PHYSICS

A quantum machine learning algorithm based on generative models

X. Gao[1]*, Z.-Y. Zhang[1,2], L.-M. Duan[1,2†]

Quantum computing and artificial intelligence, combined together, may revolutionize future technologies. A significant school of thought regarding artificial intelligence is based on generative models. Here, we propose a general quantum algorithm for machine learning based on a quantum generative model. We prove that our proposed model is more capable of representing probability distributions compared with classical generative models and has exponential speedup in learning and inference at least for some instances if a quantum computer cannot be efficiently simulated classically. Our result opens a new direction for quantum machine learning and offers a remarkable example where a quantum algorithm shows exponential improvement over classical algorithms in an important application field.

INTRODUCTION

A central task in the field of quantum computing is to find applications where a quantum computer could provide exponential speedup over any classical computer (*1*–*3*). Machine learning represents an important field with broad applications where a quantum computer may offer substantial speedup (*4*–*14*). The candidate algorithms with potential exponential speedup in runtime so far rely on efficient quantum solutions of linear algebraic problems (*6*–*11*). These algorithms require quantum random access memory (QRAM) as a critical component in addition to a quantum computer (*4*, *5*, *15*). In a QRAM, the number of required quantum routing operations (*16*) scales up exponentially with the number of qubits in those algorithms (*9*, *15*, *17*). This exponential overhead in resource requirement poses a serious challenge for its experimental implementation and is a caveat for fair comparison with the corresponding classical algorithms (*5*, *18*).

A significant school of thought regarding artificial intelligence is based on generative models (*19*–*21*), which are widely used for probabilistic reasoning as well as for supervised and unsupervised machine learning (*19*–*21*). Generative models try to capture the full underlying probability distributions for the observed data. Compared to discriminative models such as support vector machines and feed-forward neural networks, generative models can express far more complex relations among variables, which makes them broadly applicable but at the same time harder to tackle (*19*, *21*, *22*). Typical generative models include probabilistic graphical models such as the Bayesian nets and the Markov random fields (*19*, *20*, *22*), and generative neural networks such as the Boltzmann machines, the deep belief nets, and the generative adversarial networks (*21*). The probability distributions in these classical generative models can be represented by the so-called factor graphs (*19*, *21*, *22*). In section S1, we give a brief introduction to generative and discriminative models and their applications in machine learning.

Here, we propose a generative quantum machine learning algorithm that offers potential exponential improvement on three key elements of the generative models, that is, the representational power, and the runtimes for learning and inference. In our introduced quantum generative model (QGM), the underlying probability distribution describing correlations in data is generated by measuring a set of observables under a many-body entangled state. In terms of the representational power, we prove that our introduced QGM can efficiently represent any factor graphs, which include almost all the classical generative models as particular cases. Throughout this paper, the word "efficiently" means that the computational or memory cost is bounded by a polynomial function of the size of the problem. Furthermore, we show that the QGM is exponentially more expressive than factor graphs by proving that at least some instances generated by the QGM cannot be efficiently represented by any factor graph with a polynomial number of variables if a widely accepted conjecture in the computational complexity theory holds, that is, the polynomial hierarchy, which is a generalization of the famous P versus NP problem, does not collapse. For learning and inference in our QGM, we propose a general heuristic algorithm using a combination of techniques such as tensor networks, construction of parent Hamiltonians for many-body entangled states, and recursive quantum phase estimation. Although it is unreasonable to expect that the proposed quantum algorithm has polynomial scaling in runtime in all the cases (as this would imply that a quantum computer could efficiently solve any NP problems, an unlikely result), we prove that, at least for some instances, our quantum algorithm has exponential speedup over any classical algorithms, assuming that a quantum computer cannot be efficiently simulated in general by classical computers, a conjecture that is believed to hold. Very recently, a different generative model for quantum machine learning was proposed on the basis of a quantum version of the Boltzmann machine (*23*). The quantum advantages compared with the classical generative models, however, still remain unknown for that model in terms of the representational power and the runtimes for learning and inference.

The intuition for quantum speedup in our algorithm can be understood as follows: The purpose of generative machine learning is to model any data generation process in nature by finding the underlying probability distribution. As nature is governed by the law of quantum mechanics, it is too restrictive to assume that the real-world data can always be modeled by an underlying probability distribution as in classical generative models. Instead, in our QGM, correlations in data are parameterized by the underlying probability amplitudes of a many-body entangled state. As the interference of quantum probability amplitudes can lead to phenomena much more complex than those from classical probabilistic models, it is possible to achieve marked improvement in our QGM under certain circumstances.

[1]Center for Quantum Information, IIIS, Tsinghua University, Beijing 100084, PR China. [2]Department of Physics, University of Michigan, Ann Arbor, MI 48109, USA.
*Present address: Department of Physics, Harvard University, MA 02138, USA.
†Corresponding author. Email: lmduan@tsinghua.edu.cn

RESULTS

Factor graphs and our QGM

We start by defining factor graphs and our QGM. Direct characterization of a probability distribution of n binary variables has an exponential cost of 2^n. A factor graph, which includes many classical generative models as special cases, is a compact way to represent n-particle correlation (*21*, *22*). As shown in Fig. 1A, a factor graph is associated with a bipartite graph where the probability distribution can be expressed as a product of positive correlation functions of a constant number of variables. Here, without loss of generality, we have assumed constant-degree graphs, where the maximum number of edges per vertex is bounded by a constant.

