCs, TQPCs, the DBM, and the long-range RBM scale with $O(\exp(N))$, while the bond dimensions represented by the short-range RBM scale with $O(\text{poly}(N))$.

Before answering the question of whether PQCs have any quantum advantages over classical generative algorithms, we remark that entanglement entropy is not the only metric for quantifying expressive power. The second metric to quantify the expressive power of PQC and neural networks is the runtime complexity to simulate certain probability distributions. We prove that the probability distribution generated by an RBM and a DBM with $O(N)$ visible neurons and $O(\text{poly}(N))$ hidden neurons can be efficiently simulated by MPQCs in $O(\text{poly}(N))$ runtime. We further prove that instantaneous quantum polytime (IQP) circuits [41] are a special subclass of MPQCs. The probability distribution generated by IQP cannot be sampled efficiently and accurately by any classical neural network [3]. This indicates that, from the perspective of complexity theory, MPQCs have an expressive power stronger than that of classical neural networks and have the potential to become a practical application with quantum supremacy [42].

Finally, we equip MPQCs with ancillary qubits for postselection—a model we call ancillary driven MPQCs (AD-MPQCs). We show that the class of AD-MPQCs contains post-IQP circuits as a special case. Apart from the stronger expressive power, AD-MPQCs also provide additional benefits from the machine learning perspective. Specifically, AD-MPQCs with a simple structure, which we call the Bayesian quantum circuit (BQC), are devised for Bayesian learning. We indicate that the expressive power of the BQC is equivalent to that of a post-IQP circuit. From the machine learning point of view, the ancillary qubits of the BQC can be used to represent the additional information, such as a prior distribution. The BQC not only can exploit priors to improve the performance of a learning task but can also enable the estimation of prior distributions from the given data, which is highly desired for semisupervised learning [43]. A toy model is designed to verify its effectiveness. The BQC experiments are implemented in PYTHON, leveraging the pyQuil library to access the numerical simulator known as the quantum virtual machine (QVM) [44].

II. DEFINITIONS AND PRELIMINARIES

A. Boltzmann machine

The Boltzmann machine (BM), inspired by the Ising model, plays a significant role in the development of the deep neural network, which aims to learn a distribution over the set of its inputs [45,46]. Specifically, the BM can be divided into two parts: N visible units (visible neurons), $\boldsymbol{v} = \{v_i\}_{i=1}^{N}$, and M hidden units (hidden neurons), $\boldsymbol{h} = \{h_j\}_{j=1}^{M}$, where both visible and hidden neurons take binary inputs with $v_i \in \{0, 1\}$ and $h_j \in \{0, 1\}$. Given the trainable parameters w_{ij}, a_i, and b_j, the Hamiltonian is defined as $H(\boldsymbol{s}) = \sum_i a_i v_i + \sum_j b_j h_j + \sum_{i<j} w_{ij} s_i s_j$, with $\boldsymbol{s} = \{\boldsymbol{v}, \boldsymbol{h}\}$. The joint probability distribution over the visible and hidden units is defined as

$$P(\boldsymbol{v}, \boldsymbol{h}) = \frac{1}{\mathcal{Z}} e^{H(\boldsymbol{v},\boldsymbol{h})}, \tag{1}$$

where $\mathcal{Z} = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} e^{H(\boldsymbol{v},\boldsymbol{h})}$ is called the partition function. For generative tasks, the marginal probability distribution of visible units $P(\boldsymbol{v}) = \sum_{\boldsymbol{h}} P(\boldsymbol{v}, \boldsymbol{h})$ is expected to be maximized by optimizing w_{ij}, a_i, and b_j.

The RBM [47] is a special type of BM, which can be learned more efficiently. Mathematically, the Hamiltonian of the RBM is defined as $H(\boldsymbol{v}, \boldsymbol{h}) = \sum_i a_i v_i + \sum_j b_j h_j + \sum_{i,j} w_{ij} v_i h_j$, where only the inner connections between visible units and hidden units remain. An RBM is short range, if the connection between visible and hidden units is sparse and local. For short-range RBMs, the visible unit v_i only locally connects with $2k + 1$ hidden units $\{h_j, h_{j+1}, \ldots, h_{j+2k}\}$ with a small constant k or $k \sim O(\log M)$. Similarly, an RBM is nonsparse (or long range) if k satisfies $k \sim O(M)$.

A deep Boltzmann machine (DBM) [19], different from an RBM that includes only one layer of hidden units, contains many layers of hidden units. In DBMs, multiple hidden layers can be learned by training one hidden layer one at a time as for the RBM. When we calculate the probability distribution between the nth layer and the $n+1$th layer, the hidden units of the previous nth layer $\boldsymbol{h}^n$ are treated as visible units $\boldsymbol{v}^n$, and $P(\boldsymbol{v}^n, \boldsymbol{h}^{n+1})$ is obtained in the same manner as in the RBM.

Remark. Throughout the whole paper, we restrict that DBMs and RBMs only take one-dimensional input. Such a restriction is mild and has been employed in Ref. [21], since high-dimensional inputs of Boltzmann machine can be easily reshaped to the one-dimensional setting.

B. Tensor networks

An MPS is a natural choice to efficiently represent one-dimensional low-energy quantum states [40]. We denote a quantum state of one-dimensional lattice with N sites as $|\Psi\rangle = \sum_{j_1,j_2,\ldots,j_N=1}^{d} C_{j_1 j_2 \ldots j_N} |j_1\rangle \otimes |j_2\rangle \otimes \cdots \otimes |j_N\rangle$, where all sites have the same dimension d. The state $|\Psi\rangle$ can be completely described by a rank-N tensor $C_{j_1 j_2 \ldots j_N}$ with total d^N elements. However, such an exponentially scaling relation implies that the computation cost becomes expensive for large N. An MPS enables $|\Psi\rangle$ to be approximated with a high accuracy using only $O(\text{poly}(N))$ parameters. We rewrite $|\Psi\rangle$ as follows:

$$|\Psi\rangle = \sum_{l,r} C_{l,r} |l\rangle |r\rangle , \tag{2}$$

where l corresponds to the first site $l = j1$ and r corresponds to the rest of the $N-1$ sites, $r = (j_2, \ldots, j_N)$. Let $C_{l,r} = \sum_a U_{l,a} S_{a,a} V^{\dagger}_{a,r}$ be the singular value decomposition (SVD) of $C_{l,r}$. Then we have

$$|\Psi\rangle = \sum_{l,r}\sum_{a} U_{l,a} S_{a,a} V^{\dagger}_{a,r} |l\rangle |r\rangle = \sum_{a} S_a |a\rangle_l |a\rangle_r , \tag{3}$$

where $|a\rangle_l = \sum_l U_{l,a} |l\rangle$, $|a\rangle_r = \sum_r V_{r,a} |r\rangle$, and $S_a = S_{a,a}$ is the (a, a)th entry of S. Equation (3) is called the Schmidt decomposition and the entanglement of the bipartite systems l and r is characterized by S_a. Specifically, the bond dimensions D between the l and r subsystems are defined by the the number of nonzero values S as defined in Eq. (3), i.e., the rank of the matrix of S with $D = \text{rank}(S)$.

Through successively performing the SVD along each single site in turn, we can split out the rank-N tensor $C_{j_1 j_2 \ldots j_N}$ into N local tensors $\{A^{j_i}\}_{i=1}^N$. Mathematically, analogous to Eq. (3), the matrix product state of $|\Psi\rangle$ is defined as

$$\begin{aligned}
|\Psi\rangle &= \sum_{j_1\ldots j_N}\sum_{a_1} U_{j_1,a_1} S_{a_1,a_1} V^{\dagger}_{a_1,(j_2\ldots j_N)} |j_1\rangle |j_2\ldots j_N\rangle \\
&= \sum_{j_1\ldots j_N}\sum_{a_1} U_{j_1,a_1} C_{a_1,(j_2\ldots j_N)} |j_1\rangle |j_2\ldots j_N\rangle \\
&= \sum_{j_1\ldots j_N}\sum_{a_1} A^{j_1}_{a_1} U_{(a_1,j_2),a_2} C_{a_2,(j_3\ldots j_N)} |j_1\rangle |j_2\rangle |j_3\ldots j_N\rangle \\
&= \sum_{j_1\ldots j_N}\sum_{a_1\ldots a_{N-1}} A^{j_1}_{a_1} A^{j_2}_{a_1,a_2} \ldots A^{j_N}_{a_{N-1}} \prod_{i=1}^{N} |j_i\rangle ,
\end{aligned} \tag{4}$$

where S_{a_1,a_1} and $V^{\dagger}_{a_1,(j_2,\ldots,j_N)}$ have been multiplied and reshaped to a vector $C_{a_1,(j_2,\ldots,j_N)}$, and the matrix U_{j_1} is decomposed into a collection of d row vectors A^{j_1} with entries $A^{j_1}_{a_1} = U_{j_1,a_1}$. The number of parameters (elements) in the MPS scales as $O(NdD^2)$, where D represents the maximum of bond dimensions among all $S_{a,a}$. When D is small or some truncated methods are employed to keep M small, MPSs can efficiently approximate the quantum states with polynomial parameters.

C. Entanglement entropy

The entanglement (also called von Neumann entropy) $\mathcal{S}(\rho)$ of a bipartite system ρ_{AB} is defined as

$$\mathcal{S}(\rho_A) = -\text{Tr}(\rho_A \ln \rho_A) = -\text{Tr}(\rho_B \ln \rho_B) = \mathcal{S}(\rho_B), \tag{5}$$

where $\rho_A = \text{Tr}_B \rho_{AB}$ is the reduced density matrix of system A. For a quantum system A that satisfies area (volume) law, its entanglement entropy grows proportionally with the boundary area (volume) of system A, denoted as $\mathcal{S}(\rho_A) = O(|\partial A|)$ $[= O(|A|)]$.

The maximum entanglement entropy of a bipartite system is logarithmically bounded by the bond dimensions D as defined in Sec. II B, i.e., $\mathcal{S}(\rho_A) \sim \ln D$. The MPS can efficiently represent a wide class of quantum states that satisfies area law, where the bond dimensions D and the number of parameters used in the MPS to describe the state are small.

The entanglement entropy $\mathcal{S}(\rho)$ of bipartite systems closely relates to the bond dimensions D as formulated in Eq. (3). As claimed in Ref. [48], certain quantum states that satisfy one-dimensional area law can be efficiently simulated by MPS with constant bond dimensions, while one-dimensional area-law states only occupy a partial Hilbert space. In order to employ the MPS to simulate quantum states beyond one-dimensional area law, e.g., the quantum states cover the whole Hilbert space, the bond dimensions D are required to be exponentially large with respect to the size of the system.

D. Quantum circuits

Analogous to classical computers, a quantum computer accomplishes its computation by applying quantum gates to quantum bits (qubits).

