Quantum Boltzmann Machine

Mohammad H. Amin,[1,2] Evgeny Andriyash,[1] Jason Rolfe,[1] Bohdan Kulchytskyy,[3,4] and Roger Melko[3,4]
[1]*D-Wave Systems Inc., 3033 Beta Avenue, Burnaby, British Columbia, Canada V5G 4M9*
[2]*Department of Physics, Simon Fraser University, Burnaby, British Columbia, Canada V5A 1S6*
[3]*Department of Physics and Astronomy, University of Waterloo, 200 University Avenue West Waterloo, Ontario, Canada N2L 3G1*
[4]*Perimeter Institute for Theoretical Physics, Waterloo, Ontario N2L 2Y5, Canada*

(Received 21 July 2017; revised manuscript received 16 January 2018; published 23 May 2018)

Inspired by the success of Boltzmann machines based on classical Boltzmann distribution, we propose a new machine-learning approach based on quantum Boltzmann distribution of a quantum Hamiltonian. Because of the noncommutative nature of quantum mechanics, the training process of the quantum Boltzmann machine (QBM) can become nontrivial. We circumvent the problem by introducing bounds on the quantum probabilities. This allows us to train the QBM efficiently by sampling. We show examples of QBM training with and without the bound, using exact diagonalization, and compare the results with classical Boltzmann training. We also discuss the possibility of using quantum annealing processors for QBM training and application.

DOI: 10.1103/PhysRevX.8.021050

Subject Areas: Condensed Matter Physics, Quantum Information

I. INTRODUCTION

Machine learning is a rapidly growing field in computer science with applications in computer vision, voice recognition, medical diagnosis, spam filtering, search engines, etc. [1]. Machine-learning algorithms operate by constructing a model with parameters that can be determined (learned) from a large amount of example inputs, called the *training set*. The trained model can then make predictions about unseen data. The ability to do so is called *generalization*. This could be, for example, detecting an object, like a cat, in an image or recognizing a command from a voice input. One approach to machine learning is probabilistic modeling in which the probability distribution of the data ($P^{\text{data}}_{\mathbf{v}}$ for a given state $\mathbf{v}$) is approximated based upon a finite set of samples. If the process of training is successful, the learned distribution $P_{\mathbf{v}}$ has enough resemblance to the actual distribution of the data, $P^{\text{data}}_{\mathbf{v}}$, such that it can make correct predictions about unseen situations. Depending upon the details of the distributions and the approximation technique, machine learning can be used to perform classification, clustering, collaborative filtering, compression, denoising, inpainting, or a variety of other algorithmic tasks [2].

The possibility of using quantum mechanics for machine learning has been considered theoretically [3–14]. With the development of quantum annealing processors [15], it has become possible to test machine-learning ideas with an actual quantum hardware [16–19]. In all of the above works, however, the quantum processor is considered as a means to provide fast solutions to an otherwise classical problem. In other words, the model stays classical and quantum mechanics is used only to facilitate the training. In this work, we propose a quantum probabilistic model for machine learning based on Boltzmann distribution of a quantum Hamiltonian, therefore, a quantum Boltzmann machine (QBM). In our approach, the quantum nature of the processor is exploited both in the model and in the training process, which to our knowledge is the first example among the energy-based models.

The Boltzmann machine (BM) is a classic machine-learning technique, and serves as the basis of powerful deep learning models such as deep belief networks and deep Boltzmann machines [20–22]. It comprises a probabilistic network of binary units with a quadratic energy function. In principle, one could consider more general energy functions to bring in more flexibility [23–25], but training can become impractical and generalization suffers as the number of parameters grows. A BM commonly consists of visible and hidden binary units, which we jointly denote by z_a, $a = 1, \ldots, N$, where N is the total number of units. To maintain consistency with the standard notation in quantum mechanics, we use $z_a \in \{-1, +1\}$, rather than $z_a \in \{0, 1\}$; the corresponding probability distributions are identical up to a linear transformation of their parameters. To distinguish the visible and hidden variables, we use the notation $z_a = (z_\nu, z_i)$, with index ν for visible variables and i for

hidden variables. We also use vector notations $\mathbf{v}$, $\mathbf{h}$, and $\mathbf{z} = (\mathbf{v}, \mathbf{h})$ to represent states of visible, hidden, and combined units, respectively. In physics language, the quadratic energy function over binary units z_a is referred to as an Ising model with the energy function:

$$E_{\mathbf{z}} = -\sum_a b_a z_a - \sum_{a,b} w_{ab} z_a z_b. \tag{1}$$

The dimensionless parameters b_a and w_{ab} are tuned during the training [26]. In equilibrium, the probability of observing a state $\mathbf{v}$ of the visible variables is given by the Boltzmann distribution summed over the hidden variables:

$$P_{\mathbf{v}} = Z^{-1} \sum_{\mathbf{h}} e^{-E_{\mathbf{z}}}, \qquad Z = \sum_{\mathbf{z}} e^{-E_{\mathbf{z}}}, \tag{2}$$

called *marginal* distribution. The goal of unsupervised learning is to determine Hamiltonian parameters, $\theta \in \{b_a, w_{ab}\}$, such that $P_{\mathbf{v}}$ becomes as close as possible to $P_{\mathbf{v}}^{\text{data}}$ defined by the training set. To achieve this, we need to maximize the average log-likelihood, or equivalently minimize the average negative log-likelihood defined by

$$\mathcal{L} = -\sum_{\mathbf{v}} P_{\mathbf{v}}^{\text{data}} \log P_{\mathbf{v}}, \tag{3}$$

which for the probability distribution Eq. (2) is

$$\mathcal{L} = -\sum_{\mathbf{v}} P_{\mathbf{v}}^{\text{data}} \log \frac{\sum_{\mathbf{h}} e^{-E_{\mathbf{z}}}}{\sum_{\mathbf{z}'} e^{-E_{\mathbf{z}'}}}. \tag{4}$$