Our QGM is defined on a graph state $|G\rangle$ of m qubits. As a powerful tool to represent many-body entangled states, the graph state $|G\rangle$ is defined on a graph G (*24*), where each vertex in G is associated with a qubit. To prepare the graph state $|G\rangle$, all the qubits are initialized to the state $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ at the beginning, where $|0\rangle$, $|1\rangle$ denote the qubit computational basis vectors, and then a controlled phase flip gate is applied to all the qubit pairs connected by an edge in the graph G. We then introduce the following transformation to the graph state $|G\rangle$

$$|Q\rangle \equiv M_1 \otimes \cdots \otimes M_m |G\rangle \tag{1}$$

where M_i denotes an invertible (in general, nonunitary) 2×2 matrix applied on the Hilbert space of qubit i. Note that for general nonunitary M_i, the state $|Q\rangle$ cannot be directly prepared from the state $|G\rangle$ efficiently, and in our following learning algorithm, we actually first transform $|Q\rangle$ into a tensor network state and then use a combination of techniques to prepare this tensor network state. From m vertices of the graph G, we choose a subset of n qubits as the visible units and measure them in the computational basis $\{|0\rangle, |1\rangle\}$. The measurement outcomes sample from a probability distribution $Q(\{x_i\})$ of n binary variables $\{x_i, i = 1, 2, \cdots n\}$ (the other $m - n$ hidden qubits are just traced over to give the reduced density matrix). Given a graph G and the subset of visible vertices, the distribution $Q(\{x_i\})$ defines our QGM, which is parameterized efficiently by the parameters in the matrices M_i. As shown in Fig. 1C, the pure entangled quantum state $|Q\rangle$ can be written as a special tensor network state. We define our model in this form for two reasons: First, the probability distribution $Q(\{x_i\})$ needs to be general enough to include all the factor graphs; second, for the specific form of the state $|Q\rangle$ introduced in this paper, the parameters in this model can be conveniently trained by a quantum algorithm on any given dataset.

Representational power of our QGM

Representational power is a key property of a generative model. It sets the ultimate limit to which the model might be useful. The more probability distributions a generative model can efficiently represent, the wider applications it can potentially have. The representational power is also closely related to the so-called generalization ability of a probabilistic model (see section S2). In this subsection, we prove that the QGM introduced above is exponentially more expressive in terms of the representation power compared with the classical factor graphs. This result is more accurately described by theorems 1 and 2. First, we show that any factor graph can be viewed as a special case of our QGM by the following theorem.

Theorem 1

The QGM defined above can efficiently represent probability distributions from any constant-degree factor graphs with arbitrary precision. Concretely, the number of parameters in the QGM can be bounded by $O(2^k s)$, where s is the number of function nodes in the factor graph and k is the degree of the graph.

As probabilistic graphical models and generative neural networks can all be reduced to constant-degree factor graphs (*22*), the above theorem shows that our proposed QGM is general enough to include those probability distributions in widely used classical generative models. In section S3, we show how to reduce typical classical generative models to factor graphs with a bounded degree. Actually, by reducing the degree of the factor graph through equivalence transformation (see fig. S4), we can consider the factor graphs with degree $k = 3$ without loss of generality.

Proof of theorem 1: First, in any factor graph with its degree bounded by a constant k, each of the functions (denoted by square nodes in Fig. 1) has at most k variables and therefore, by the universal

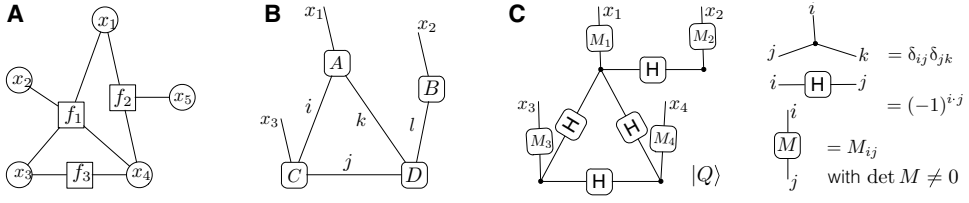

Fig. 1. Classical and quantum generative models. (**A**) Illustration of a factor graph, which includes widely used classical generative models as its special cases. A factor graph is a bipartite graph where one group of the vertices represents variables (denoted by circles) and the other group of vertices represents positive functions (denoted by squares) acting on the connected variables. The corresponding probability distribution is given by the product of all these functions. For instance, the probability distribution in (A) is $p(x_1, x_2, x_3, x_4, x_5) = f_1(x_1, x_2, x_3, x_4) f_2(x_1, x_4, x_5) f_3(x_3, x_4)/Z$, where Z is a normalization factor. Each variable connects to at most a constant number of functions that introduce correlations in the probability distribution. (**B**) Illustration of a tensor network state. Each unshared (shared) edge represents a physical (hidden) variable, and each vertex represents a complex function of the variables on its connected edges. The wave function of the physical variables is defined as a product of the functions on all the vertices, after summation (contraction) of the hidden variables. Note that a tensor network state can be regarded as a quantum version of the factor graph after partial contraction (similar to the marginal probability in the classical case), with positive real functions replaced by complex functions. (**C**) Definition of a QGM introduced in this paper. The state $|Q\rangle$ represented here is a special kind of tensor network state, with the vertex functions fixed to be three types as shown on the right side. Without the single-qubit invertible matrix M_i, which contains the model parameters, the wave function connected by Hadamard and identity matrices just represent a graph state. To get a probability distribution from this model, we measure a subset of n qubits (among total m qubits corresponding to the physical variables) in the computational basis under this state. The unmeasured $m - n$ qubits are traced over to get the marginal probability distribution $P(\{x_i\})$ of the measured n qubits. We prove in this paper that the $P(\{x_i\})$ is general enough to include probability distributions of all the classical factor graphs and special enough to allow a convenient quantum algorithm for the parameter training and inference.