As stated in Ref. [49], a set of single-qubit and two-qubit gates, which consists of rotation gates and controlled-NOT (CNOT) gates, is universal for quantum computation. In other words, any function computable in this model can be computed using only these gates. We denote the phase rotation gate R_ϕ, the z-axis rotation gate $R_Z(\theta)$, the x-axis rotation gate $R_X(\gamma)$, and the y-axis rotation gate $R_Y(\alpha)$ as follows:

$$R_\phi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}, \quad R_X(\gamma) = \begin{pmatrix} \cos(\gamma/2) & i\sin(\gamma/2) \\ i\sin(\gamma/2) & \cos(\gamma/2) \end{pmatrix},$$

$$R_Z(\theta) = \begin{pmatrix} e^{i\theta/2} & 0 \\ 0 & e^{-i\theta/2} \end{pmatrix},$$

$$R_Y(\alpha) = \begin{pmatrix} \cos(\alpha/2) & -\sin(\alpha/2) \\ \sin(\alpha/2) & \cos(\alpha/2) \end{pmatrix}.$$

The CNOT gate, defined as

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

flips the target qubit iff the the control qubit is $|1\rangle$. Other quantum gates can be represented by the above universal gate

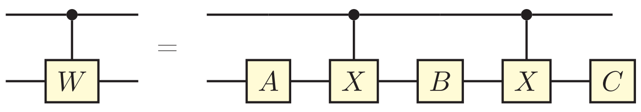


FIG. 1. Simulation of controlled unitary gates.

set, e.g., the Pauli-Z gate, defined as $Z = \left(\begin{smallmatrix}1 & 0\\ 0 & -1\end{smallmatrix}\right)$, can be represented by $R_Z(\theta = \pi)$; the T gate, defined as $T = \left(\begin{smallmatrix}1 & 0\\ 0 & e^{i\pi/4}\end{smallmatrix}\right)$, can be represented by $R_\phi(\phi = \pi/4)$; the Hadamard gate (H gate), defined as $H = 1/\sqrt{2}\left(\begin{smallmatrix}1 & 1\\ 1 & -1\end{smallmatrix}\right)$; can be represented by $R_X(\gamma = \pi/2)R_Z(\theta = \pi/2)R_X(\gamma = \pi/2)$; and the two-qubit controlled-Z gate (shorted as CZ gate), defined as

$$CZ = \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & -1\end{pmatrix}, \qquad (6)$$

can be represented by $(\mathbb{I} \otimes H)\text{CNOT}(\mathbb{I} \otimes H)$.

Proposition 1 below demonstrates how to use the universal gate set to express other quantum gates [50].

Proposition 1. A controlled unitary W gate (CW) can be simulated by a quantum network composed of single-qubit gates and a CNOT gate. Suppose that $W = R_Z(\theta)R_Y(\alpha)R_Z(\beta)$, then as shown in Fig. 1, it can be simulated by the quantum circuits A, B, and C, where $A = R_Z(\theta)R_Y(\alpha/2)$, $B = R_Y(-\alpha/2)R_Z(-\theta/2 - \beta/2)$, and $C = R_Z(\beta/2 - \theta/2)$.

IQP circuits

The instantaneous quantum polynomial (IQP) circuit consists of commute gates that are diagonal in the Z basis. The basic framework of IQP circuits is illustrated in Fig. 2.

Given N qubits, the IQP circuits can generate distributions $p_I = |\langle 0^{\otimes N}|H^{\otimes N}U_Z H^{\otimes N}|0^{\otimes N}\rangle|^2$, where U_Z is composed of $O(\text{poly}(N))$ commuting gates, e.g., the single-qubit T gate and the CZ gate.

IQP circuits have been proven to be capable of generating probability distributions p_I that cannot be classically simulated efficiently [51]. The main result of IQP circuits is summarized in the following proposition.

Proposition 2. If the output probability distributions generated by uniform families of IQP circuits could be weakly classically simulated to within multiplicative error $1 \leqslant c \leqslant \sqrt{2}$, then post $- BPP = PP$ and the polynomial hierarchy would collapse to its third level.

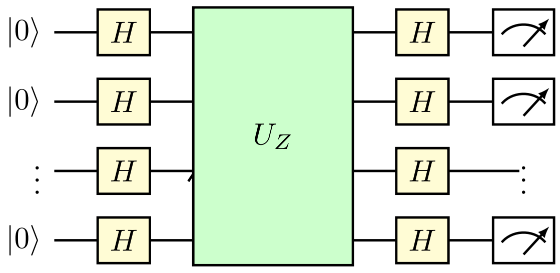


FIG. 2. A general framework of IQP circuits.

E. Parametrized quantum circuits

Parametrized quantum circuits (PQCs), as a special type of quantum circuit model, are composed of a set of parametrized single-qubit and controlled single-qubit gates. In this work, a PQC is used to implement a unitary transformation operator $U(\boldsymbol{\theta})$ with $O(\text{poly}(N))$ parametrized quantum gates, where N is the number of input qubits.

Several recent works [30,31,35] have employed PQCs to accomplish generative tasks. One major reason is that the superposition property allows the number of trainable parameters to be dramatically reduced. In generative tasks, PQCs produce the probability $q(X = \boldsymbol{x}) = |\langle \boldsymbol{x}|\Psi_G\rangle|^2$ measured by the computational basis $|\boldsymbol{x}\rangle$, where

$$|\Psi_G\rangle = U(\boldsymbol{\theta})\,|0\rangle^{\otimes N}. \qquad (7)$$

The parameters $\boldsymbol{\theta}$ can be optimized using only classical approaches,

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}[q(X), p(X)], \qquad (8)$$

where $\mathcal{L}[\cdot,\cdot]$ is a loss function that measures the dissimilarity of the generated and the targeted probability distributions. For example, suppose that the loss function is negative log-likelihood [52], then the optimizing process is

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{D}\sum_{i=1}^{D} -\log q(X = \boldsymbol{x}_i), \quad X \sim p(X), \qquad (9)$$

where the dataset $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^{D}$ is sampled from the targeted probability distribution $p(X)$, the size of $\mathcal{D}$ is D, and each example of $\mathcal{D}$ is denoted as $\boldsymbol{x}_i$ for $i \in [1, D]$.

Another loss function that is broadly employed is the maximum mean discrepancy (MMD). The MMD loss is defined as

$$\mathcal{L} = \left\| \sum_{\boldsymbol{\lambda}}\sum_{x^i \in \boldsymbol{x}} q(x^i, \boldsymbol{\lambda})\phi(x^i) - \sum_{x^i \in \boldsymbol{x}} p(x^i)\phi(x^i) \right\|^2, \qquad (10)$$

where $\phi(x^i)$ maps the ith input data, x^i, into a high-dimensional reproducing Kernel Hilbert space [53], and $\sum_{x^i \in \boldsymbol{x}} p(x^i)$ refers to the target probability distribution. More details about the MMD loss and how to optimize it by employing the gradient descent method with unbiased estimation are introduced in Refs. [31,54].

In the following, we define two types of PQCs that are the focus of the present work, where the major difference is the layout of quantum gates to compose $U(\boldsymbol{\theta})$.

1. Multilayer parametrized quantum circuits

Multilayer parametrized quantum circuits (MPQCs) are composed of L blocks, where each block implements $U(\boldsymbol{\theta}^i)$, with $i \in [1, L]$ and $L \sim \text{poly}(N)$. A unitary operator $U(\boldsymbol{\theta}) = \prod_{i=1}^{L} U(\boldsymbol{\theta}^i)$ is applied to N input qubits. An example of MPQCs is illustrated in Fig. 3. In each block, the arrangement of quantum gates is identical. Moreover, each qubit is operated with at least one parametrized gate (denoted by yellow color), and CNOT gates within the block can connect arbitrary two qubits. Another requirement in MPQCs is the amount of CNOT gates is no larger than N in each block.

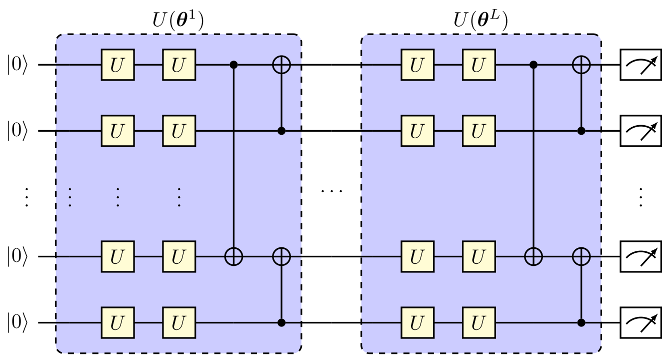


FIG. 3. Illustration of MPQCs.

Using MPQCs to accomplish generative tasks has been explored in Refs. [30,31], while the layout of quantum gates in each block and the optimization methods have been varied.

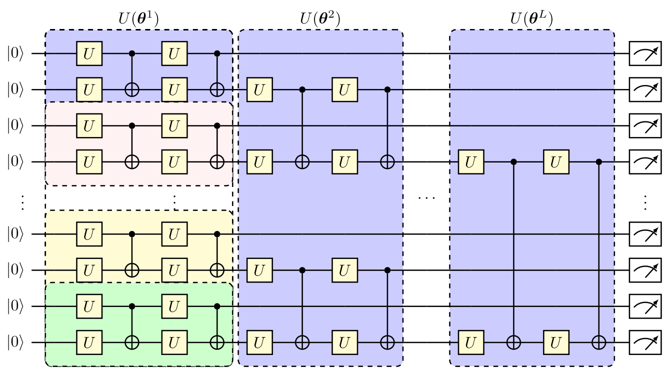
2. *Tensor network parametrized quantum circuits*

Another type of PQC is the tensor network PQC (TPQC), which generally inherits the tensor network structures, i.e., MPSs, tree tensor networks, or multiscale entanglement renormalization ansatz. In other words, CNOT gates can only connect two local qubits. Mathematically, the quantum state $|\Psi_G\rangle$ generated by TPQCs is formulated as

$$|\Psi_G\rangle = \prod_{i=1}^{L}\bigotimes_{j=1}^{M_i} U\left(\boldsymbol{\theta}_j^i\right)|0\rangle^{\otimes N}, \tag{11}$$

where M_i represents the number of local blocks in the block $U(\boldsymbol{\theta}^i)$ and N/M_i is constant for all i. For example, a TPQC that inherits the layout of a tree tensor network is given in Fig. 4.

Figure 5 illustrates another example of a TPQC that inherits the layout of an MPS tensor network. Employing TPQCs to accomplish generative tasks has been investigated in Ref. [35]. We remark that in this paper, we only consider TPQCs with one-dimensional tensor network structures, e.g., matrix product states and tree tensor networks.



$U(\boldsymbol{\theta}^1)$
$U(\boldsymbol{\theta}^2)$
$U(\boldsymbol{\theta}^L)$
$|0\rangle$
U

FIG. 4. An example of a TPQC that inherits the layout of a tree tensor network, where the CNOT gates in different layers have different local constraints.

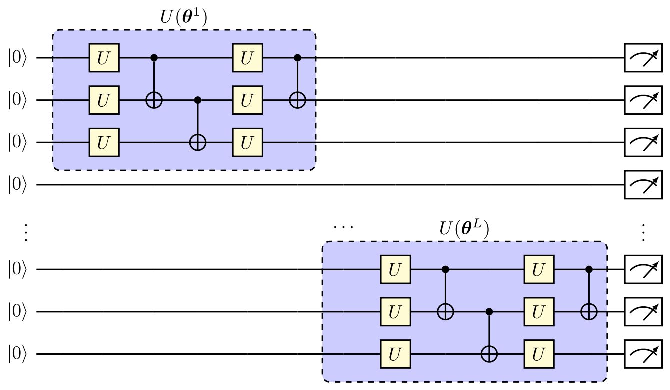


FIG. 5. An example of TPQCs that inherit the layout of MPSs.

III. EXPRESSIVE POWER OF PARAMETRIZED QUANTUM CIRCUITS

For ease of understanding, we divide this section into two parts. We first introduce the main result of this study in Sec. III A. We then elaborate the technical details in Sec. III B.

A. Main result

The goal of a generative learning network is to learn a distribution $q(x)$ that approximates a targeted probability distribution $p(x)$ within a tolerable error ϵ. The expressive power of a generative learning machine directly determines how well the generated distribution can match the target distribution, e.g., Eq. (8). The stronger the expressive power is, the smaller the dissimilarity of two distributions will be. The formal definition of the expressive power of generative models is the following.

Definition 1 (Comparison of expressive power). Consider two generative models, A and B, that are selected from MPQCs, TPQCs, short-range RBMs, long-range RBMs, and DBMs as formulated in Sec. II. We say that the expressive power of model A is no weaker than that of model B if the following relation is satisfied:

$$(\mathcal{E}_A \geqslant \mathcal{E}_B) \wedge (\mathcal{C}_A \geqslant \mathcal{C}_B), \tag{12}$$

where $\mathcal{E}_X$ and $\mathcal{C}_X$ are the maximal bond dimension and the runtime of the generative model X, respectively, to simulate a certain *probability distribution*.

Following conventions [55], the runtime complexity of quantum circuit models refers to the gate complexity. The definition of gate complexity is as follows.