The minimization can be done using the gradient decent technique. In each iteration, the parameter θ is changed by a small step in the direction opposite to the gradient:

$$\delta\theta = -\eta \partial_\theta \mathcal{L}, \tag{5}$$

where the learning rate η controls the step sizes. An important requirement for applicability of the gradient decent technique is the ability to calculate the gradients $\partial_\theta \mathcal{L}$ efficiently. Using Eq. (4), we have

$$\begin{aligned} \partial_\theta \mathcal{L} &= \sum_{\mathbf{v}} P_{\mathbf{v}}^{\text{data}} \frac{\sum_{\mathbf{h}} \partial_\theta E_{\mathbf{z}} e^{-E_{\mathbf{z}}}}{\sum_{\mathbf{h}} e^{-E_{\mathbf{z}}}} - \frac{\sum_{\mathbf{z}} \partial_\theta E_{\mathbf{z}} e^{-E_{\mathbf{z}}}}{\sum_{\mathbf{z}} e^{-E_{\mathbf{z}}}} \\ &= \overline{\langle \partial_\theta E_{\mathbf{z}} \rangle_{\mathbf{v}}} - \langle \partial_\theta E_{\mathbf{z}} \rangle, \end{aligned} \tag{6}$$

where $\langle \cdots \rangle$ and $\langle \cdots \rangle_{\mathbf{v}}$ are Boltzmann averages with free and fixed visible variables, respectively, and $\overline{\langle \cdots \rangle_{\mathbf{v}}} \equiv \sum_{\mathbf{v}} P_{\mathbf{v}}^{\text{data}} \langle \cdots \rangle_{\mathbf{v}}$ denotes double averaging. Fixing visible variables to the data is usually called *clamping*. Using Eq. (1) for $E_{\mathbf{z}}$, we obtain

$$\delta b_a = \eta(\overline{\langle z_a \rangle_{\mathbf{v}}} - \langle z_a \rangle), \tag{7}$$

$$\delta w_{ab} = \eta(\overline{\langle z_a z_b \rangle_{\mathbf{v}}} - \langle z_a z_b \rangle). \tag{8}$$

The gradient steps are expressed in terms of differences between the clamped (i.e., fixed $\mathbf{v}$) and unclamped averages. These two terms are sometimes called *positive* and *negative* phases. Since the averages can be estimated using sampling, the process of gradient estimation can be done efficiently provided that we have an efficient way of performing sampling. The machine-learning community has developed tools to achieve this goal. A simplified architecture of the Boltzmann machine, known as the restricted Boltzmann machine, makes it possible to compute analytically the averages in the clamped phase (more discussion on this architecture are found in Sec. II B). As for the estimation of the computationally intractable negative phase, an approximative sampling technique, called contrastive divergence [27], has been proven successful in practical applications.

II. QUANTUM BOLTZMANN MACHINE: UNSUPERVISED LEARNING

One can replace the classical spins or bits in Eq. (1) with quantum bits (qubits) (cf. Ref. [28]). The mathematics of quantum mechanics is based on matrices (operators) with dimensionality equal to the number of possible states (2^N). This is in contrast to vectors with dimensionality equal to the number of variables (N) used in common machine-learning techniques. For instance, instead of the energy function Eq. (1), one considers a $2^N \times 2^N$ diagonal matrix, called the Hamiltonian:

$$H = -\sum_a b_a \sigma_a^z - \sum_{a,b} w_{ab} \sigma_a^z \sigma_b^z. \tag{9}$$

This Hamiltonian is constructed in such a way that its diagonal elements are energy values [Eq. (1)] corresponding to all 2^N binary states $\mathbf{z}$ ordered lexicographically. To generate such a Hamiltonian, we replace z_a in Eq. (1) with a $2^N \times 2^N$ matrix,

$$\sigma_a^z \equiv \overbrace{I \otimes \cdots \otimes I}^{a-1} \otimes \sigma_z \otimes \overbrace{I \otimes \cdots \otimes I}^{N-a}, \tag{10}$$

where $\otimes$ means tensor product (sometimes called the Kronecker or outer product) and

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{11}$$

Every element in Eq. (10) is an identity matrix (I) except the ath element, which is a Pauli matrix (σ_z). Equation (1) will therefore be replaced by the diagonal Hamiltonian where b_a and w_{ab} are still scalars. Figure 1(a) shows an example of such a model with visible and hidden qubits depicted as blue and red circles, respectively. We represent eigenstates of this Hamiltonian by $|\mathbf{v}, \mathbf{h}\rangle$, where again $\mathbf{v}$ and $\mathbf{h}$ denote visible and hidden variables, respectively.