approximation theorem (*25*), it can be approximated arbitrarily well with a restricted Boltzmann machine with k visible variables and 2^k hidden variables. In section S4, we give an accurate description of the universal approximation theorem and an explanation of the restricted Boltzmann machine. As the degree k of the factor graph can be reduced to a very small number (such as $k = 3$), the number of hidden variables 2^k only represents a moderate constant cost, which does not increase with the system size. In a restricted Boltzmann machine, only the visible and the hidden variables are connected by the two-variable correlator that takes the generic form $f(x_1, x_2) = e^{ax_1x_2+bx_1+cx_2}$, where x_1, x_2 denote the binary variables and a, b, c are real parameters. The representation precision using the universal approximation theorem does not affect the number of variables but only depends on the range of the parameters a, b, c, e.g., arbitrary prevision can be achieved if a, b, c are allowed to vary over the whole real axis. As $Q(\{x_i\})$ has a similar factorization structure as the factor graph after measuring the visible qubits x_i under a diagonal matrix M_i (see Fig. 2), it is sufficient to show that each correlator $f(x_1, x_2)$ can be constructed in the QGM. This construction can be achieved by adding one hidden variable (qubit) j with the invertible matrix M_j between two visible variables x_1 and x_2. As shown in Fig. 2, $Q(\{x_1, x_2\})$ can be calculated by contracting two copies of a diagram from Fig. 1C as $\langle Q|x_1, x_2\rangle\langle x_1, x_2|Q\rangle$. To make this equal to arbitrary correlation $f(x_1, x_2)$ between the two variables x_1 and x_2, we first take the matrix $M_j^\dagger M_j = \lambda_1|+\rangle\langle+| + \lambda_2|-\rangle\langle-|$, where $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$, and λ_1 and λ_2 terms account for positive and negative correlations, respectively. Here, positive (negative) correlation means that two variables tend to be the same (different). The convex hull of these positive and negative correlations generates any symmetric correlation of the form $e^{ax_1x_2-ax_1/2-ax_2/2}$ between the two variables x_1 and x_2. Then, we take additional single-qubit matrix D_1 and D_2 to account for the remaining difference between $-ax_1/2 - ax_2/2$ and $bx_1 + cx_2$ in the correlation function $f(x_1, x_2)$. Specifically, we take D_1 and D_2 to be diagonal with eigenvalues $\sqrt{d_1(x_1)}$ and $\sqrt{d_2(x_2)}$, respectively. As shown in Fig. 2 (C and D), the correlator between x_1 and x_2 in the QGM is then given by $d_1(x_1)d_2(x_2)[\lambda_1\delta_{x_1x_2} + \lambda_2(1 - \delta_{x_1x_2})]/2$, and one simple solution with $d_1(0) = d_2(0) = \lambda_1/2 = 1$ and $d_1(1) = e^{b + a/2}$, $d_2(1) = e^{c + a/2}$, $\lambda_2 = 2e^{-a/2}$ makes it equal to the correlation function $f(x_1, x_2)$ with arbitrary coefficients a, b, c. From the above proof, each function node introduces the number of parameters of the order $O(2^k)$ independent of the representation precision. For a factor graph of s function nodes (note that s is upper bounded by n^k, where n is the number of variables), the total number of parameters is of the order $O(2^k s)$. This completes the proof.

Furthermore, we can show that the QGM is exponentially more expressive than factor graphs in representing some probability distributions. This is summarized by the following theorem.

Theorem 2

If the polynomial hierarchy in the computational complexity theory does not collapse, there exist probability distributions that can be efficiently represented by a QGM but cannot be efficiently represented even under approximation by conditional probabilities from any classical generative models that are reducible to factor graphs.

The rigorous proof of this theorem is lengthy and technical, and we thus include it in section S5. Here, we briefly summarize the major idea of the proof: In the QGM, we construct the underlying distribution by adding up the probability amplitudes (complex numbers), while in the factor graphs, we only add up probabilities (positive real numbers). The complexity of these two addition problems turns out to be different: Adding the positive probabilities up to get the normalization factor in

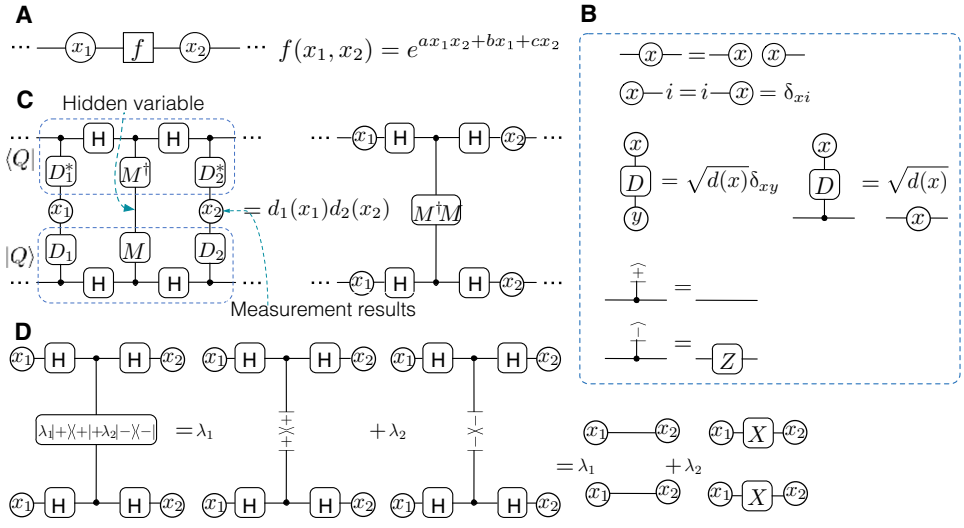

Fig. 2. Efficient representation of factor graphs by the QGM. (**A**) General form of correlation functions of two binary variables in a factor graph, with parameters a, b, c being real. This correlation acts as the building block for general correlations in any factor graphs by use of the universal approximation theorem (*25*). (**B**) Notations of some common tensors and their identities: D is a diagonal matrix with diagonal elements $\sqrt{d(x)}$ with $x = 0, 1$; Z is the diagonal Pauli matrix diag(1, −1); and $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$. (**C**) Representation of the building block correlator $f(x_1, x_2)$ in a factor graph [see (A)] by the QGM with one hidden variable (unmeasured) between two visible variables x_1, x_2 (measured in the computational basis). As illustrated in this figure, $f(x_1, x_2)$ can be calculated as contraction of two copies of a diagram from Fig. 1C as $\langle Q|x_1, x_2\rangle\langle x_1, x_2|Q\rangle$, where x_1, x_2 are measurement results of the two visible variables. We choose the single-bit matrix D_1, D_2 to be diagonal with $D_1 = \mathrm{diag}(\sqrt{d_1(0)}, \sqrt{d_1(1)})$ and $D_2 = \mathrm{diag}(\sqrt{d_2(0)}, \sqrt{d_2(1)})$. For simplification of this graph, we have used the identities shown in (B). (**D**) Further simplification of the graph in (C), where we choose the form of the single-bit matrix $M^\dagger M$ acting on the hidden variable to be $M^\dagger M = \lambda_1|+\rangle\langle+| + \lambda_2|-\rangle\langle-|$ with positive eigenvalues λ_1, λ_2. We have used the identities in (B) and the relation $HZH = X$, where X (H) denotes the Pauli (Hadamard) matrix, respectively. By solving the values of λ_1, λ_2, $d_1(x_1)$, $d_2(x_2)$ in terms of a, b, c (see the proof of theorem 1), this correlator of the QGM exactly reproduces the building block correlator $f(x_1, x_2)$ of the factor graph.