Definition 2 (Gate complexity, [56,57]). Let elementary gates be the CNOT gates and arbitrary single qubit gates. The gate complexity for a unitary U is the minimal number of elementary gates that is needed to implement U.

Remark. The motivation to employ bond dimensions as measures to quantify the expressive power of generative model comes from the following fact. As explained in Sec. II, a larger bond dimension for a generative model implies that it can represent a larger set of quantum states living in a Hilbert space. Equivalently, bond dimensions measure the diversity of distributions (quantum states) that can be produced by the generative model.

By employing the definition of expressive power, our main result in this paper is as follows.

Theorem 1. The expressive power of MPQCs with $O(\text{poly}(N))$ single qubit gates and CNOT gates and classical neural networks with $O(\text{poly}(N))$ trainable parameters, where N refers to the number of qubits or the visible units, can be ordered as follows: MPQCs > DBM > long-range RBM > short-range RBM, unless the polynomial hierarchy (PH) collapses.

Remark. Note that our result, Theorem 1, does not contradict Ref. [20]. Recall that Gao and Duan [20] claimed that a deep neural network can efficiently represent most quantum states in the following sense. Specifically, given any N-qubit quantum state $|\Phi\rangle$ that is generated by a quantum circuit with T circuit depth, their study showed that a DBM with N visible neurons and $O(NT)$ hidden neurons is sufficient to compute the probability amplitude $\langle \boldsymbol{v}|\Phi\rangle$, given a fixed basis $|\boldsymbol{v}\rangle$. However, we quantify the expressive power of different generative models by their ability to simulate the "whole probability distribution," instead of just one probability amplitude in any given fixed basis.

B. Proof sketch of Theorem 1

Theorem 1 can be proved following Lemma 1, Theorem 2, and Theorem 3. Lemma 1 below indicates that MPQCs and TPQCs are capable of simulating MPSs with exponentially large bond dimensions.

Lemma 1. MPQCs and TPQCs with N qubits and $O(\text{poly}(N))$ blocks, where each block contains $O(N)$ trainable parameters and at most N CNOT gates, can efficiently represent MPSs with bond dimensions $D = O(\exp(N))$.

The proof of Lemma 1 is given in Appendix A. Lemma 1 and the results in Refs. [21,24] together establish the following relation in terms of the bond dimensions:

$$\mathcal{E}_{\text{MPQC}} = \mathcal{E}_{\text{DBM}} = \mathcal{E}_{\text{L-RBM}} = \mathcal{E}_{\text{TPQC}} > \mathcal{E}_{\text{S-RBM}}, \tag{13}$$

where $\mathcal{E}_{\text{L-RBM}}$ and $\mathcal{E}_{\text{S-RBM}}$ represent the maximum bond dimensions of the long-range RBM and the short-range RBM, respectively. In particular, the algorithm proposed by Chen *et al.* [21] indicates that when we reformulate the short-range RBM (long-range RBM) by the MPS, the corresponding bond dimensions are constant [exponentially scales with respect to N visible neurons, i.e., $D = O(\exp(N))$]. Moreover, since both short-range and long-range RBMs can be treated as the special case of the DBM (by setting the number of hidden layers as one), we conclude that the DBM can efficiently represent the MPS with $O(\exp(N))$ bond dimensions. Likewise, the Ref. [24] obtains a result similar to that of Ref. [21], i.e., $\mathcal{E}_{\text{L-RBM}} > \mathcal{E}_{\text{S-RBM}}$.

Remark. The close relation between bond dimensions and entanglement entropy, as explained in Sec. II, provides the following implication. The short-range RBM can only efficiently represent a certain class of quantum states satisfied with area law, while the long-range RBM and the long-range DBM have the ability to simulate quantum states satisfying volume law entanglement.

Theorems 2 and 3 characterize the expressive power of different generative models in terms of the computation complexity to approximate a certain distribution. In particular, Theorem 2 proves that probability distributions represented by RBMs and DBMs with N visible neurons and $O(\text{poly}(N))$ hidden neurons can be efficiently simulated by MPQCs.

Theorem 2. Any probability distribution produced by short-range RBMs, long-range RBMs, or DBMs with N visible neurons and $O(\text{poly}(N))$ hidden neurons can be generated by some MPQCs with $O(\text{poly}(N))$ circuit depths (runtime complexity).

The proof of Theorem 2 is provided in Appendix B. Note that the proof of Theorem 2 follows from a mapping rule, which we designed, to transform RBMs or DBMs with N visible neurons and $O(\text{poly}(N))$ hidden neurons to MPQCs with $O(\text{poly}(N))$ quantum gates.

Next, by making connections with IQP circuits, Theorem 3 quantifies the expressive power of MPQCs is stronger than that of DBMs and RBMs.

Theorem 3. There exist probability distributions generated by MPQCs with N qubits and $O(\text{poly}(N))$ quantum gates cannot be simulated efficiently by classical neural networks unless the PH collapses.

The proof of Theorem 3 is demonstrated in Appendix C.

Since it has been proven that a DBM (long-range RBM) has an expressive power stronger than that of a long-range RBM (short-range RBM) [20,21], i.e., $\mathcal{C}_{\text{DBM}} > \mathcal{C}_{\text{L-RBM}}$ and $\mathcal{C}_{\text{L-RBM}} > \mathcal{C}_{\text{S-RBM}}$, we employ the conclusion of Theorems 2 and 3 to obtain

$$\mathcal{C}_{\text{MPQC}} > \mathcal{C}_{\text{DBM}} > \mathcal{C}_{\text{L-RBM}} > \mathcal{C}_{\text{S-RBM}}. \tag{14}$$

Combining Eqs. (13) and (14), it is straightforward to compare the expressive power of different generative models as formulated in Definition 1, which concludes the proof of Theorem 1.

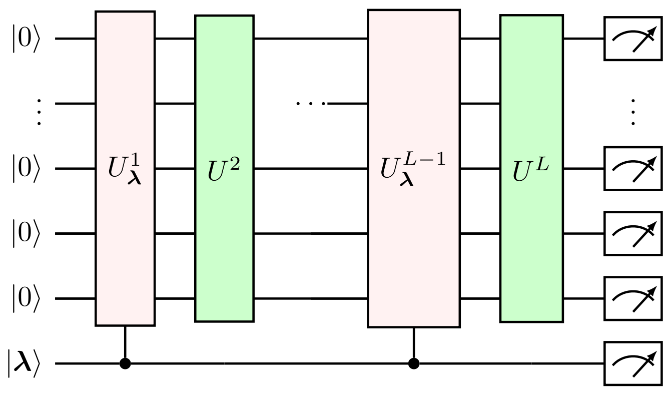
IV. BAYESIAN QUANTUM CIRCUIT

In Bayesian inference, additional information about a prior probability distribution $p(\boldsymbol{\lambda})$ which represents our beliefs about the parameters of the learning algorithm is given, and the posterior probability distribution $p(\boldsymbol{\lambda}|\boldsymbol{x})$ can be obtained by Bayes' rule,

$$p(\boldsymbol{\lambda}|\boldsymbol{x}) = p(\boldsymbol{\lambda})p(\boldsymbol{x}|\boldsymbol{\lambda}) \Big/ \int_{\boldsymbol{\lambda}} p(\boldsymbol{\lambda})p(\boldsymbol{x}|\boldsymbol{\lambda})d\boldsymbol{\lambda},$$

where $p(\boldsymbol{x}|\boldsymbol{\lambda})$ is known as the likelihood function. It has been shown that the performance of many learning tasks can be dramatically improved if Bayesian models are employed [58–62].

Considering the significance of the Bayesian approach in classical machine learning, we devise a Bayesian quantum circuit (BQC) that enables PQCs to accomplish quantum machine learning tasks with Bayesian advantages. We remark that our BQC is an interesting quantum method for a Bayesian generative model based on PQCs. The proposed BQC is capable of explicitly and efficiently generating prior, likelihood, and posterior distributions. Furthermore, we demonstrate that



$|0\rangle$
$|0\rangle$
$|0\rangle$
$|0\rangle$
$|\boldsymbol{\lambda}\rangle$
$U^1_{\boldsymbol{\lambda}}$
U^2
$U^{L-1}_{\boldsymbol{\lambda}}$
U^L

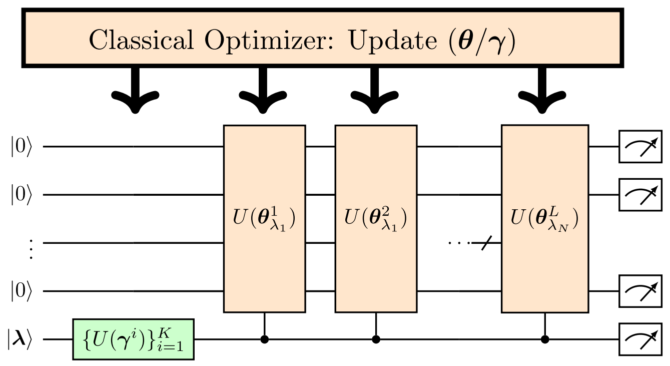
FIG. 6. A general framework of an AD-MPQC. The arrangement of quantum gates in each block is identical.

the BQC may possess an expressive power stronger than that of MPQCs studied in the previous section.

A. Layouts and optimization of the BQC

Before elaborating on the BQC, we first define the ancillary driven MPQCs (AD-MPQCs). AD-MPQCs can be divided into two parts, of which the first part aims to generate the targeted distribution and the second part aims to conduct postselection. In contrast to MPQCs, in which all blocks are directly applied to the data qubits, some blocks in AD-MPQCs are conditionally applied to the data qubits for specific ancillary quantum states. A general layout of an AD-MPQC is illustrated in Fig. 6, in which the common shared blocks are highlighted in green and $|\boldsymbol{\lambda}\rangle$ represents all possible combinations of M ancillary qubits with $|\boldsymbol{\lambda}\rangle = \{|0\rangle, |1\rangle\}^{\otimes M}$.

The BQC in Fig. 7 is a special case of AD-MPQCs in which the commonly shared blocks (green blocks in Fig. 6) do not exist. In the BQC, after applying K blocks $\{U(\boldsymbol{\gamma}^i)\}_{i=1}^K$ to M ancillary qubits, the generated state is $|\Psi_A\rangle = \prod_{i=1}^K U(\boldsymbol{\gamma}^i)\,|0\rangle^{\otimes M}$. Measuring the state $|\Psi_A\rangle$ by computational basis, the prior distribution $q(\boldsymbol{\lambda}) = |\langle\boldsymbol{\lambda}|\Psi_A\rangle|^2$ is generated. Similarly, after conditionally applying L blocks $\{U(\boldsymbol{\theta}^i_{\lambda_i})\}_{i=1}^L$ to N data qubits iff the ancillary state is $|\lambda_i\rangle$, $\forall\lambda_i \in \boldsymbol{\lambda}$, and measuring by computational basis $|\boldsymbol{x}\rangle$, the likelihood distribution $q(\boldsymbol{x}|\lambda_i) = |\langle\boldsymbol{x}, \lambda_i|\Psi_{\boldsymbol{x},\boldsymbol{\lambda}}\rangle|^2$ is generated, where $|\Psi_{\boldsymbol{x},\boldsymbol{\lambda}}\rangle$ is the



Classical Optimizer: Update $(\boldsymbol{\theta}/\boldsymbol{\gamma})$
$|0\rangle$
$|0\rangle$
$|0\rangle$
$|\boldsymbol{\lambda}\rangle$
$U(\boldsymbol{\theta}^1_{\lambda_1})$
$U(\boldsymbol{\theta}^2_{\lambda_1})$
$U(\boldsymbol{\theta}^L_{\lambda_N})$
$\{U(\gamma^i)\}_{i=1}^K$

FIG. 7. The general scheme of the proposed BQC.