We can now define matrix exponentiation through Taylor expansion, $e^{-H} = \sum_{k=0}^{\infty} (1/k!)(-H)^k$. For a diagonal

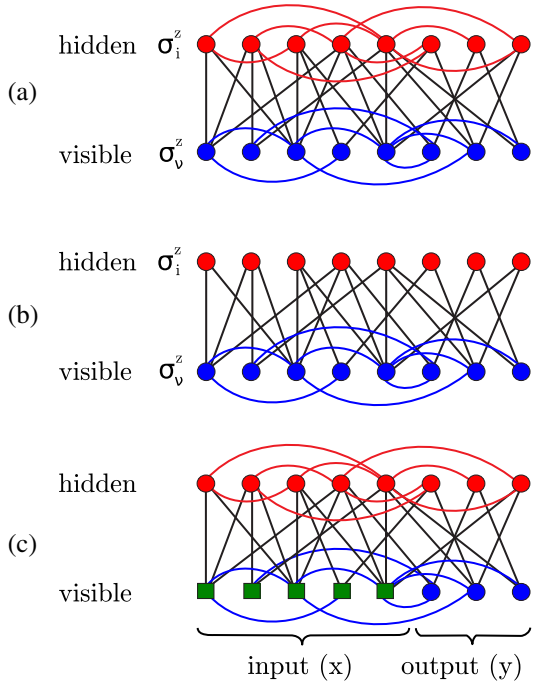

FIG. 1. (a) An example of a quantum Boltzmann machine with visible (blue) and hidden (red) qubits. (b) A semirestricted quantum Boltzmann machine with no lateral connection between the hidden variables. (c) Discriminative learning with QBM. The (green) squares represent classical input $\mathbf{x}$, which are not necessarily binary numbers. The input applies energy biases to the hidden and output qubits according to the coupling coefficients represented by solid lines.

Hamiltonian, e^{-H} is a diagonal matrix with its 2^N diagonal elements being e^{-E_z} corresponding to all the 2^N states. With the partition function given by $Z = \mathrm{Tr}[e^{-H}]$ [cf. Eq. (2)], we define the density matrix as

$$\rho = Z^{-1} e^{-H}. \tag{12}$$

The diagonal elements of ρ are therefore Boltzmann probabilities of all the 2^N states. For a given state $|\mathbf{v}\rangle$ of the visible variables, we can obtain the marginal Boltzmann probability $P_\mathbf{v}$ as

$$P_\mathbf{v} = \mathrm{Tr}[\Lambda_\mathbf{v}\rho], \tag{13}$$

where $\Lambda_\mathbf{v}$ limits the trace only to diagonal terms that correspond to the visible variables being in state $\mathbf{v}$. Thus, $\Lambda_\mathbf{v}$ is a diagonal matrix with diagonal elements being either 1, when the visible variables are in state $\mathbf{v}$, or 0 otherwise. In operator notation, we write

$$\Lambda_\mathbf{v} = |\mathbf{v}\rangle\langle\mathbf{v}| \otimes \mathcal{I}_\mathbf{h}, \tag{14}$$

where $\mathcal{I}_\mathbf{h}$ is the identity matrix acting on the hidden variables, and

$$|\mathbf{v}\rangle\langle\mathbf{v}| \equiv \prod_\nu \left(\frac{1+\mathbf{v}_\nu\sigma_\nu^z}{2}\right) \tag{15}$$

is a projection operator in the subspace of visible variables. Equations (2) and (13) are equivalent when the Hamiltonian and therefore the density matrix are diagonal, but Eq. (13) also holds for nondiagonal matrices.

We can now introduce noncommutative effects of quantum mechanics by adding off-diagonal terms to the Hamiltonian. The simplest example is a transverse field to the Ising Hamiltonian, which we consider here, but our approach is applicable to any form of off-diagonal elements. Let us introduce nondiagonal matrices,

$$\sigma_a^x \equiv \overbrace{I\otimes\cdots\otimes I}^{a-1} \otimes \sigma_x \otimes \overbrace{I\otimes\cdots\otimes I}^{N-a}, \quad \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

which represent transverse components of spin. The transverse Ising Hamiltonian is written as

$$H = -\sum_a \Gamma_a \sigma_a^x - \sum_a b_a \sigma_a^z - \sum_{a,b} w_{ab}\sigma_a^z\sigma_b^z. \tag{16}$$

Every eigenstate of H is now a superposition in the computation basis made of the classical states $|\mathbf{v},\mathbf{h}\rangle$. As the probabilistic model for the QBM, we use quantum Boltzmann distribution with the density matrix Eq. (12), which now has off-diagonal elements. In each measurement the states of the qubits are read out in the σ_z basis and the outcome will be a classical value ± 1. Because of the statistical nature of quantum mechanics, after each measurement a classical output $\mathbf{v}$ for the visible variables will appear with the probability $P_\mathbf{v}$ given by Eq. (13).