factor graphs is a nondecreasing process and its complexity is at a lower level of the polynomial hierarchy in the computational complexity theory according to the Stockmeyer's theorem (*26*). In contrast, the summation of complex numbers to get the normalization factor in the QGM involves marked oscillating behaviors, similar to the sign problem in the quantum Monte Carlo method (*27*), which puts its computational complexity at a much higher level. Because of this difference in the computational complexity, by adding up probability amplitudes in the QGM, we can generate a much wider type of distributions, which are hard to represent by factor graphs through adding up the positive probabilities. To represent some distributions generated by the QGM with factor graphs, the number of parameters in the factor graphs is required to scale up exponentially with the size of the QGM, and therefore, there is an exponential gap between the representational power of these two models if the polynomial hierarchy in the computational complexity theory does not collapse.

Theorems 1 and 2 show that the QGM is much more powerful to represent probability distributions compared with the classical factor graphs. Representational power is important for the success of a machine learning model. It is closely related to the generalization ability. Generalization ability characterizes the ability of a model to make a good prediction for new data by using as few training data as possible. Higher representational power usually implies better generalization ability in practice (*21*, *28*). Similar to the principle of Occam's razor in physics, a good choice for the machine learning model is the one with a minimum number of parameters but still can well explain the observed training data. The representational power characterizes this feature by representing a wide class of distributions using as few parameters as possible. Our QGM can efficiently represent some probability distributions that are out of the reach of the classical factor graphs, as the latter may need an exponentially large number of parameters for the representation. This suggests that our proposed QGM is a good choice for the machine learning model in terms of the representational power.

As any factor graphs can be efficiently represented by our QGM, this suggests that we should not expect that an arbitrary state $|Q\rangle$ can be prepared efficiently with a quantum computer. If we can prepare arbitrary $|Q\rangle$ efficiently, we can efficiently solve the inference problem for any factor graph, which is known to be an NP-hard problem and unlikely to be fully solved with a quantum computer. The detailed proof of this statement is included in section S6. For applications in generative learning, we have the freedom to choose the topology and the parameter form of $|Q\rangle$ and only need to use a subset of states $|Q\rangle$ that can be efficiently prepared. Normally, we first construct the parent Hamiltonian for the state $|Q\rangle$ (see section S7) and then use the tensor network formalism for its preparation, inference, and learning, as will be explained in the next section.

A quantum algorithm for inference and training of our QGM

For a generative model to be useful for machine learning, apart from its representational power and generalization ability, we also need to have an efficient algorithm for both training and inference. Training is the process of tuning parameters in a model to represent the probability distribution as close as possible to that of the dataset. This usually involves minimization of a cost function, which determines how close these two distributions are. Once we have a trained model, we make inference to extract useful information for analysis or prediction. Most of inference tasks can be reduced to computing conditional probability $\sum_y p(x, y|z)$ (*22*), where x, y, z denote different variable sets. For training, we choose to minimize the Kullback-Leibler (KL) divergence $D(q_d||p_\theta) = -\sum_v q_d(v)\log(p_\theta(v)/q_d(v))$ between q_d, the distribution of the given data sample, and p_θ, the distribution of the generative model, with the whole parameter set denoted by θ. Typically, one minimizes $D(q_d||p_\theta)$ by optimizing the model parameters θ using the gradient descent method (*21*). The θ-dependent part $D(\theta)$ of $D(q_d||p_\theta)$ can be expressed as $-\sum_{v\in \text{data set}} \log p_\theta(v)/M$, where M denotes the total number of data points.

In our QGM, both the conditional probability $\sum_y p(x, y|z)$ and the gradient of the KL divergence $\partial_\theta D(\theta)$ can be conveniently calculated using the structure of state $|Q\rangle$ defined in Fig. 1. We first define a tensor network state $|Q(z)\rangle \equiv (I \otimes \langle z|)|Q\rangle$ by projecting the variable set z to the computational basis. As shown in section S8, the conditional probability can be expressed as

$$\sum_y p(x, y|z) = \frac{\langle Q(z)|O|Q(z)\rangle}{\langle Q(z)|Q(z)\rangle} \quad (2)$$

which is the expectation value of the operator $O = |x\rangle\langle x|$ under the state $|Q(z)\rangle$. For inference problems, we typically only need to know the distribution $p(x|z) = \sum_y p(x, y|z)$ in the label x by a constant precision. This can be conveniently achieved by measuring the operator O under the state $|Q(z)\rangle$. The measurement results correspond to an importance sampling and automatically give the dominant x labels for which the probability distribution $p(x|z)$ has notable nonzero values. Similarly, we show in section S8 that $\partial_\theta D(\theta)$ can be reduced to a combination of terms taking the same form as Eq. 2, with operator O replaced by $O_1 = (\partial_{\theta_i} M_i)M_i^{-1} + \text{H.c.}$ or $O_2 = |v_i\rangle\langle v_i|(\partial_{\theta_i} M_i)M_i^{-1} + \text{H.c.}$, where θ_i denotes a specific parameter in the invertible matrix M_i, v_i is the qubit of data v corresponding to the variable x_i, and H. c. stands for the Hermitian conjugate term. The variable z in this case takes the value of v (or v excluding v_i) expressed in a binary string.