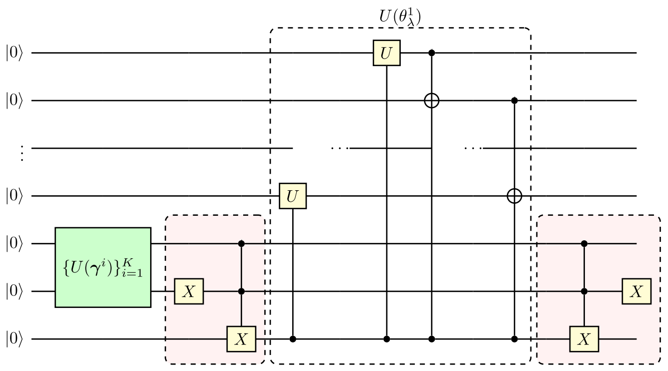


FIG. 8. An example of conditionally applying $U(\boldsymbol{\theta}^1_{\lambda_k})$ to data qubits iff $|\lambda_k\rangle = |01\rangle$.

quantum state generated by data qubits and ancillary qubits after applying a total of $K + |\boldsymbol{\lambda}|L$ blocks.

In the BQC, the parametrized gates in $U(\boldsymbol{\theta}^i_{\lambda_i})$ are controlled rotational qubits gates, e.g., controlled phase gate $CR_\phi(\phi)$; controlled rotation gate along the x axis, $CR_X(\gamma)$; controlled rotation gate along the y axis, $CR_Y(\alpha)$; and controlled rotation gate along the z axis, $CR_Z(\theta)$, which are controlled by the ancillary quantum state $|\boldsymbol{\lambda}\rangle$. To reduce the gate complexity, we introduce a flag qubit that is conditionally activated for the specified ancillary state, which enables each parametrized controlled rotational gate to have only one control qubit. As a result of this extra controlled qubit, the CNOT gates used in MPQCs are replaced by N Toffoli gates. Each Toffoli gate can be efficiently implemented by ten single-qubit gates and six CNOT gates. We give an intuitive example of how to apply the block $U(\boldsymbol{\theta}^1_{\lambda_k})$ to the data qubits iff the ancillary state is $\boldsymbol{\lambda}_k = |10\rangle$ in Fig. 8. The green region represents encoding the state $|\Psi_A\rangle$ into ancillary qubits. The two pink regions represent how to conditionally activate and uncompute the flag qubit for the specific ancillary state $|01\rangle$. The black dotted box illustrates how the block $U(\boldsymbol{\theta}^1_{\lambda_k})$ is conditionally applied to the data register for the specified ancillary state $|\lambda_k\rangle = |01\rangle$.

In the training process, we employ the MMD defined in Eq. (10) as the loss function. By measuring the data register and the ancillary register, the joint distribution $q(x^i, \boldsymbol{\lambda})$ is obtained by $q(x^i, \boldsymbol{\lambda}) = \sum_{\lambda_k\in\boldsymbol{\lambda}} |\langle x^i, \lambda_k|\Phi_{\boldsymbol{x},\boldsymbol{\lambda}}\rangle|^2$, where $|\Phi_{\boldsymbol{x},\boldsymbol{\lambda}}\rangle$ refers to the entanglement quantum states generated by the BQC.

B. Expressive power of the BQC and AD-MPQCs

Here we discuss the expressive power of the BQC and AD-MPQC. Given N data qubits and M ancillary qubits, the generated quantum state of the BQC is

$$|\Psi\rangle = \sum_{i=1}^{2^M} \alpha_i \prod_{j=1}^{L} U\left(\boldsymbol{\theta}^j_{\lambda_i}\right)|0\rangle^{\otimes N}|\lambda_i\rangle\,, \tag{15}$$

where α_i stands for the probability amplitude of state $|\lambda_i\rangle$ with $\sum_i |\alpha_i|^2 = 1$. Since $\langle\lambda_i|\lambda_j\rangle = \delta_{ij}$, the generated states corresponding to different ancillary quantum states are independent. When there is only one ancillary quantum state $|\boldsymbol{\lambda}| = 1$, the number of string operators is one and the BQC

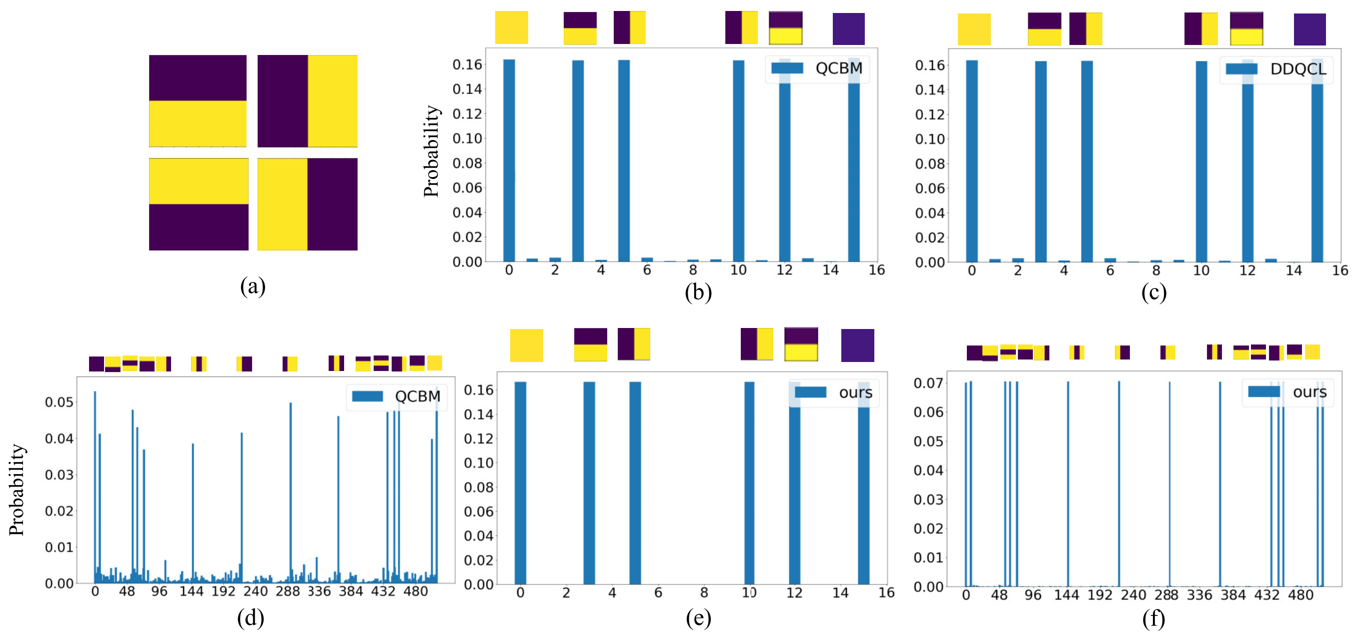


FIG. 9. The generative results obtained from DDQCL, the QCBM, and our model. Since the BAS dataset can be regard as a set of binary images, it can be mapped into different integers, as the x axis of figures. Panels (b), (c), and (e) are the generated result of 2×2 BAS images using the QCBM, DDQCL, and the BQC respectively. Figures (d) and (f) are the generated results of 3×3 BAS images using the QCBM and the BQC, respectively.

is equivalent to an MPQC. This implies that the expressive power of the BQC cannot be worse than that of MPQCs. Additionally, since the BQC is a special case of AD-MPQCs, the expressive power of AD-MPQCs cannot be worse than that of the BQC. Therefore, from the perspective of the entanglement entropy, the expressive power of the BQC and the AD-MPQCs cannot be worse than that of MPQCs. Since the post-IQP circuit can be efficiently formulated by both AD-MPQCs and the BQC, a better expressive power of the BQC is obtained compared to MPQCs from the perspective of computational complexity.

V. NUMERICAL EXPERIMENTS

A. Generating bar-and-stripe dataset

To demonstrate the advancements of the proposed BQC, we first use the BQC to accomplish generative tasks, e.g., generating 2×2 and 3×3 bars and stripes (BAS) datasets. A BAS dataset is composed of vertical bars and horizontal stripes, and some examples of BAS are shown in Fig. 9(a). For $n \times m$ pixels, the number of images that belong to BAS is $N_{\text{BAS}} = 2^n + 2^m - 2$. The target distribution of such a generative task is denoted as $p(\boldsymbol{x})$, where $p(\boldsymbol{x}_i) = 1/N_{\text{BAS}}$ iff $\boldsymbol{x}_i$ is a valid BAS image. The generated probability distribution of the BQC, $q(\boldsymbol{x}) = \sum_{\lambda_i \in \boldsymbol{\lambda}} q(\boldsymbol{x}, \lambda_i)$, aims to approximate the targeted distribution $p(\boldsymbol{x})$, where the $\boldsymbol{x}$ refers to the generated the images, $|\boldsymbol{\lambda}|$ refers to the number of valid BAS patterns, and $q(\boldsymbol{x}, \lambda_i)$ refers to the probability distribution of the generated images given specific λ_i.

We compare the generative performance of the BQC with two existing MPQCs in the literature, i.e., data-driven quantum circuit learning (DDQCL) [30] and the quantum circuit born machine (QCBM) [31]. Two major differences between DDQCL and the QCBM are the layout of CNOT gates in each block and the optimization methods. In DDQCL, the topology of CNOT gates is based on the topology of quantum devices, such as chain, star, and all connections. A gradient-free optimization approach is employed, i.e., the swarm optimization algorithm. In the QCBM, the topology of CNOT gates is determined by the Chow-Liu tree algorithm, which is inspired by the graphical models to efficiently extract information from training data among different nodes. The unbiased gradient-based optimization approach is employed in the training process. In accordance with the conventions in previous study, in the BQC, all BAS patterns are encoded in the qubits, where each data qubit stands for a pixel of the BAS image.

For the task of generating BAS images, the prior is a uniform distribution, since all BAS images are expected to be generated with the same probability. Through applying K blocks to the ancillary register with M qubits, the generated quantum state $|\boldsymbol{\lambda}\rangle$ is formulated as $|\boldsymbol{\lambda}\rangle = \prod_{i=1}^{K} U(\boldsymbol{\gamma}^i)\,|0\rangle^{\otimes M}$, where $q(\boldsymbol{\lambda} = \lambda_i) = |\langle \lambda_i | \boldsymbol{\lambda} \rangle|^2 = 1/N_{\text{BAS}}$, $|\boldsymbol{\lambda}| = N_{\text{BAS}}$, and $M = \lceil \log N_{\text{BAS}} \rceil$. Since the BAS patterns are encoded into the qubits, the total number of data qubits is $N = n \times m$. For the specified ancillary state $\boldsymbol{\lambda} = \lambda_i$, L blocks $\{U(\boldsymbol{\theta}^i_{\lambda_i})\}_{i=1}^{L}$ are conditionally applied to the N data qubits, where total $|\boldsymbol{\lambda}|L$ blocks are required in the BQC. Since there exists a one-to-one mapping that each λ_i aims to represent a specific BAS image, we have $q(\boldsymbol{x} = x_i) = q(\boldsymbol{x} = x_i, \boldsymbol{\lambda} = \lambda_i)$, where $q(\boldsymbol{x} = x_i, \boldsymbol{\lambda} = \lambda_j) = 0$ for $i \neq j$. We remark that it is a special case in generative tasks.

We first train the BQC to generate BAS images with 2×2 pixels, where $N_{\text{BAS}} = 6$ valid images are expected to be generated uniformly after learning. In the experiment, the numbers of data qubits and ancillary qubits are set to be $N=4$ and $M=3$, respectively. Since the prior distribution is known, the parameters of $K=2$ blocks $\{U_j(\boldsymbol{\gamma})\}_{j=1}^2$ are fixed, where the generated state is $\prod_{j=1}^2 U_j(\boldsymbol{\gamma})\,|0\rangle^{\otimes M} = 1/\sqrt{6}\sum_i |\lambda_i\rangle$, with $\lambda_i \in \boldsymbol{\lambda}$ and $|\boldsymbol{\lambda}|=6$. In the numerical simulation, we use the function provided by a QVM to directly generate the prior distribution $p(\boldsymbol{\lambda})$. In the learning process, we set $L=2$ blocks $\{U_j(\boldsymbol{\theta}^i_{\lambda_i})\}_{i=1}^2$ for the specified ancillary quantum state, where each block only contains 4 $CR_Y(\alpha)$ gates (interacting with 4 data qubits separately) and the number of Toffoli gates that connect 2 qubits in sequence is also 4, as illustrated in Fig. 8. A total of 48 trainable parameters are updated in the learning process.