To train a QBM, we change the Hamiltonian parameters θ such that the probability distributions $P_\mathbf{v}$ becomes close to $P_\mathbf{v}^{\mathrm{data}}$ of the input data. This is achieved by minimizing the negative log-likelihood, which from Eqs. (3), (12), and (13) is

$$\mathcal{L} = -\sum_\mathbf{v} P_\mathbf{v}^{\mathrm{data}} \log \frac{\mathrm{Tr}[\Lambda_\mathbf{v} e^{-H}]}{\mathrm{Tr}[e^{-H}]}. \tag{17}$$

The gradient of $\mathcal{L}$ is given by

$$\partial_\theta \mathcal{L} = \sum_\mathbf{v} P_\mathbf{v}^{\mathrm{data}} \left(\frac{\mathrm{Tr}[\Lambda_\mathbf{v}\partial_\theta e^{-H}]}{\mathrm{Tr}[\Lambda_\mathbf{v} e^{-H}]} - \frac{\mathrm{Tr}[\partial_\theta e^{-H}]}{\mathrm{Tr}[e^{-H}]}\right). \tag{18}$$

Once again, we hope to be able to estimate the gradients efficiently using sampling. However, since H and $\partial_\theta H$ are now matrices that do not commute, we have $\partial_\theta e^{-H} \neq -e^{-H}\partial_\theta H$, and therefore we do not trivially obtain expectations of $\partial_\theta H$, as in the classical case. Writing $e^{-H} = [e^{-\delta\tau H}]^n$, where $\delta\tau \equiv 1/n$, we have

$$\partial_\theta e^{-H} = -\sum_{m=1}^{n} e^{-m\delta\tau H}\partial_\theta H \delta\tau e^{-(n-m)\delta\tau H} + \mathcal{O}(\delta\tau^2)$$
$$\xrightarrow[[n\to\infty]]{} -\int_0^1 d\tau e^{-\tau H}\partial_\theta H e^{(\tau-1)H}. \tag{19}$$

Tracing over both sides and using the permutation property of the trace, we find

$$\mathrm{Tr}[\partial_\theta e^{-H}] = -\mathrm{Tr}[e^{-H}\partial_\theta H], \tag{20}$$

which is the same as the classical relation. Plugging this into the second term of Eq. (18) gives

$$\frac{\mathrm{Tr}[\partial_\theta e^{-H}]}{\mathrm{Tr}[e^{-H}]} = -\langle \partial_\theta H\rangle, \tag{21}$$

where $\langle\cdots\rangle \equiv \mathrm{Tr}[\rho\ldots]$ denotes Boltzmann averaging. This term can be estimated by sampling from the physical distribution Eq. (12). However, the first term in Eq. (18),

$$\frac{\mathrm{Tr}[\Lambda_{\mathbf{v}}\partial_\theta e^{-H}]}{\mathrm{Tr}[\Lambda_{\mathbf{v}} e^{-H}]} = -\int_0^1 d\tau \frac{\mathrm{Tr}[\Lambda_{\mathbf{v}} e^{-\tau H}\partial_\theta H e^{(\tau-1)H}]}{\mathrm{Tr}[\Lambda_{\mathbf{v}} e^{-H}]}, \tag{22}$$

corresponds to computation of operators propagated in imaginary time. As such, it requires numerical estimation using, e.g., exact diagonalization or stochastic techniques such as the quantum Monte Carlo technique. Since a separate computation is needed for each vector in a data set; this renders the training of a QBM inefficient and basically impractical for large data sets. A workaround for this problem is to introduce a properly defined upper bound for $\mathcal{L}$ and minimize it, as we discuss below. We call this approach bound-based QBM (bQBM). Minimizing a bound on the negative log-likelihood is a common practice in machine learning.

A. Bound-based QBM

One can define a lower bound for the probabilities using the Golden-Thompson inequality [29,30]:

$$\mathrm{Tr}[e^A e^B] \geq \mathrm{Tr}[e^{A+B}], \tag{23}$$

which holds for any Hermitian matrices A and B. We can therefore write

$$\mathrm{Tr}[e^{-H} e^{\ln(\Lambda_{\mathbf{v}}+\epsilon)}] \geq \mathrm{Tr}[e^{-H+\ln(\Lambda_{\mathbf{v}}+\epsilon)}], \tag{24}$$

where ϵ is a small positive number. Taking ϵ to zero gives $\mathrm{Tr}[e^{-H+\ln(\Lambda_{\mathbf{v}}+\epsilon)}] \to \mathrm{Tr}[e^{-H_{\mathbf{v}}}]$, where

$$H_{\mathbf{v}} = \langle \mathbf{v}|H|\mathbf{v}\rangle \tag{25}$$

is the *clamped* Hamiltonian because every visible qubit σ_ν^z is clamped to its corresponding classical data value $\mathbf{v}_\nu$ due to an infinite energy penalty from $\ln(\Lambda_{\mathbf{v}}+\epsilon)$.

We therefore can write

$$P_{\mathbf{v}} = \frac{\mathrm{Tr}[\Lambda_{\mathbf{v}} e^{-H}]}{\mathrm{Tr}[e^{-H}]} \geq \frac{\mathrm{Tr}[e^{-H_{\mathbf{v}}}]}{\mathrm{Tr}[e^{-H}]}, \tag{26}$$

which leads to

$$\mathcal{L} \leq \tilde{\mathcal{L}} \equiv -\sum_{\mathbf{v}} P_{\mathbf{v}}^{\mathrm{data}} \log \frac{\mathrm{Tr}[e^{-H_{\mathbf{v}}}]}{\mathrm{Tr}[e^{-H}]}. \tag{27}$$