With the above simplification, training or inference in the QGM is reduced to preparation of the tensor network state $|Q(z)\rangle$. With an algorithm similar to the one in (*29*), we use recurrent quantum phase estimation to prepare the state $|Q(z)\rangle$. For this purpose, we first construct the parent Hamiltonian $H(z)$, which has a unique ground state given by $|Q(z)\rangle$ with zero eigenenergy in the generic case as shown in (*30*). The procedure is illustrated in Fig. 3, where the variables $z = \{z_i\}$ are grouped as in Fig. 3A. In this case, the corresponding local tensors in the tensor network state $|Q(z)\rangle$ are all easy to compute and of a constant degree. The parent Hamiltonian $H(z)$ is constructed through contraction of these local tensors as shown in Fig. 3 (C and D) (*30*).

By constructing the parent Hamiltonian for the tensor network state, the quantum algorithm for training and inference in the QGM is realized through the following steps:

Step 1: Construct a classical description of a series of tensor network states $\{|Q_t\rangle\}$ with $t = 0, 1,..., n$, as shown in Fig. 3C, by reduction from $|Q_n\rangle = |Q(z)\rangle$. The initial state $|Q_0\rangle$ is a product state $|0\rangle^{\otimes n}$, and $|Q_t\rangle$ is constructed from $|Q_{t-1}\rangle$ by adding one more tensor in $|Q(z)\rangle$ that is not contained in $|Q_{t-1}\rangle$ and setting the uncontracted virtual indices as 0.

Step 2: Construct a classical description of the parent Hamiltonian H_t for each $|Q_t\rangle$ with the method illustrated in Fig. 3 (B to D) (*30*).

Step 3: On a quantum computer, starting from $|Q_0\rangle$, we prepare $|Q_1\rangle,..., |Q_n\rangle$ sequentially. Suppose that we have prepared $|Q_{t-1}\rangle$. The following substeps will prepare $|Q_t\rangle$ based on the recursive quantum phase estimation to measure the eigenstates of the parent Hamiltonian H_t.

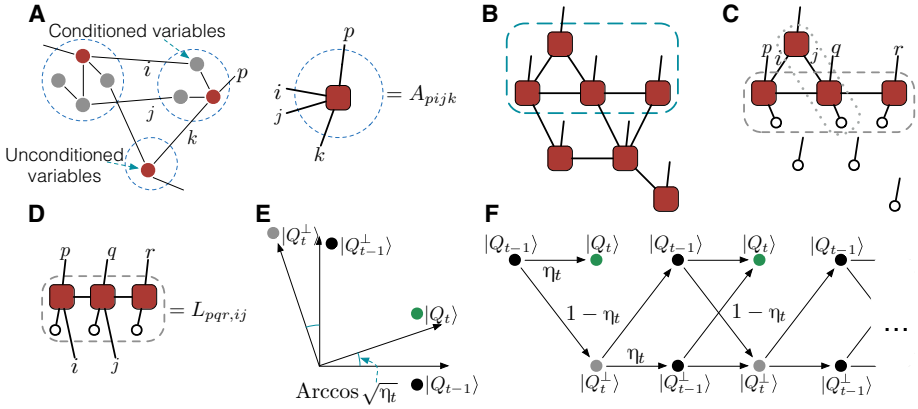

Fig. 3. Illustration of our training algorithm for the QGM. (**A**) Training and inference of the QGM are reduced to measuring certain operators under the state $|Q(z)\rangle$. The key step of the quantum algorithm is therefore to prepare the state $|Q(z)\rangle$, which is achieved by recursive quantum phase estimation of the constructed parent Hamiltonian. The variables in the set z whose values are specified are called conditioned variables, whereas the other variables that carry the binary physical index are called unconditioned variables. We group the variables in a way such that each group contains only one unconditioned variable and different groups are connected by a small constant number of edges (representing virtual indices or hidden variables). Each group then defines a tensor with one physical index (denoted by p) and a small constant number of virtual indices (denoted by i, j, k in the figure). (**B**) Tensor network representation of $|Q(z)\rangle$, where a local tensor is defined for each group specified in (A). (**C**) Tensor network representation of $|Q_t\rangle$, where $|Q_t\rangle$ are the series of states reduced from $|Q_z\rangle$. In each step of the reduction, one local tensor is moved out. The moved-out local tensors are represented by the unfilled circles, each carrying a physical index set to 0. For the edges between the remaining tensor network and the moved-out tensors, we set the corresponding virtual indices to 0 (represented by the unfilled circles). (**D**) Construction of the parent Hamiltonian. The figure shows how to construct one term in the parent Hamiltonian, which corresponds to a group of neighboring local tensors such as those in the dashed box in (C). After contraction of all virtual indices among the group, we get a tensor $L_{pqr,ij}$, which defines a linear map L from the virtual indices i, j to the physical indices p, q, r. As the indices i, j take all the possible values, the range of this mapping L spans a subspace range(L) in the Hilbert space $H_{p,q,r}$ of the physical indices p, q, r. This subspace has a complementary orthogonal subspace inside $H_{p,q,r}$ denoted by comp(L). The projector to the subspace comp(L) then defines one term in the parent Hamiltonian, and by this definition, $|Q_t\rangle$ lies in the kernel space of this projector. We construct each local term with a group of neighboring tensors. Each local tensor can be involved in several Hamiltonian terms [as illustrated in (C) by the dashed box and the dotted box]; thus, some neighboring groups have nonempty overlap, and they generate terms that, in general, do not commute. By this method, one can construct the parent Hamiltonian whose ground state generally uniquely defines the state $|Q_t\rangle$ (*30*) [technically, this step requires the tensor network to be injective (*29*), a condition that is generically satisfied for random choice of the tensor networks (*30*)]. (**E**) States involved in the evolution from $|Q_{t-1}\rangle$ to $|Q_t\rangle$ by quantum phase estimation applied on their parent Hamiltonians. $|Q_{t-1}^{\perp}\rangle$, $|Q_t^{\perp}\rangle$ represent the states orthogonal to $|Q_{t-1}\rangle$, $|Q_t\rangle$, respectively, inside the two-dimensional (2D) subspace spanned by $|Q_{t-1}\rangle$ and $|Q_t\rangle$. The angle between $|Q_t\rangle$ and $|Q_{t-1}\rangle$ is determined by the overlap $\eta_t = |\langle Q_t|Q_{t-1}\rangle|^2$. (**F**) State evolution under recursive application of the quantum phase estimation algorithm. Starting from the state $|Q_{t-1}\rangle$, we always stop at the state $|Q_t\rangle$, following any branch of this evolution, where η_t and $1-\eta_t$ denote the probabilities of the corresponding outcomes.