When the BQC is applied to generate 3×3 BAS images, with $N_{\text{BAS}}=14$, the numbers of data qubits N and ancillary qubits M are set as 9 and 4, respectively. A uniformly ancillary state is first generated by using the function provided by the QVM. Analogous to the 2×2 BAS case, we set $L=2$ and each block contains 9 $CR_Y(\alpha)$ gates (interacting with 9 data qubits separately) and 9 Toffoli gates. Therefore, a total of 112 parameters are updated in the learning process.

Since QVM allows us to read the quantum states directly, the distribution of BAS images can be accessed accurately as measuring infinite times. The experimental results are illustrated in Fig. 9. Here we define the accuracy as N_{BAS}/N, where N represents the total number of generated images and N_{BAS} represents the number of generated images that have BAS patterns. As shown in Table I, the BQC outperforms state-of-the-art PQCs, where the accuracy to generate BAS 2×2 and 3×3 images is 99.96% and 98.65%, respectively.

TABLE I. Accuracies for generative 2×2 and 3×3 BAS datasets.

	Model	DDQCL	QCBM	BQC
2×2	Accuracy (%)	83.82	98.46	99.96
3×3	Accuracy (%)		65.36	98.65

B. Learning prior distribution

How to learn a prior distribution $q(\boldsymbol{\lambda})$ efficiently and accurately is one critical topic in machine learning, e.g., to learn the class priors in semisupervised learning. Meanwhile, class priors are also important in learning very sparse data and developing binary classifiers to discriminate positive and unlabeled data [63,64]

To confirm the effectiveness of the BQC to learn class prior distributions $q(\boldsymbol{\lambda})$ from given data, we devise a toy model. Specifically, the training data (referred to the test data with unlabeled class in the above example) are sampled from a joint distribution $p(\boldsymbol{x},\boldsymbol{\lambda})$, i.e., $p(\boldsymbol{x},\boldsymbol{\lambda}) = p(\boldsymbol{\lambda})p(\boldsymbol{x}|\boldsymbol{\lambda})$, with $|\boldsymbol{\lambda}|=2$, where the known class conditional densities are $p(\boldsymbol{x}|\boldsymbol{\lambda}=\lambda_1)\sim\mathcal{N}_1(\mu_1,\sigma_1)$ and $p(\boldsymbol{x}|\boldsymbol{\lambda}=\lambda_2)\sim\mathcal{N}_2(\mu_2,\sigma_2)$. $\mathcal{N}_1(\mu_1,\sigma_1)$ and $\mathcal{N}_2(\mu_2,\sigma_2)$ are two Gaussian distributions with means μ_1 and μ_2 and variations σ_1 and σ_2, respectively. In this toy model, the means and variances of $\mathcal{N}_1$ and $\mathcal{N}_2$ are set as $\mu_1=16$, $\mu_2=64$ and $\sigma_1=2$, $\sigma_2=4$, respectively. For each class, the known class conditional density distribution is generated by applying $L=7$ blocks $\{U(\boldsymbol{\theta}^i)_{\lambda_k}\}_{i=1}^L$ to the 7 data qubits with $N=7$. Alternatively, 14 blocks are employed to describe $p(\boldsymbol{x}|\boldsymbol{\lambda})$ with a total of 98 fixed parameters. Since $\boldsymbol{x}$ is encoded into qubits as the variable in $\mathcal{N}_1$ and $\mathcal{N}_2$, it is represented as a bit string and should be integers, where the maximum value of $\boldsymbol{x}$ is $\boldsymbol{x}_{\max}=2^N$ and N is the number of qubits.

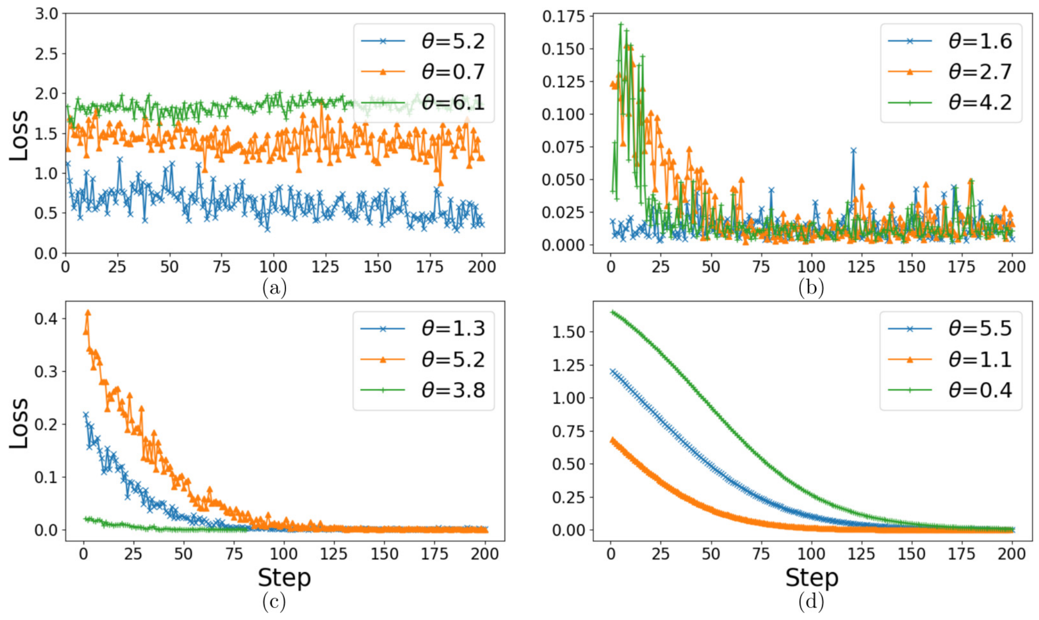


FIG. 10. The figure illustrates the MMD loss functions for $p(\lambda_1)=0.70$ with different parameter settings.

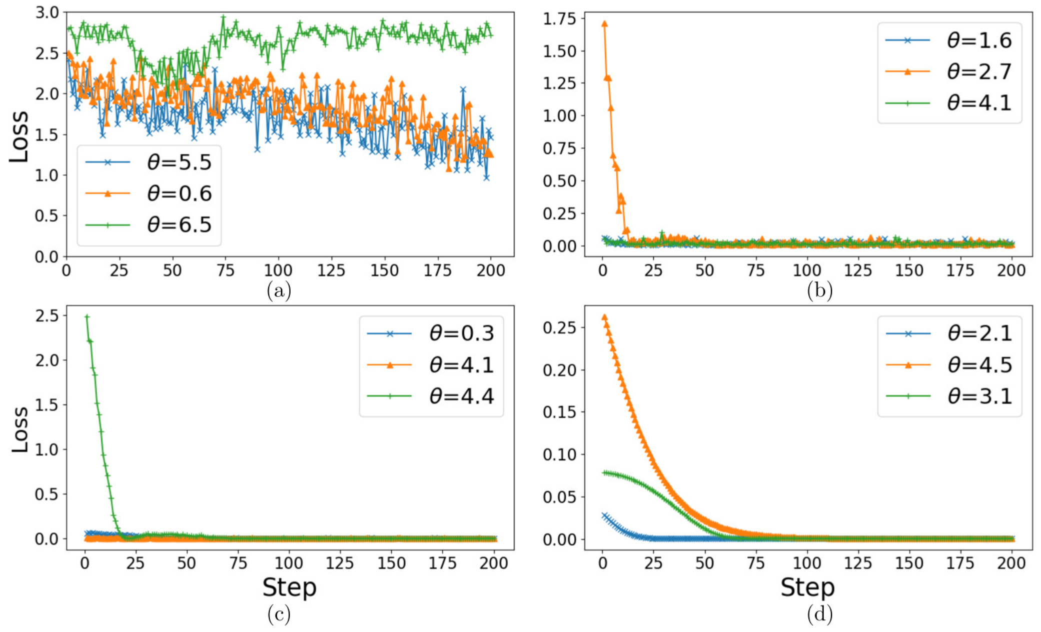


FIG. 11. The figure illustrates the MMD loss functions for $p(\lambda_1) = 0.85$ with different parameter settings.

In the training process, we estimate two sets of targeted coefficients, i.e., $p(\lambda_1) = 0.7$, $p(\lambda_2) = 0.3$ and $p(\lambda_1) = 0.85$, $p(\lambda_2) = 0.15$, respectively. Due to $|\boldsymbol{\lambda}| = 2$, we set the number of ancillary qubits as $M = 1$ and employ 1 block $U(\boldsymbol{\gamma}^i)$ to learn the class prior distribution, where the blocks only contains one parametrized $R_y(\alpha)$ gate. We first use the BQC to learn the targeted coefficient $p(\lambda_1) = 0.7$, and the MMD loss functions with three different measurement settings and two gradient descent optimization methods are shown in Fig. 10. For each setting, we repeat the experiments three times with varied initialized parameters, which are indicated by different colors. The training loss by setting the number of measurements as 100 and employing the general stochastic gradient descent optimization method is shown in Fig. 10(a). Then, we employ the unbiased gradient descent method [54] and set the number of measurements as 100, 1000, and ∞. The training losses are illustrated in Figs. 10(b), 10(c), and 10(d), respectively. We next use the BQC to learn the targeted coefficient $p(\lambda_1) = 0.85$. Following the same parameter settings, the training losses are shown in Fig. 11. The two numerical simulation results are listed in Table II. The small variance is mainly caused by the fact that the limited parameters $\boldsymbol{\theta}$ cannot approximate $\mathcal{N}_1$ and $\mathcal{N}_2$ well. We remark that different initial parameters have subtle influences on the convergence but the number of measurement determines if the loss can be converged.

VI. CONCLUSION AND DISCUSSION

In this paper, our first contribution is to the evaluation of the expressive power of parametrized quantum circuits and classical neural networks. Characterized by the entanglement entropy, we prove that MPQCs, TPQCs, long-range RBMs, and long-range DBMs can efficiently simulate MPSs that the corresponding bond dimensions exponentially scale with the number of inputs (visible neurons) N, which cannot be efficiently simulated by the short-range RBMs. We next prove that MPQCs can efficiently simulate probability distributions generated by an IQP circuit. These distributions are difficult to simulate efficiently by classical neural networks unless the polynomial hierarchy collapses. We therefore see that MPQCs have expressive power stronger than that of classical neural networks.

Our second contribution is the proposal of the BQC to accomplish Bayesian learning tasks. The BQC is a special case of AD-MPQCs that can efficiently simulate the probability distribution generated by post-IQP circuits. It has stronger expressive power over MPQCs without ancillary qubits. In

TABLE II. Learning prior distribution with $M = 1$ and $N = 7$. Here QVM* stands for employing a general optimization method, while the QVM employs the unbiased estimation optimization method.

Methods	Measurements	$P(\lambda_1)$	$P(\lambda_2)$	Variance	$P(\lambda_1)$	$P(\lambda_2)$	Variance
Target		0.70	0.30		0.85	0.15	
QVM*	100	0.163	0.867	4.47×10^{-2}	0.555	0.445	1.44×10^{-1}
QVM	100	0.706	0.294	1.66×10^{-4}	0.868	0.132	8.47×10^{-5}
QVM	1000	0.702	0.298	5.74×10^{-6}	0.856	0.144	5.94×10^{-6}
QVM	∞	0.701	0.299	6.12×10^{-10}	0.855	0.145	4.67×10^{-9}

addition, the postselection operation enables the BQC to accomplish machine learning tasks without knowledge about prior distributions. We perform two numerical simulations to validate the effectiveness of the BQC. The first numerical simulation uses the BQC to generate BAS images, in which the BQC outperforms state-of-the-art PQCs. The second numerical simulation uses the BQC to learn the class prior distribution, which is highly desirable for semisupervised learning. The simulation results demonstrate that the BQC can accurately estimate the prior distributions. These two tasks can be efficiently implemented on near-term quantum devices.