Instead of minimizing $\mathcal{L}$, we now minimize its upper bound $\tilde{\mathcal{L}}$ using the gradient

$$\partial_\theta \tilde{\mathcal{L}} = \sum_{\mathbf{v}} P_{\mathbf{v}}^{\mathrm{data}} \left(\frac{\mathrm{Tr}[e^{-H_{\mathbf{v}}}\partial_\theta H_{\mathbf{v}}]}{\mathrm{Tr}[e^{-H_{\mathbf{v}}}]} - \frac{\mathrm{Tr}[e^{-H}\partial_\theta H]}{\mathrm{Tr}[e^{-H}]} \right)$$
$$= (\overline{\langle \partial_\theta H_{\mathbf{v}}\rangle_{\mathbf{v}}} - \langle \partial_\theta H\rangle), \tag{28}$$

where

$$\overline{\langle\cdots\rangle_{\mathbf{v}}} = \sum_{\mathbf{v}} P_{\mathbf{v}}^{\mathrm{data}} \langle\cdots\rangle_{\mathbf{v}} = \sum_{\mathbf{v}} P_{\mathbf{v}}^{\mathrm{data}} \frac{\mathrm{Tr} e^{-H_{\mathbf{v}}}\ldots}{\mathrm{Tr} e^{-H_{\mathbf{v}}}}. \tag{29}$$

Taking θ to be b_a, w_{ab}, and using $\delta\theta = -\eta\partial_\theta\tilde{\mathcal{L}}$, we obtain

$$\delta b_a = \eta(\overline{\langle \sigma_a^z\rangle_{\mathbf{v}}} - \langle \sigma_a^z\rangle), \tag{30}$$

$$\delta w_{ab} = \eta(\overline{\langle \sigma_a^z\sigma_b^z\rangle_{\mathbf{v}}} - \langle \sigma_a^z\sigma_b^z\rangle). \tag{31}$$

Again the gradient steps are expressed in terms of differences between the unclamped and clamped averages, $\langle\cdots\rangle$ and $\langle\cdots\rangle_{\mathbf{v}}$, which can be obtained by sampling from a Boltzmann distribution with Hamiltonians H and $H_{\mathbf{v}}$, respectively. In Sec. IV, we give examples of training QBM and compare the results of minimizing $\mathcal{L}$ using Eq. (18) and minimizing its upper bound $\tilde{\mathcal{L}}$ using Eq. (28).

One may also attempt to train Γ_a using the upper bound $\tilde{\mathcal{L}}$. From Eq. (28) we obtain

$$\delta\Gamma_a = \eta(\overline{\langle \sigma_a^x\rangle_{\mathbf{v}}} - \langle \sigma_a^x\rangle). \tag{32}$$

There are a few problems with using Eq. (32) to train Γ_a. First of all, one cannot calculate $\langle\sigma_a^x\rangle$ by sampling in the σ_a^z basis. Therefore, measurement in the σ_a^x basis is needed to estimate $\langle\sigma_a^x\rangle$. Moreover, the first term in Eq. (32) is always zero for visible variables; i.e., $\overline{\langle\sigma_\nu^x\rangle_{\mathbf{v}}} = 0$, $\forall\,\nu$. Since $\langle\sigma_\nu^x\rangle > 0$ for positive Γ_ν, $\delta\Gamma_\nu$ will always be negative, which means $\Gamma_\nu \to 0$, $\forall\,\nu$. This is inconsistent with what we obtain when we train Γ_ν using the exact gradient Eq. (18). Therefore, vanishing Γ_ν is an artifact of the upper-bound minimization. In other words, we cannot learn the transverse field using the upper bound. Instead, it should be treated as a hyperparameter. One may still train the transverse field using the exact log-likelihood, but it becomes quickly intractable as the size of the QBM grows.

B. Restricted QBM

Thus far we have not imposed any restrictions on the connectivity between visible and hidden qubits or lateral connectivity among visible or hidden qubits. We note that calculation of the first term in Eqs. (30) and (31), sometimes called positive phase, requires sampling from distributions with clamped Hamiltonians [Eq. (25)]. This sampling can become computationally expensive for a large data set, because it has to be done for every input data element. If we *restrict* our QBM to have no lateral connectivity in the hidden layer (RQBM) [see Fig. 1(b)], the hidden qubits become uncoupled in the positive phase and the calculations can be carried out exactly. We can write the clamped Hamiltonian Eq. (25) as

$$H_{\mathbf{v}} = -\sum_i [\Gamma_i \sigma_i^x + b_i^{\text{eff}}(\mathbf{v})\sigma_i^z], \tag{33}$$

where $b_i^{\text{eff}}(\mathbf{v}) = b_i + \sum_\nu w_{i\nu}\mathbf{v}_\nu$. Expectations $\langle\sigma_i^z\rangle_{\mathbf{v}}$ entering Eq. (30) can be computed exactly:

$$\langle\sigma_i^z\rangle_{\mathbf{v}} = \frac{b_i^{\text{eff}}}{D_i}\tanh D_i, \tag{34}$$

where $D_i = \sqrt{\Gamma_i^2 + (b_i^{\text{eff}})^2}$. Notice that for the restricted type of a BM architecture known as RBM, Eq. (34) reduces to the classical expression,

$$\langle\sigma_i^z\rangle_{\mathbf{v}} = \tanh b_i^{\text{eff}}, \tag{35}$$

in the limit $\Gamma_i \to 0$. We emphasize that unlike the classical RBM, in which there are no lateral connections in both hidden and visible layers (for contrastive divergence techniques to work), RQBM requires only their absence in the hidden layer, usually called a semirestricted Boltzmann machine [31]. In Sec. IV, we give an example of training RQBM and illustrate the importance of using Eq. (34) instead of their classical limit Eq. (35).