Substep 1: We use the quantum phase estimation algorithm to measure whether the eigenenergy of the parent Hamiltonian H_t is zero (*31*), which implements a projective measurement to the corresponding eigenspaces $\{|Q_t\rangle\langle Q_t|, I-|Q_t\rangle\langle Q_t|\}$ of H_t with zero and nonzero eigenenergies, respectively (see Fig. 3E). Similarly, we can implement the projective measurement $\{|Q_{t-1}\rangle\langle Q_{t-1}|, I-|Q_{t-1}\rangle\langle Q_{t-1}|\}$ by the quantum phase estimation using the parent Hamiltonian H_{t-1}.

Substep 2: Starting from $|Q_{t-1}\rangle$, we perform the projective measurement $\{|Q_t\rangle\langle Q_t|, I-|Q_t\rangle\langle Q_t|\}$. If the result is $|Q_t\rangle$, we succeed and skip the following substeps. Otherwise, we get $|Q_t^{\perp}\rangle$ lying in the plane spanned by $|Q_{t-1}\rangle$ and $|Q_t\rangle$.

Substep 3: We perform the projective measurement $\{|Q_{t-1}\rangle\langle Q_{t-1}|, I-|Q_{t-1}\rangle\langle Q_{t-1}|\}$ on the state $|Q_t^{\perp}\rangle$. The result is either $|Q_{t-1}\rangle$ or $|Q_{t-1}^{\perp}\rangle$.

Substep 4: We perform the projective measurement $\{|Q_t\rangle\langle Q_t|, I-|Q_t\rangle\langle Q_t|\}$ again. We either succeed in getting $|Q_t\rangle$, with probability $\eta_t = |\langle Q_t|Q_{t-1}\rangle|^2$, or have $|Q_t^{\perp}\rangle$. In the latter case, we go back to the substep 3 and continue until success.

Step 4: After successful preparation of the state $|Q(z)\rangle$, we measure the operator O (for inference) or O_1, O_2 (for training), and the expectation value of the measurement gives the required conditional probability (for inference) or the gradient of the KL divergence (for training).

In the following, we analyze the runtime of this algorithm for getting the correct answer. Step 1 computes n tensor network states with runtime $\mathcal{O}(n)$ because we only add one tensor with a constant number of variables in constructing each of n states $|Q_t\rangle$ with t from 1 to n. Step 2 computes n Hamiltonians with runtime $\mathcal{O}(n^{c+1})$ for some constant c, which denotes the number of nodes involved for constructing a local term of the parent Hamiltonians with the method shown in Fig. 3D [thus, there are $\mathcal{O}(n^c)$ local terms]; H_t constructed by this method generically has the unique ground state $|Q_t\rangle$ (*30*) and the larger c is, the more likely there is no ground-state degeneracy.

A quantum computer is required for implementation of step 3. When this step gives the correct answer, the correctness of this algorithm is guaranteed. Let Δ_t denote the energy gap for the parent Hamiltonian H_t. Because we require the precision of the quantum phase estimation to be Δ to distinguish the ground state from the excited states of H_t, the quantum phase estimation algorithm in substep 1 has runtime complexity $\tilde{\mathcal{O}}(n^{2c}/\Delta)$ for the QGM defined on a constant-degree graph (*32*–*34*), where $\Delta = \min_t \Delta_t$ and $\tilde{\mathcal{O}}(\cdot)$ suppresses slowly growing factors such as $(n/\Delta)^{o(1)}$ (*6*, *30*). The key idea of step 3 is that the subspace spanned by $\{|Q_t\rangle, |Q_{t-1}\rangle\}$, as shown in Fig. 3E, is the invariant subspace of all the projective operators involved. It can be seen from the evolution tree of step 3 (shown in Fig. 3F) that the construction of $|Q_t\rangle$ from $|Q_{t-1}\rangle$ can terminate within $2k+1$ iterations with probability

$$\mathrm{term}(k) = 1-(1-\eta_t)\left(\eta_t^2+(1-\eta_t)^2\right)^k \tag{3}$$

This implies that within s steps of iteration, the probability of failure to get $|Q_t\rangle$ is smaller than $p_{\mathrm{fail}}(s) < s\eta e/2$ for all t, where e is the base of the

natural logarithm and $\eta = \min_t \eta_t$ (*30*). Denote the probability of failure to get $|Q_n\rangle$ as ε, then we require $(1 - p_{\text{fail}})^n \geq 1 - \varepsilon$, so $s > n/(2\eta\varepsilon\varepsilon)$ is sufficient. Thus, the runtime of preparing $|Q_t\rangle$ in step 3 is $\tilde{\mathcal{O}}(n^{2c+1}/(\eta\Delta\varepsilon))$. Iterating from 1 to n, we can get $|Q(z)\rangle$ with a probability of at least $1 - \varepsilon$. Measuring operator O over $|Q(z)\rangle$ $1/\delta$ times, the result will be an approximation of the conditional probability to an additive error δ as shown in Eq. 2.

Therefore, the total runtime of approximating the conditional probability is

$$T = \tilde{\mathcal{O}}(n^{2c+2}/(\eta\Delta\varepsilon\delta)) \tag{4}$$

The gap Δ_t and the overlap η_t depend on the topology of the graph G, the variable set z, and the parameters of the matrices M_i. If these two quantities are bounded by poly (n) for all the steps t from 1 to n, this quantum algorithm will be efficient with its runtime bounded by poly (n).