A PQC is a hybrid quantum classical learning scheme that has accomplished various learning tasks using a limited number of quantum gates and a shallow quantum circuit depth. With the benefit of the strong expressive power and efficient implementation on near-term quantum devices, PQCs have the potential to tackle practical problems with quantum advantages. One future direction is to explore how to use PQCs to solve practical machine learning problems and to investigate whether the proposed quantum learning model can provide a definitive quantum advantage.

ACKNOWLEDGMENTS

This research is supported by ARC Projects No. FL-170100117 and No. DP-180103424. M.H. is supported by an ARC Future Fellowship under Grant No. FT140100574. The PYTHON codes for our experiments are available upon request.

APPENDIX A: PROOF OF LEMMA 1

Proof of Lemma 1. The proof of Lemma 1 leverages the well-known result of tensor networks, MPSs can only efficiently simulate quantum circuits with polynomial logarithmic circuit depth [65].

In PQCs, only CNOT gates can increase bond dimensions. Specifically,

$$\text{CNOT} = \sum_{\sigma,\tau,\sigma',\tau'\in\{0,1\}} \mathcal{O}_{\sigma,\tau}^{\sigma',\tau'} |\sigma'\rangle|\tau'\rangle\langle\tau|\langle\sigma|, \tag{A1}$$

where $\mathcal{O}_{\sigma,\tau}^{\sigma',\tau'}$ is a rank-4 tensor with $\mathcal{O}_{\sigma=0,\tau=0}^{\sigma'=0,\tau'=0} = \mathcal{O}_{\sigma=1,\tau=1}^{\sigma'=1,\tau'=1} = \mathcal{O}_{\sigma=0,\tau=1}^{\sigma'=1,\tau'=1} = \mathcal{O}_{\sigma=1,\tau=1}^{\sigma'=1,\tau'=0} = 1$ and 0 otherwise. The CNOT gate can be decomposed into two local tensors with bond dimension $D = 2$. One possible solution is

$$\mathcal{O}_{\sigma,\tau}^{\sigma',\tau'} = \sum_{b\in\{0,1\}} W_{1b}^{\sigma',\sigma} W_{2b}^{\tau',\tau}, \tag{A2}$$

where $W_{1b}^{\sigma',\sigma}$ and $W_{2b}^{\tau',\tau}$ correspond to two local rank-3 tensors, and their explicit representations are as follows:

$$W_{10}^{\sigma',\sigma} = [1 \quad 00 \quad 0], \quad W_{11}^{\sigma',\sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$
$$W_{20}^{\tau',\tau} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad W_{21}^{\tau',\tau} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{A3}$$

Suppose that there exists k CNOT gates between the ith and $(i+1)$th qubits, where the first i qubits and the remaining $N-i$ qubits compose a bipartite system; the maximal bond dimensions of such a bipartite system are 2^k.

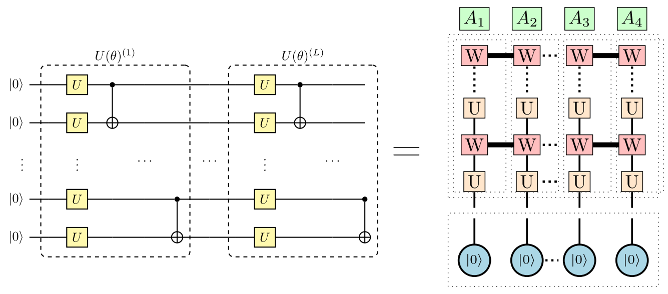


FIG. 12. The mapping between a MPQC and a MPS.

Since the bond dimensions exponentially scale with the number of CNOT gates, $O(\text{poly}(\log D))$ blocks are required to generate an MPS with bond dimensions D.

Figure 12 depicts a mapping between a MPQC and a MPS, where, for illustrative purpose, we assume that $N-1$ CNOT gates are applied to the data qubits in sequence. The middle section of Fig. 12 indicates the effects of CNOT gates and parametrized single-qubit gates. All local tensors applied to the same qubit can be merged into one local tensor [cf. Eq. (4)] and yield the corresponding MPS, as shown in the right section of Fig. 12.

Following the same routine, MPSs can only simulate TPQCs with polynomial logarithmic circuit depth [35].

Alternatively, when MPQCs and TPQCs have the polynomial circuit depth, the generated states have exponential dimensions, which cannot be efficiently simulated by MPSs. ■

APPENDIX B: PROOF OF THEOREM 2

For ease of understanding, we reformulate the DBM to a two-layer graph structure as the RBM does before elaborating the proof of Theorem 2.

Lemma 2. The DBM with N visible neuron and $O(\text{poly}(N))$ hidden neurons can be reformulated as a long-range RBM with N visible neurons and $O(\text{poly}(N))$ virtual and hidden neurons.

Proof of Theorem 2. We first illustrate how to use MPQCs to efficiently simulate an RBM. We then generalize the result to a DBM by leveraging the conclusion of Lemma 2. Note that we restrict the weights $\{w, a, b\}$ of the RBM and the DBM to real constants. Our analysis can be easily generalized to the complex value case.

There are two preprocessing steps before transforming an RBM to MPQCs. Suppose that the original RBM contains N visible neurons and M hidden neurons, as shown in the upper left panel of Fig. 13. The first preprocessing step is the splitting rule, where we substitute the original hidden neuron by new hidden neurons so that each hidden neuron only connects to one visible neuron. We exemplify the splitting rule in the upper and lower right panels of Fig. 13. In particular, for the original RBM, the hidden neuron h_k connects to visible neurons v_i and v_j with weights $\{w_{ki}, a_i, b_k\}$ and $\{w_{kj}, a_j, b_k\}$, respectively. The first step of reformulation is introducing the hidden neurons h_o and h_l to ensure that each hidden neuron only connects to one visible neuron. Denote that the weights for h_o and h_l are $\{w'_{oi}, a'_i, b'_o\}$ and $\{w'_{lj}, a'_j, b'_l\}$, respectively.

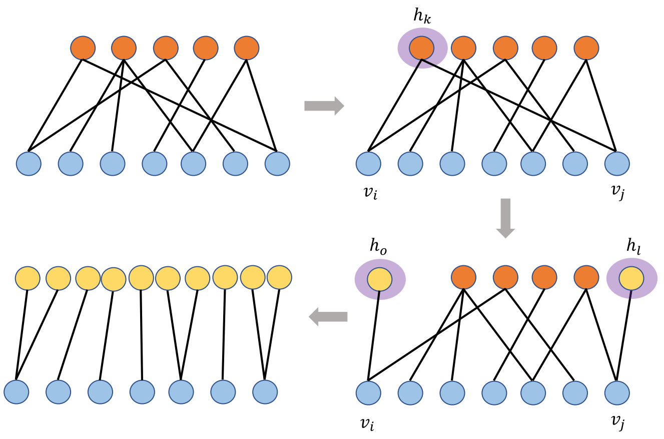


FIG. 13. The first preprocessing step to transform the RBM to MPQCs. The upper left panel shows the original RBM. The upper right panel illustrates that the hidden neuron h_k, as highlighted by purple shadow, is selected and split. The lower right panel demonstrates the splitting result, where each two new hidden neurons h_o and h_l only connect to the visible neurons v_i and v_j, respectively. Following the same routine, the lower left panel exhibits the result of splitting all hidden neurons of the original RBM.

The mapping rule should satisfy the following equation:

$$\sum_{h_k} e^{h_k w_{ki} v_i + h_k w_{kj} v_j + a_i v_i + a_j v_j + b_k h_k}$$
$$= \sum_{h_o, h_l} e^{h_o w'_{oi} v_i + b'_o h_o + a'_i v_i + h_l w'_{lj} v_j + b'_l h_l + a'_j v_j}. \tag{B1}$$

Since there are four combinations for v_i and v_j, i.e., $(v_i, v_j) \in \{\{0, 0\}, \{0, 1\}, \{1, 0\}, \{1, 1\}\}$, a possible solution for Eq. (B1) is

$$\begin{aligned}
a'_i &= a_i,\ a_j = a'_j,\ b'_o = \ln\left(\frac{e^{w_{kj}+w_{ki}} - e^{b_k}}{e^{w_{kj}} - 1 - e^{w_{kj}+w_{ki}} + e^{b_k}}\right),\\
b'_l &= \ln\left(\frac{e^{b_k+w_{kj}} - e^{b_k+w_{kj}+w_{ki}} + e^{2b_k} - e^{w_{kj}+w_{ki}}}{e^{w_{kj}} - 1}\right),\\
w'_{lj} &= \ln\left(e^{w_{kj}} + \frac{(1 - e^{b_k})e^{2w_{kj}+w_{ki}} - e^{w_{kj}+w_{ki}} + e^{b_k}}{e^{b_k+w_{kj}}(1 - e^{e^{w_{ki}}}) + e^{2b_k} - e^{w_{kj}+w_{ki}}}\right),\\
w'_{oi} &= \ln\left(1 + \frac{e^{b_k} + e^{w_{kj}} + e^{w_{ki}} - e^{b_k+w_{ki}}}{e^{b_k+w_{ki}+w_{kj}} - e^{2b_k} - e^{w_{ki}+w_{kj}} - e^{b_k}}\right).
\end{aligned} \tag{B2}$$

The second preprocessing step is the merging rule. As shown in Fig. 14, all hidden neurons that connect to the same visible neuron are merging into the new hidden neuron. In particular, we denote $\{w'_{ji}, b'_j, a'_i\}$, $\{w'_{ki}, b'_k, a'_i\}$, and $\{w''_{li}, b''_l, a''_i\}$ as the weights between h_j and v_i, h_k and v_i, and h_l and v_i, respectively. The merging rule that transforms h_j and h_k to h_l should satisfy the following equation:

$$\sum_{h_j, h_k} e^{h_j w'_{ji} v_i + b'_j h_j + h_k w'_{ki} v_i + b'_k h_k + a'_i v_i}$$
$$= \sum_{h_l} e^{h_l w''_{li} v_i + b''_l h_j + a''_i v_i}. \tag{B3}$$

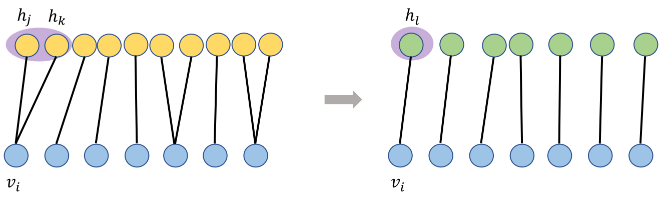


FIG. 14. The second preprocessing step to transform the RBM to MPQCs. The left (right) panel shows the RBM after the first (second) preprocessing step. In particular, all hidden neurons that connect to the same visible neuron are merging into the new hidden neuron. For example, hidden neurons h_j and h_k that connect to v_i are substituted by h_l.

Since v_i can take two values, i.e., $v_i \in \{0, 1\}$, a possible solution for Eq. (B3) is

$$\begin{aligned}
a''_i &= a'_i,\ b''_l = \ln(1 + e^{b'_k} + e^{b'_l} + e^{b'_k+b'_l}),\\
w''_{li} &= \ln\left(\frac{e^{w'_{ji}+b'_j} + e^{w'_{ki}+b'_k} + e^{w'_{ji}+w'_{ki}+b'_j+b'_k}}{1 + e^{b'_j} + e^{b'_k} + e^{b'_j+b'_k}}\right).
\end{aligned} \tag{B4}$$

Note that the preprocessed RBM, with at most N hidden neurons as shown in the right panel of Fig. 14, expresses the same probability distribution as the original RBM does. Following such an observation, we employ MPQCs to simulate the preprocessed RBM. We demonstrate the mapping rule in Fig. 15. In particular, as shown in the upper panel of Fig. 15, we initially apply Hadamard gates to an input state, which represents the probability distribution of the RBM without hidden neurons. Mathematically, given an RBM with N visible neurons and zero hidden neurons, the probability $P(\boldsymbol{v})$ for $\boldsymbol{v} \in \{0, 1\}^N$ is $1/2^N$. The equivalent mapping for MPQCs is applying N Hadamard gates on the input state $|0\rangle^N$.