III. SUPERVISED LEARNING

One important application of machine learning is classification in which a category (label) is assigned to each data point. For example, in spam detection the goal is to determine which of the two labels, "spam" or "not spam," should be assigned to a given text. The process of inferring a functional relation between input and label from a set of labeled data is called *supervised* learning. Denoting the feature vector (input) by $\mathbf{x}$ and label (output) by $\mathbf{y}$, the problem is to infer a function $g(\mathbf{x}):\mathbf{x}\to\mathbf{y}$ from the set of labeled data $(\mathbf{x}_i, \mathbf{y}_i)$. In probabilistic approaches to this problem, which are our main interest here, the output $\mathbf{y}$ that is most probable, subject to the input $\mathbf{x}$, is chosen as the label. Therefore, the function $g(\mathbf{x})$ is determined by the conditional probability $P_{\mathbf{y}|\mathbf{x}}$ of output given input:

$$g(\mathbf{x}) = \arg\max_{\mathbf{y}} P_{\mathbf{y}|\mathbf{x}}. \tag{36}$$

The end goal of training is to make $P_{\mathbf{y}|\mathbf{x}}$ as close as possible to the conditional distribution of the data, $P^{\text{data}}_{\mathbf{y}|\mathbf{x}}$. Assuming that the data come with a joint probability distribution $P^{\text{data}}_{\mathbf{x},\mathbf{y}}$, we can write: $P^{\text{data}}_{\mathbf{y}|\mathbf{x}} = P^{\text{data}}_{\mathbf{x},\mathbf{y}}/P^{\text{data}}_{\mathbf{x}}$, where $P^{\text{data}}_{\mathbf{x}} = \sum_{\mathbf{y}} P^{\text{data}}_{\mathbf{x},\mathbf{y}}$ is the marginal distribution.

Two possible approaches to supervised learning are *discriminative* and *generative* learning [32]. In the discriminative approach, for each $\mathbf{x}$ we try to learn the conditional distribution $P^{\text{data}}_{\mathbf{y}|\mathbf{x}}$. If an input $\mathbf{x}$ appears in the training set with probability $P^{\text{data}}_{\mathbf{x}}$, the loss function can be written as

$$\begin{aligned}\mathcal{L}_{\text{discr}} &= -\sum_{\mathbf{x}} P^{\text{data}}_{\mathbf{x}} \sum_{\mathbf{y}} P^{\text{data}}_{\mathbf{y}|\mathbf{x}} \log P_{\mathbf{y}|\mathbf{x}} \\ &= -\sum_{\mathbf{x},\mathbf{y}} P^{\text{data}}_{\mathbf{x},\mathbf{y}} \log P_{\mathbf{y}|\mathbf{x}}.\end{aligned} \tag{37}$$

In the generative approach, on the other hand, we learn the joint probability distribution without separating input from output. The loss function is therefore

$$\begin{aligned}\mathcal{L}_{\text{gen}} &= -\sum_{\mathbf{x},\mathbf{y}} P^{\text{data}}_{\mathbf{x},\mathbf{y}} \log P_{\mathbf{x},\mathbf{y}} \\ &= \mathcal{L}_{\text{discr}} - \sum_{\mathbf{x}} P^{\text{data}}_{\mathbf{x}} \log P_{\mathbf{x}},\end{aligned} \tag{38}$$

where we use $P_{\mathbf{x},\mathbf{y}} = P_{\mathbf{y}|\mathbf{x}}P_{\mathbf{x}}$. Notice that the first term is just $\mathcal{L}_{\text{discr}}$ while the second term measures the difference between the probability distribution of the training set inputs and the marginal distribution $P_{\mathbf{x}}$. This second term is called cross entropy [it is equal to Kullback-Leibler (KL) divergence defined in Eq. (54), up to a constant]. Now, we explore the possibility of applying QBM to both cases.

A. Generative learning

Generative learning with loss Eq. (38) can be done with the methods of Sec. II by treating input and output $(\mathbf{x}, \mathbf{y})$ jointly as the visible data $\mathbf{v} = [\mathbf{x}, \mathbf{y}]$ in a QBM. At the end of training, the QBM provides samples with a joint probability $P_{\mathbf{x},\mathbf{y}}$ that is close to $P^{\text{data}}_{\mathbf{x},\mathbf{y}}$. Therefore, the conditional probability

$$P_{\mathbf{y}|\mathbf{x}} = \frac{P_{\mathbf{x},\mathbf{y}}}{P_{\mathbf{x}}} = \frac{\text{Tr}[\Lambda_{\mathbf{x}}\Lambda_{\mathbf{y}}e^{-H}]}{\text{Tr}[\Lambda_{\mathbf{x}}e^{-H}]} \tag{39}$$

should also match $P^{\text{data}}_{\mathbf{y}|\mathbf{x}}$ as desired for supervised training. However, there is a problem when it comes to sampling from this conditional for a given $\mathbf{x}$. If the input $\mathbf{x}$ appears with a very small probability ($P_{\mathbf{x}} \ll 1$), it would require a large amount of samples from $P_{\mathbf{x},\mathbf{y}}$ and $P_{\mathbf{x}}$ to reliably calculate $P_{\mathbf{y}|\mathbf{x}}$ using Eq. (39).