Gradient of the KL divergence for each parameter is constituted by several simple terms (see section S8), where each term is similar to the expression of the conditional probability except that the operator O is replaced by O_1 or O_2. The number of terms is proportional to M, which is the number of training data for a full gradient descent method, or the batch size using the stochastic gradient descent method, which usually requires only $O(1)$ data for calculation of the gradient (*22*).

Quantum speedup

Although we do not expect the above algorithm to be efficient in the worst case [even for the simplified classical generative model such as the Bayesian network, the worst-case complexity is at least NP hard (*35*)], we know that the QGM with the above heuristic algorithm will provide exponential speedup over classical generative models for some instances. In section S9, we give a rigorous proof that our inference and learning algorithm has an exponential speedup over any classical algorithm for some instances under a well-accepted conjecture in the computational complexity theory. We arrive at the following theorem.

Theorem 3

There exist instances for computing the conditional probabilities or the gradients of the KL divergence in our QGM to an additive error $1/\text{poly}(n)$ such that (i) our algorithm can achieve those calculations in a polynomial time and (ii) any classical algorithm cannot accomplish them in a polynomial time if universal quantum computing cannot be efficiently simulated by a classical computer.

The proof of this theorem is lengthy and can be found in the Supplementary Materials. Here, we briefly summarize the major idea. We construct a specific $|Q(z)\rangle$ that corresponds to the tensor network representation of the history state for universal quantum circuits rearranged into a 2D spatial layout. The history state is a powerful tool in quantum complexity theory (*36*). It is a superposition state of the entire computing history of an arbitrary universal quantum circuit

$$\left|\psi_{\text{history}}\right\rangle = \frac{1}{\sqrt{T+1}} \sum_{t=0}^{T} |t\rangle \otimes V_t \cdots V_1 |0\rangle^{\otimes m} \tag{5}$$

where T is the number of total gates in a quantum circuit, which is assumed to be a polynomial function of the total qubit number m; $|t\rangle$ is a quantum state encoding the step counter t; $|0\rangle^{\otimes m}$ is the input state of the circuits with m qubits; and V_t is the tth gate. A similar type of the history state has been used before to prove the QMA (quantum Merlin-Arthur) hardness for spin systems in a 2D lattice (*37*). For this specific $|Q(z)\rangle$, we prove that both the gap Δ_t and the overlap η_t scale as $1/\text{poly}(n)$ for all the steps t, by calculating the parent Hamiltonian of $|Q_t\rangle$ with proper grouping of the local tensors. Our heuristic algorithm for training and inference therefore can be accomplished in a polynomial time if all the conditional probabilities that we need can be generated from those history states. On the other hand, our specific state $|Q(z)\rangle$ encodes universal quantum computation through representation of the history state, so it not only should include a large set of useful distributions but also cannot be achieved by any classical algorithm in a polynomial time if universal quantum computation cannot be efficiently simulated by a classical computer.

Here, we focus on the complexity theoretical proofs to support the quantum advantages of our proposed QGM in terms of the representational power and the runtimes for learning and inference. Before ending the paper, we briefly discuss the possibility to apply the model here for solving practical problems in machine learning. We mention two examples in section S10 and show a sketchy diagram there on how to embed our QGM model into a typical machine learning pipeline. The first example is on classification of handwritten digits, which is a typical task for supervised learning. The second example is on completion of incomplete pictures, a typical unsupervised learning task. The diagrams in section S10 illustrate the basic steps for these two examples, which require a combination of both classical and quantum computing. The numerical test for these two examples requires efficient classical simulation of the quantum computing part, which puts restrictions on the topology of the underlying graph states, requires a lot of numerical optimizations, and is still an ongoing project. The efficiency and runtimes for real-world problems need to be tested with a quantum computer or classical simulation of sufficiently large quantum circuits, which, of course, is not a easy task and remains an interesting question for the future.

Summary

In summary, we have introduced a QGM for machine learning and proven that it offers potential exponential improvement in the representational power over widely used classical generative models. We have also proposed a heuristic quantum algorithm for training and making inference on our model, and proven that this quantum algorithm offers exponential speedup over classical algorithms at least for some instances if quantum computing cannot be efficiently simulated by a classical computer. Our result combines the tools of different areas and generates an intriguing link between quantum many-body physics, quantum computational complexity theory, and the machine learning frontier. This result opens a new route to apply the power of quantum computation to solving the challenging problems in machine learning and artificial intelligence, which, apart from its fundamental interest, may have wide applications in the future.

SUPPLEMENTARY MATERIALS

Supplementary material for this article is available at http://advances.sciencemag.org/cgi/content/full/4/12/eaat9004/DC1

Section S1. A brief introduction to generative models
Section S2. Representational power and generalization ability
Section S3. Reduction of typical generative models to factor graphs
Section S4. Universal approximation theorem
Section S5. Proof of theorem 2