In order to simulate the probability distribution of the target RBM, we iteratively attach the hidden neurons and find the corresponding parameters for MPQCs. As shown in the

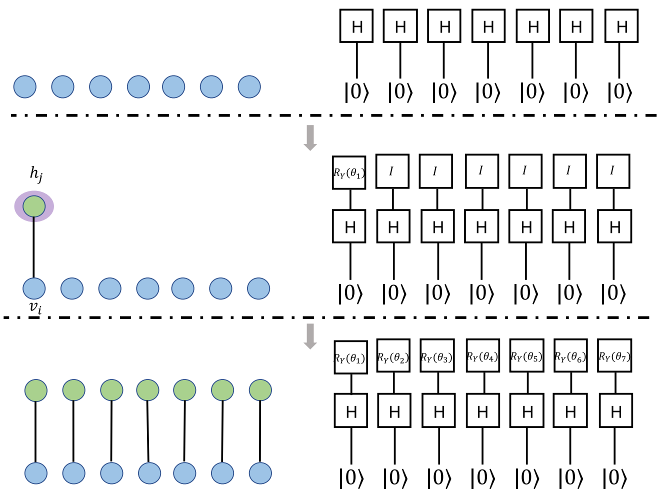


FIG. 15. The mapping rule from the RBM to MPQCs. The left panel shows the RBM and the right panel illustrates the equivalent MPQC representation. An iterative updating rule is employed to map RBM to MPQCs, where each hidden neuron corresponds to a parametrized rotation single-qubit gate along the Y axis.

middle part of Fig. 15, h_j only connects to v_i with parameters $\{w''_{ji}, a''_i, b''_j\}$, where we apply $R_Y(\theta_i)$ to the ith qubit of MPQCs to simulate such an operation. The mapping rule to between θ_i and $\{w''_{ji}, a''_i, b''_j\}$ is as follows. Denote the probability distribution function before introducing the connection between h_j and v_i as $\Phi(\boldsymbol{v})$, i.e., $\Phi(\boldsymbol{v}) = \sum_h e^{H(\boldsymbol{v},\boldsymbol{h})}/\mathcal{Z}$. After introducing the connection between h_j and v_i, the probability distribution is denoted as $\tilde{\Phi}(\boldsymbol{v})$, i.e.,

$$\tilde{\Phi}(\boldsymbol{v}) = \frac{1}{C}\sum_{h_j} e^{a''_i v_i + h_j w''_{ji} v_i + b''_j h_j}\Phi(\boldsymbol{v}), \tag{B5}$$

where $\{a''_i, b''_j, w''_{ji}\}$ are defined in Eq. (B4), $C = \mathcal{Z}/\mathcal{Z}'$, $\mathcal{Z}' = \sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}'} e^{H(\boldsymbol{v},\boldsymbol{h}')}$, and $\boldsymbol{h}' = \{\boldsymbol{h}, h_j\}$. For MPQCs, denote the probability amplitude before applying $R_Y(\theta_i)$ to the ith qubit as $|\Phi(\boldsymbol{v})\rangle$. Applying the $R_Y(\theta_i)$ gate to the state $|\Phi(\boldsymbol{v})\rangle$ generates the state $|\tilde{\Phi}(\boldsymbol{v})\rangle$, with

$$|\tilde{\Phi}(\boldsymbol{v})\rangle = [R_Y(\theta_i)\otimes \mathbb{I}^{\otimes N-1}]\,|\Phi(\boldsymbol{v})\rangle\,. \tag{B6}$$

The distribution represented by the RBM and MPQCs satisfies the relation $\Phi(\boldsymbol{v}) = |\langle \boldsymbol{v}|\Phi(\boldsymbol{v})\rangle|^2$. We denote $A_i(0)$ as $A_i(0) = \sum_{\boldsymbol{v}\setminus v_i}\sum_{v_i=0,\boldsymbol{h}} e^{H(\boldsymbol{v},\boldsymbol{h})}/\mathcal{Z}$, which refers to the summation of probability among all 2^{N-1} bases $|v_1, \ldots, v_i = 0, \ldots, v_N\rangle$ with $\boldsymbol{v}\setminus v_i \in \{0,1\}^{N-1}$. The normalization constraint implies $A_i(1) = 1 - A_i(0)$ and $A_i(1) = \sum_{\boldsymbol{v}\setminus v_i}\sum_{v_i=1,\boldsymbol{h}} e^{H(\boldsymbol{v},\boldsymbol{h})}/\mathcal{Z}$.

We now explore the optimal parameter of θ_i such that

$$\tilde{\Phi}(\boldsymbol{v}) = |\,\langle \boldsymbol{v}|\tilde{\Phi}(\boldsymbol{v})\rangle\,|^2. \tag{B7}$$

By expanding Eq. (B7) using Eqs. (B5) and (B6), we obtain

$$\frac{1}{C}\left(1+e^{b''_j}\right) = [\cos(\theta_i) - \sin(\theta_i)]^2 A_i(0). \tag{B8}$$

The solution for θ_i is $\theta_i = \arcsin\left(\frac{1+e^{b''_j}}{A_i(0)C} - 1\right)/2$.

We then explain how to use MPQCs to simulate a DBM following the same procedure. As concluded in Lemma 2, any DBM can be transformed into a two-layer RBM by introducing virtual visible neurons. After the transformation, the MPQCs can simulate the corresponding probability distribution by using the abovementioned method, where $O(\text{poly}(N))$ ancillary qubits are introduced to simulate the virtual visible neurons. Since each (virtual) visible neuron corresponds to one ancillary qubit that is applied to a parametrized single-qubit gate, the total number of quantum gates to simulate the probability distribution represented by the RBM and the DBM is at most $O(\text{poly}(N))$. ■

Proof of Lemma 2. The proof of the Lemma 2 leverages the conclusion of Ref. [20]. We illustrate the reformulation in Fig. 16. ■

APPENDIX C: PROOF OF THEOREM 3

Two major differences between TPQCs and MPQCs are that (i) CNOT gates in TPQCs cannot connect any two qubits arbitrarily and (ii) the blocks are replicated based on the structure of the tensor networks. This restriction limits the expressive power of TPQCs.

Proof of Theorem 3. This theorem can be proved by combining Theorem 4 below with Proposition 2. Theorem 4 shows that any IQP circuits with N qubits and $O(\text{poly}(N))$ commuting gates can be transformed into MPQCs with $O(\text{poly}(N))$

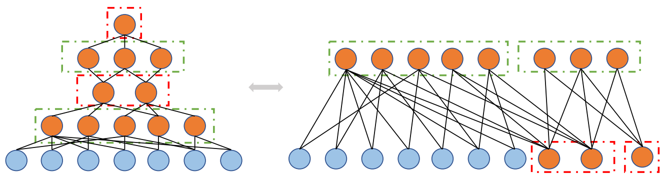


FIG. 16. The reformulation from a DBM to an RBM by introducing virtual visible neurons. The blue and orange nodes refer to the visible and hidden neurons, respectively. The hidden layers with odd numbers, as highlighted by green dash-dotted boxes, form the hidden layer of the reformulated RBM. The hidden layers with even numbers, as highlighted by red dash-dotted boxes, form the virtual visible neurons of the reformulated RBM.

blocks. As stated in Proposition 2, there exist probability distributions, generated by IQP, that cannot be efficiently simulated by classical circuits (including DBMs or long-range RBMs). ■

Theorem 4. MPQCs can efficiently simulate any IQP circuits with N qubits and $O(\text{poly}(N))$ commuting gates, with at most $O(\text{poly}(N))$ blocks, where each block contains no more than $7N$ single-qubit gates and $N-1$ CNOT gates.

Proof. A general IQP circuit is shown in Fig. 17. Before proving that an IQP circuit can be efficiently simulated by MPQCs, we first define the arrangement of quantum gates in each block. As shown in Fig. 18, from left to right in each block, the seven parametrized single-qubit gates are R_X, R_Z, R_X, R_ϕ, R_Z, R_Y, and R_Z, followed by $N-1$ CNOT gates, where the controlled qubit of all of them is the first qubit. For simplicity, we use $(\theta_i, \ldots, \theta_7)$ to represent the composition of the seven parametrized qubit gates.

Note that $H = R_X(\pi/2)R_Z(\pi/2)R_X(\pi/2)$. Hence, it is not hard to see that the initial and final layers of IQP circuits, where N H gates are separately applied to N qubits, can be simulated by choosing parameters $(\pi/2, \pi/2, \pi/2, 0, 0, 0, 0)$ and (0,0,0,0,0,0,0) in the first and second blocks, respectively.

Next we demonstrate that the internal diagonal matrix U_Z can also be simulated using the predefined block structure. Without loss of generality, we assume that the ith circuit depth in U_Z contains M_T T gates and M_{CZ} CZ gates, with $M_T + M_{CZ} \leqslant N$. For example, the colored region in Fig. 17 indicates that $M_T = 2$ and $M_{CZ} = 2$.

Similar to the simulation of H gates, 2 blocks are sufficient to simulate M_T ($M_T \leqslant N$) T gates at the same circuit depth.

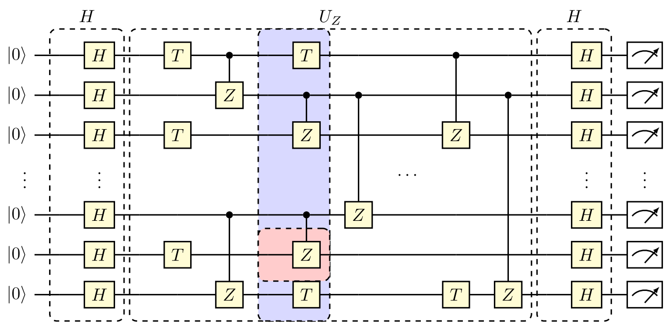


FIG. 17. The arrangement of quantum gates in a general IQP circuit.

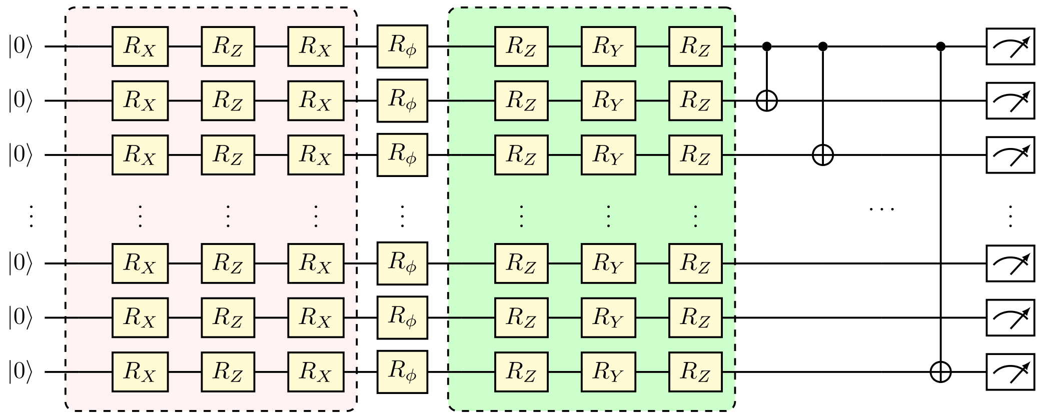


FIG. 18. The arrangement of quantum gates in each block.

Since $T = R_\phi(\pi/4)$, then the T gates can be simulated by application of $(0, 0, 0, \pi/4, 0, 0, 0)$ followed by (0,0,0,0,0,0,0).