In a classical BM, one can sample from the conditional distribution by clamping the input variables $\mathbf{x}$ to the data

and sampling the output $\mathbf{y}$. Using the same tricks as in Sec. II A, we can write

$$P^{\text{classical}}_{\mathbf{y}|\mathbf{x}} = \frac{\text{Tr}[\Lambda_{\mathbf{y}} e^{-H_{\mathbf{x}}}]}{\text{Tr}[e^{-H_{\mathbf{x}}}]} = P^{\text{clamped}}_{\mathbf{y}|\mathbf{x}}, \tag{40}$$

where $H_{\mathbf{x}} = \langle \mathbf{x}|H|\mathbf{x}\rangle$. This means for any $\mathbf{x}$, we can sample $P^{\text{clamped}}_{\mathbf{y}|\mathbf{x}}$ from $H_{\mathbf{x}}$ and that will give us $P_{\mathbf{y}|\mathbf{x}}$ in an efficient way regardless of how small $P_{\mathbf{x}}$ is. For quantum Hamiltonians, when $[H, \Lambda_{\mathbf{x}}] \neq 0$, we know that $\Lambda_{\mathbf{x}} e^{-H} \neq e^{-H_{\mathbf{x}}}$. Therefore, $P^{\text{clamped}}_{\mathbf{y}|\mathbf{x}}$ is not necessarily equal to $P_{\mathbf{y}|\mathbf{x}}$ and there is no easy way to draw samples from $P_{\mathbf{y}|\mathbf{x}}$.

One might still hope that the clamped distribution is not too far off from Eq. (39) and can be used as an approximation, $P^{\text{clamped}}_{\mathbf{y}|\mathbf{x}} \approx P_{\mathbf{y}|\mathbf{x}}$. As we see in an example in Sec. IV C, this is not true in general, and thus generative supervised learning is not feasible within the above framework.

B. Discriminative learning

In discriminative learning one distinguishes input from output during the training [2] and learns the conditional probability distribution using Eq. (37). This can be done by clamping the input $\mathbf{x}$ in both positive and negative phases. Since the input is always clamped, its role is just to apply biases to the other variables and therefore we do not need to assign any qubits to the input [see Fig. 1(c)]. The Hamiltonian of the system for a particular state of the input $\mathbf{x}$ is given by

$$H_{\mathbf{x}} = -\sum_a [\Gamma_a \sigma^x_a + b^{\text{eff}}_a(\mathbf{x})\sigma^z_a] - \sum_{a,b} w_{ab}\sigma^z_a\sigma^z_b, \tag{41}$$

where indices a and b range over both hidden and visible (output only) variables. Here, the input $\mathbf{x}$ provides a bias

$$b^{\text{eff}}_a(\mathbf{x}) = b_a + \sum_\mu w_{a\mu} x_\mu \tag{42}$$

to the ath qubit, where b_a and $w_{a\mu}$ are tunable parameters. Notice that x_μ does not need to be restricted to binary numbers, which can bring more flexibility to the supervised learning.

The probability of measuring an output state $\mathbf{y}$ once the input is set to state $\mathbf{x}$ is given by

$$P_{\mathbf{y}|\mathbf{x}} = \frac{\text{Tr}[\Lambda_{\mathbf{y}} e^{-H_{\mathbf{x}}}]}{\text{Tr}[e^{-H_{\mathbf{x}}}]}, \qquad \Lambda_{\mathbf{y}} = \mathcal{I}_{\mathbf{x}} \otimes |\mathbf{y}\rangle\langle\mathbf{y}| \otimes \mathcal{I}_{\mathbf{h}}, \tag{43}$$

where $H_{\mathbf{x}}$ is given by Eq. (41) and $\mathcal{I}_{\mathbf{x}}$ is an identity matrix acting on the input variables. The negative log-likelihood is given by Eq. (37). We can define a clamped Hamiltonian,

$$H_{\mathbf{x},\mathbf{y}} = \langle \mathbf{y}|H_{\mathbf{x}}|\mathbf{y}\rangle, \tag{44}$$

and show that

$$P_{\mathbf{y}|\mathbf{x}} \gtrsim \frac{\text{Tr}[e^{-H_{\mathbf{x},\mathbf{y}}}]}{\text{Tr}[e^{-H_{\mathbf{x}}}]}. \tag{45}$$

Again we introduce an upper bound $\tilde{\mathcal{L}}$ for the $\mathcal{L}$:

$$\mathcal{L}_{\text{discr}} \leq \tilde{\mathcal{L}}_{\text{discr}} = -\sum_{\mathbf{x},\mathbf{y}} P^{\text{data}}_{\mathbf{x},\mathbf{y}} \log \frac{\text{Tr}[e^{-H_{\mathbf{x},\mathbf{y}}}]}{\text{Tr}[e^{-H_{\mathbf{x}}}]}. \tag{46}$$

The derivative of $\tilde{\mathcal{L}}$ with respect to a Hamiltonian parameter θ is given by

$$\partial_\theta \tilde{\mathcal{L}} = \overline{\langle \partial_\theta H_{\mathbf{x},\mathbf{y}}\rangle_{\mathbf{x},\mathbf{y}}} - \overline{\langle \partial_\theta H_{\mathbf{x}}\rangle_{\mathbf{x}}}, \tag{47}$$

where

$$\overline{\langle A\rangle_{\mathbf{x}}} = \sum_{\mathbf{x}} P^{\text{data}}_{\mathbf{x}} \frac{\text{Tr}[e^{-H_{\mathbf{x}}} A]}{\text{Tr}[e^{-H_{\mathbf{x}}}]}, \tag{48}$$

$$\overline{\langle A\rangle_{\mathbf{x},\mathbf{y}}} = \sum_{\mathbf{x},\mathbf{y}} P^{\text{data}}_{\mathbf{x},\mathbf{y}} \frac{\text{Tr}[e^{-H_{\mathbf{x},\mathbf{y}}} A]}{\text{Tr}[e^{-H_{\mathbf{x},\mathbf{y}}}]}. \tag{49}$$