REFERENCES AND NOTES

1. P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **41**, 303–332 (1999).
2. R. P. Feynman, Simulating physics with computers. *Int. J. Theor. Phys.* **21**, 467–488 (1982).
3. S. Lloyd, Universal quantum simulators. *Science* **273**, 1073–1078 (1996).
4. J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd, Quantum machine learning. *Nature* **549**, 195–202 (2017).
5. C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, L. Wossnig, Quantum machine learning: A classical perspective. arXiv:1707.08561 [quant-ph] (26 July 2017).
6. A. W. Harrow, A. Hassidim, S. Lloyd, Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **103**, 150502 (2009).
7. N. Wiebe, D. Braun, S. Lloyd, Quantum algorithm for data fitting. *Phys. Rev. Lett.* **109**, 050505 (2012).
8. S. Lloyd, M. Mohseni, P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning. arXiv:1307.0411 [quant-ph] (1 July 2013).
9. S. Lloyd, M. Mohseni, P. Rebentrost, Quantum principal component analysis. *Nat. Phys.* **10**, 631–633 (2014).
10. P. Rebentrost, M. Mohseni, S. Lloyd, Quantum support vector machine for big data classification. *Phys. Rev. Lett.* **113**, 130503 (2014).
11. I. Cong, L. Duan, Quantum discriminant analysis for dimensionality reduction and classification. *New J. Phys.* **18**, 073011 (2016).
12. F. G. S. L. Brandão, K. Svore, Quantum speed-ups for semidefinite programming. arXiv:1609.05537 [quant-ph] (18 September 2016).
13. F. G. S. L. Brandão, A. Kalev, T. Li, C. Y.-Y. Lin, K. M. Svore, X. Wu, Quantum SDP solvers: Large speed-ups, optimality, and applications to quantum learning. arXiv:1710.02581 [quant-ph] (6 October 2017).
14. E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, D. Preda, A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. *Science* **292**, 472–475 (2001).
15. V. Giovannetti, S. Lloyd, L. Maccone, Quantum random access memory. *Phys. Rev. Lett.* **100**, 160501 (2008).
16. X. X. Yuan, J.-J. Ma, P.-Y. Hou, X.-Y. Chang, C. Zu, L.-M. Duan, Experimental demonstration of a quantum router. *Sci. Rep.* **5**, 12452 (2015).
17. V. Giovannetti, S. Lloyd, L. Maccone, Architectures for a quantum random access memory. *Phys. Rev. A* **78**, 052310 (2008).
18. S. Aaronson, Read the fine print. *Nat. Phys.* **11**, 291–293 (2015).
19. T. Jebara, *Machine Learning: Discriminative and Generative* (Springer Science & Business Media, 2012), vol. 755.
20. S. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach* (Prentice Hall Press, ed. 3, 2009).
21. I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning* (MIT Press, 2016).
22. C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer, 2006).
23. M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, R. Melko, Quantum Boltzmann machine. *Phys. Rev. X* **8**, 021050 (2018).
24. R. Raussendorf, H. J. Briegel, A one-way quantum computer. *Phys. Rev. Lett.* **86**, 5188–5191 (2001).
25. N. Le Roux, Y. Bengio, Representational power of restricted Boltzmann machines and deep belief networks. *Neural Comput.* **20**, 1631–1649 (2008).
26. L. J. Stockmeyer, The polynomial-time hierarchy. *Theor. Comput. Sci.* **3**, 1–22 (1976).
27. E. Y. Loh Jr., J. E. Gubernatis, R. T. Scalettar, S. R. White, D. J. Scalapino, R. L. Sugar, Sign problem in the numerical simulation of many-electron systems. *Phys. Rev. B* **41**, 9301–9307 (1990).
28. S. Shalev-Shwartz, S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms* (Cambridge Univ. Press, 2014).
29. M. Schwarz, K. Temme, F. Verstraete, Preparing projected entangled pair states on a quantum computer. *Phys. Rev. Lett.* **108**, 110502 (2012).
30. D. Perez-Garcia, F. Verstraete, M. M. Wolf, J. I. Cirac, Peps as unique ground states of local Hamiltonians. *Quantum Inf. Comput.* **8**, 650–663 (2008).
31. D. S. Abrams, S. Lloyd, Quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors. *Phys. Rev. Lett.* **83**, 5162–5165 (1999).
32. D. Nagaj, P. Wocjan, Y. Zhang, Fast amplification of QMA. *Quantum Inf. Comput.* **9**, 1053–1068 (2009).
33. M. A. Nielsen, I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge Univ. Press, 2000).
34. D. W. Berry, G. Ahokas, R. Cleve, B. C. Sanders, Efficient quantum algorithms for simulating sparse Hamiltonians. *Commun. Math. Phys.* **270**, 359–371 (2007).
35. P. Dagum, M. Luby, Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artif. Intell.* **60**, 141–153 (1993).
36. A. Yu. Kitaev, A. H. Shen, M. N. Vyalyi, *Classical and Quantum Computation* (American Mathematical Society, 2002), vol. 47.
37. R. Oliveira, B. M. Terhal, The complexity of quantum spin systems on a two-dimensional square lattice. *Quantum Inf. Comput.* **8**, 900–924 (2008).
38. X. Gao, S.-T. Wang, L.-M. Duan, Quantum supremacy for simulating a translation-invariant Ising spin model. *Phys. Rev. Lett.* **118**, 040502 (2017).
39. S. Aaronson, A. Cojocaru, A. Gheorghiu, E. Kashefi, On the implausibility of classical client blind quantum computing. arXiv:1704.08482 [quant-ph] (27 April 2017).
40. J. Kempe, A. Kitaev, O. Regev, The complexity of the local Hamiltonian problem. *SIAM J. Comput.* **35**, 1070–1097 (2006).

Acknowledgments: We thank D. Deng, J. Liu, T. Liu, M. Lukin, S. Lu, L. Wang, S. Wang, and Z. Wei for helpful discussions. **Funding:** This work was supported by the Ministry of Education and the National Key Research and Development Program of China (2016YFA0301902). L.-M.D. and Z.-Y.Z. also acknowledge support from the MURI program. **Author contributions:** X.G. and Z.-Y.Z. carried out the work under L.-M.D.'s supervision. All the authors made substantial contributions to this work. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Materials. Additional data related to this paper may be requested from the authors. Correspondence and requests for materials should be addressed to L.-M.D. (lmduan@tsinghua.edu.cn) or X.G. (gaoxungx@gmail.com).

Submitted 16 April 2018
Accepted 7 November 2018
Published 7 December 2018
10.1126/sciadv.aat9004

Citation: X. Gao, Z.-Y. Zhang, L.-M. Duan, A quantum machine learning algorithm based on generative models. *Sci. Adv.* **4**, eaat9004 (2018).

ScienceAdvances

A quantum machine learning algorithm based on generative models

X. Gao, Z.-Y. Zhang, and L.-M. Duan

Sci. Adv. **4** (12), eaat9004. DOI: 10.1126/sciadv.aat9004

Science Advances (ISSN 2375-2548) is published by the American Association for the Advancement of Science. 1200 New York Avenue NW, Washington, DC 20005. The title *Science Advances* is a registered trademark of AAAS.