We next prove how to use predefined blocks to efficiently simulate a CZ gate. Suppose that a CZ gate is applied to the kth qubit, which is controlled by the jth qubit, with $j \leqslant k$. Since the explicit connection between the two qubits may not exist in the predefined block, we first use 14 blocks to simulate a SWAP gate that switches the jth controlled qubit to the first qubit. We then use 6 blocks to simulate the CZ gate that is applied to the kth qubit and controlled by the first qubit. Last, 14 blocks are employed to simulate another SWAP gate to switch the first control qubit back to its original position. For example, as shown in the left panel of Fig. 19, the CZ gate as indicated by the blue box can be represented by an equivalent circuit controlled by the first qubit.

The central problem in simulating the SWAP operation is how to simulate a single CNOT gate applied arbitrarily to two qubits, since a SWAP gate is composed of three CNOT gates, as illustrate in Fig. 20. The first and third CNOT gate of the SWAP operation can be simulated by 4 blocks. Recall that, in Proposition 1, a single CNOT gate (namely, the CX gate) can be decomposed into $X = A_1B_1C_1$, where $A_1 = R_Z(0)R_Y(\pi/2)$, $B_1 = R_Y(-\pi/2)R_Z(\pi)$, and $C_1 = R_Z(\pi)$. We set all parameters of the first block as 0 except the parameters corresponding to the kth qubit, which are set as $(0, 0, 0, 0, 0, \pi/2, 0)$ to simulate A_1. Next, we set all parameters of the second block as 0 except the parameters corresponding to the kth qubit, which are set as $(0, 0, 0, 0, 0, -\pi/2, \pi)$ to simulate B_1. Then, we set all parameters of the third block as 0 except the parameters corresponding to the kth qubit, which are set as $(0, 0, 0, 0, 0, 0, \pi)$ to simulate C_1. Last, all parameters of the fourth block are set as 0.

Six blocks are required to simulate the second reversed CNOT gate (R-CNOT gate) in the SWAP operation. Since R-CNOT $= (H \otimes H)\text{CNOT}(H \otimes H)$, we use 4 blocks to simulate the $(H \otimes H)$CNOT and then use an extra 2 blocks to simulate the last two Hadamard gates. For the first 4 blocks, the parameters of the first three parametrized gates that are applied to the first and ith qubits are set as $\pi/2$, $\pi/2$, and $\pi/2$, with the aim of simulating two H gates. The remaining parameters of the first 4 blocks follow with the same setting as simulating the CNOT gate as defined above. The last 2 blocks follow a similar setting as simulating the Hadamard layer, where the first three parametrized gates that are applied to the first and ith qubits simulate two H gates and the remaining parameters are set as zero. To conclude, a SWAP gate can be composed of a total of 14 blocks.

Finally, because the CZ gate can be reformulated as $CZ = (\mathbb{I} \otimes H)\text{CNOT}\mathbb{I} \otimes H)$, it can also be simulated by using 6 blocks.

In summary, since H gates, T gates, and CZ gates can be efficiently simulated by using a constant number of blocks, $O(N)$ blocks are sufficient to simulate an IQP circuit with $O(\text{poly}(N))$ T and CZ gates. ■

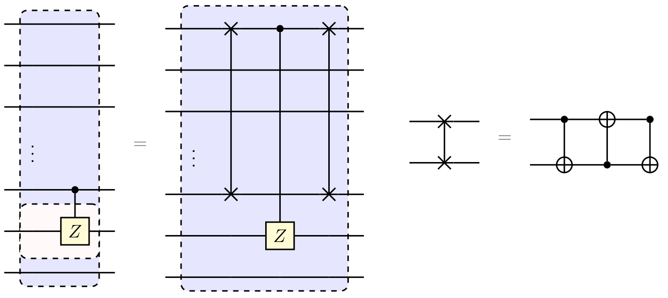


FIG. 19. The left panel illustrates an equivalent circuit described by SWAP operation. The right panel shows the implementation of SWAP by two CNOT gates and one reversed CNOT gate.

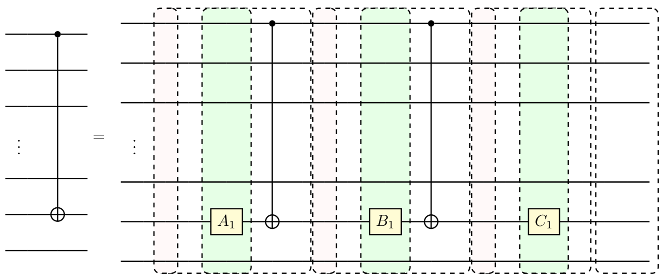


FIG. 20. Simulating a single CNOT gate by using 4 blocks.

[1] A. W. Harrow and A. Montanaro, Nature (London) **549**, 203 (2017).
[2] S. Aaronson and A. Arkhipov, in *Proceedings of The Forty-Third Annual ACM Symposium on the Theory of Computing* (ACM, New York, 2011), p. 333.
[3] D. Shepherd and M. J. Bremner, Proc. R. Soc. London, Ser. A **465**, 1413 (2009).
[4] J. Preskill, Quantum **2**, 79 (2018).
[5] A. Neville, C. Sparrow, R. Clifford, E. Johnston, P. M. Birchall, A. Montanaro, and A. Laing, Nat. Phys. **13**, 1153 (2017).
[6] M. J. Bremner, A. Montanaro, and D. J. Shepherd, Quantum **1**, 8 (2017).
[7] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Nature (London) **549**, 195 (2017).
[8] P. Rebentrost, M. Mohseni, and S. Lloyd, Phys. Rev. Lett. **113**, 130503 (2014).
[9] S. Lloyd, M. Mohseni, and P. Rebentrost, Nat. Phys. **10**, 631 (2014).
[10] P. Wittek, *Quantum Machine Learning: What Quantum Computing Means to Data Mining* (Academic, San Diego, 2014).
[11] N. Wiebe, D. Braun, and S. Lloyd, Phys. Rev. Lett. **109**, 050505 (2012).
[12] G. D. Paparo, V. Dunjko, A. Makmal, M. A. Martin-Delgado, and H. J. Briegel, Phys. Rev. X **4**, 031002 (2014).
[13] V. Dunjko, J. M. Taylor, and H. J. Briegel, Phys. Rev. Lett. **117**, 130501 (2016).
[14] T. G. Dietterich, in *International Workshop on Multiple Classifier Systems* (Springer, Berlin, 2000), pp. 1–15.
[15] M. E. Tipping, J. Mach. Learn. Res. **1**, 211 (2001).
[16] A. Y. Ng and M. I. Jordan, in *Advances in Neural Information Processing Systems 14* (MIT Press, Cambridge, MA, USA, 2002), pp. 841–848.
[17] T. Hofmann, B. Schölkopf, and A. J. Smola, Ann. Stat. **36**, 1171 (2008).
[18] G. E. Hinton and R. R. Salakhutdinov, Science **313**, 504 (2006).
[19] R. Salakhutdinov and H. Larochelle, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (PMLR, Sardinia, Italy, 2010), pp. 693–700.
[20] X. Gao and L.-M. Duan, Nat. Commun. **8**, 662 (2017).
[21] J. Chen, S. Cheng, H. Xie, L. Wang, and T. Xiang, Phys. Rev. B **97**, 085104 (2018).
[22] I. Glasser, N. Pancotti, M. August, I. D. Rodriguez, and J. I. Cirac, Phys. Rev. X **8**, 011006 (2018).
[23] I. Glasser, N. Pancotti, and J. I. Cirac, arXiv:1806.05964.
[24] D.-L. Deng, X. Li, and S. Das Sarma, Phys. Rev. X **7**, 021021 (2017).
[25] J. Carrasquilla and R. G. Melko, Nat. Phys. **13**, 431 (2017).
[26] G. Carleo and M. Troyer, Science **355**, 602 (2017).
[27] J. Liu, H. Shen, Y. Qi, Z. Y. Meng, and L. Fu, Phys. Rev. B **95**, 241104(R) (2017).
[28] E. P. Van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nat. Phys. **13**, 435 (2017).
[29] K. Ch'ng, J. Carrasquilla, R. G. Melko, and E. Khatami, Phys. Rev. X **7**, 031038 (2017).
[30] M. Benedetti, D. Garcia-Pintos, O. Perdomo, V. Leyton-Ortega, Y. Nam, and A. Perdomo-Ortiz, npj Quantum Inf. **5**, 45 (2019).
[31] J.-G. Liu and L. Wang, Phys. Rev. A **98**, 062324 (2018).
[32] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz, Phys. Rev. X **7**, 041052 (2017).
[33] E. Farhi and H. Neven, arXiv:1802.06002.
[34] M. Schuld and N. Killoran, Phys. Rev. Lett. **122**, 040504 (2019).
[35] W. Huggins, P. Patil, B. Mitchell, K. B. Whaley, and E. M. Stoudenmire, Quantum Sci. Technol. **4**, 024001 (2019).
[36] E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. G. Green, and S. Severini, npj Quantum Inf. **4**, 65 (2018).
[37] R. LaRose, A. Tikku, É. O'Neel-Judy, L. Cincio, and P. J. Coles, npj Quantum Inf. **5**, 57 (2019).
[38] E. Farhi and A. W. Harrow, arXiv:1602.07674.
[39] J. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell, N. Didier, E. S. Fried, S. Hong *et al.*, arXiv:1712.05771.
[40] U. Schollwöck, Ann. Phys. **326**, 96 (2011).
[41] E. Bernstein and U. Vazirani, SIAM J. Comput. **26**, 1411 (1997).
[42] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven, Nat. Phys. **14**, 595 (2018).
[43] S. Basu, A. Banerjee, and R. Mooney, in *Proceedings of 19th International Conference on Machine Learning* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002), pp. 27–34.
[44] R. S. Smith, M. J. Curtis, and W. J. Zeng, arXiv:1608.03355.
[45] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, Cognit. Sci. **9**, 147 (1985).
[46] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, Technical Report, California University of San Diego, La Jolla Institute for Cognitive Science, 1985.
[47] V. Nair and G. E. Hinton, in *Proceedings of the 27th International Conference on Machine Learning* (Omnipress, Madison, WI, USA, 2010), pp. 807–814.
[48] R. Orús, Ann. Phys. **349**, 117 (2014).
[49] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University, Cambridge, England, 2010).
[50] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Phys. Rev. A **52**, 3457 (1995).
[51] M. J. Bremner, R. Jozsa, and D. J. Shepherd, Proc. R. Soc. A **467**, 459 (2011).
[52] J. C. Platt, in *Advances in Large Margin Classifiers* (MIT Press, Cambridge, MA, USA, 1999), pp. 61–74.
[53] A. Berlinet and C. Thomas-Agnan, *Reproducing Kernel Hilbert Spaces in Probability and Statistics* (Springer Science & Business Media, New York, 2011).
[54] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Phys. Rev. A **98**, 032309 (2018).
[55] A. Montanaro, npj Quantum Inf. **2**, 15023 (2016).
[56] A. M. Childs, Lecture notes on quantum algorithms, University of Maryland (2017).
[57] A. M. Childs and W. van Dam, Rev. Mod. Phys. **82**, 1 (2010).
[58] P. Cheeseman and J. Stutz, in *Advances in Knowledge Discovery and Data Mining* (American Association for Artificial Intelligence, USA, 1996), pp. 153–180.

[59] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (MIT Press, Cambridge, MA, USA, 2014), pp. 3581–3589.
[60] L. Fei-Fei, R. Fergus, and P. Perona, Comput. Vision Image Understanding **106**, 59 (2007).
[61] D. P. Kingma and M. Welling, arXiv:1312.6114.
[62] Z. Ghahramani, Nature (London) **521**, 452 (2015).
[63] S. Jain, M. White, M. W. Trosset, and P. Radivojac, arXiv:1601.01944.
[64] C. Kemp, T. L. Griffiths, S. Stromsten, and J. B. Tenenbaum, in *Proceedings of the 16th International Conference on Neural Information Processing Systems* (MIT Press, Cambridge, MA, USA, 2003), pp. 257–264.
[65] J. C. Bridgeman and C. T. Chubb, J. Phys. A: Math. Theor. **50**, 223001 (2017).