The gradient descent steps in the parameter space are given by

$$\delta b_a = \eta(\overline{\langle \sigma^z_a\rangle_{\mathbf{x},\mathbf{y}}} - \overline{\langle \sigma^z_a\rangle_{\mathbf{x}}}), \tag{50}$$

$$\delta w_{ab} = \eta(\overline{\langle \sigma^z_a\sigma^z_b\rangle_{\mathbf{x},\mathbf{y}}} - \overline{\langle \sigma^z_a\sigma^z_b\rangle_{\mathbf{x}}}), \tag{51}$$

$$\delta w_{a\mu} = \eta(\overline{\langle \sigma^z_a x_\mu\rangle_{\mathbf{x},\mathbf{y}}} - \overline{\langle \sigma^z_a x_\mu\rangle_{\mathbf{x}}}). \tag{52}$$

Notice that $\mathbf{x}$ is not only clamped in the positive phase (the first expectations), but also is clamped in the negative phase (the second expectations). The positive phase can be computed analytically if we use RQBM [Fig. 1(c) with no lateral connection among the hidden variables]. The negative phase has to be computed by sampling for each data point $\mathbf{x}$. This can make the calculation of the gradient steps prohibitively expensive for large data sets, unless a very fast sampling method is available.

IV. EXAMPLES

In this section, we describe a few toy examples illustrating the ideas described in the previous sections. In all examples studied, the training data were generated as a mixture of M factorized distributions (modes), each peaked around a random point. Every mode (k) is constructed by randomly selecting a center point $\mathbf{s}^k = [s^k_1, s^k_2, \ldots, s^k_N]$, with $s^k_i \in \{\pm 1\}$, and using the Bernoulli distribution, $p^{N-d^k_{\mathbf{v}}}(1-p)^{d^k_{\mathbf{v}}}$, where p is the probability of qubit ν being aligned with s^k_ν, and $d^k_{\mathbf{v}}$ is the Hamming distance between $\mathbf{v}$ and $\mathbf{s}^k$. The average probability distribution over M such modes gives our data distribution:

$$P_{\mathbf{v}}^{\text{data}} = \frac{1}{M}\sum_{k=1}^{M} p^{N-d_{\mathbf{v}}^k}(1-p)^{d_{\mathbf{v}}^k}. \tag{53}$$

In all our examples, we choose $p = 0.9$. The modes' center points were randomly and independently generated for each training set from a uniform distribution. Each data set contained 1000 training examples.

To have a measure of the quality of learning, we subtract from $\mathcal{L}$ its minimum $\mathcal{L}_{\text{min}} = -\sum_{\mathbf{v}} P_{\mathbf{v}}^{\text{data}} \log P_{\mathbf{v}}^{\text{data}}$, which happens when $P_{\mathbf{v}} = P_{\mathbf{v}}^{\text{data}}$. The difference, commonly called Kullback-Leibler divergence,

$$\text{KL} = \mathcal{L} - \mathcal{L}_{\text{min}} = \sum_{\mathbf{v}} P_{\mathbf{v}}^{\text{data}} \log \frac{P_{\mathbf{v}}^{\text{data}}}{P_{\mathbf{v}}}, \tag{54}$$

is a non-negative number measuring the difference between the two distributions; $\text{KL} = 0$ if and only if the two distributions are identical.

A. Fully visible model

We start with a fully visible example to compare BM with QBM and evaluate the quality of the bound Eq. (27) by training bQBM. We consider a fully connected model with $N = 10$ qubits. In this case, the negative log-likelihood $\mathcal{L}$ is a concave function of its parameters and, therefore, has a unique global minimum. Classical BM has $N(N+1)/2$ trainable parameters (b_a, w_{ab}). The Hamiltonian of QBM has the form Eq. (16) where we restrict all Γ_a to be the same ($= \Gamma$). In order to understand the efficiency of QBM in representing the data, we train the exact log-likelihood using Eq. (18) and treat b_a, w_{ab}, and Γ as trainable parameters. This can be done using exact diagonalization for small systems. We also perform training of the bound $\tilde{\mathcal{L}}$ using Eq. (28) treating (b_a, w_{ab}) as trainable parameters but fixing Γ to some *ad hoc* nonzero value $\Gamma = 2$. Comparing the training results of QBM with bQBM will give us some idea of the efficiency of training the bound $\tilde{\mathcal{L}}$.

Since all expectations entering the gradients of log-likelihood are computed exactly, we use the second-order optimization routine BFGS [33]. The results of training BM, QBM, and bQBM are given in Fig 2(a). The x axis in the figure corresponds to iterations of BFGS. QBM is able to learn the data noticeably better than BM, and bQBM approaches the value close to the one for QBM.

In order to visualize the training process, we keep track of the average values of classical and quantum parts of the Hamiltonian during the training:

$$E_{\text{cl}} = -\left\langle \sum_a b_a \sigma_a^z + \sum_{a,b} w_{ab}\sigma_a^z\sigma_b^z \right\rangle,$$
$$E_{\text{q}} = -\left\langle \sum_a \Gamma_a \sigma_a^x \right\rangle. \tag{55}$$

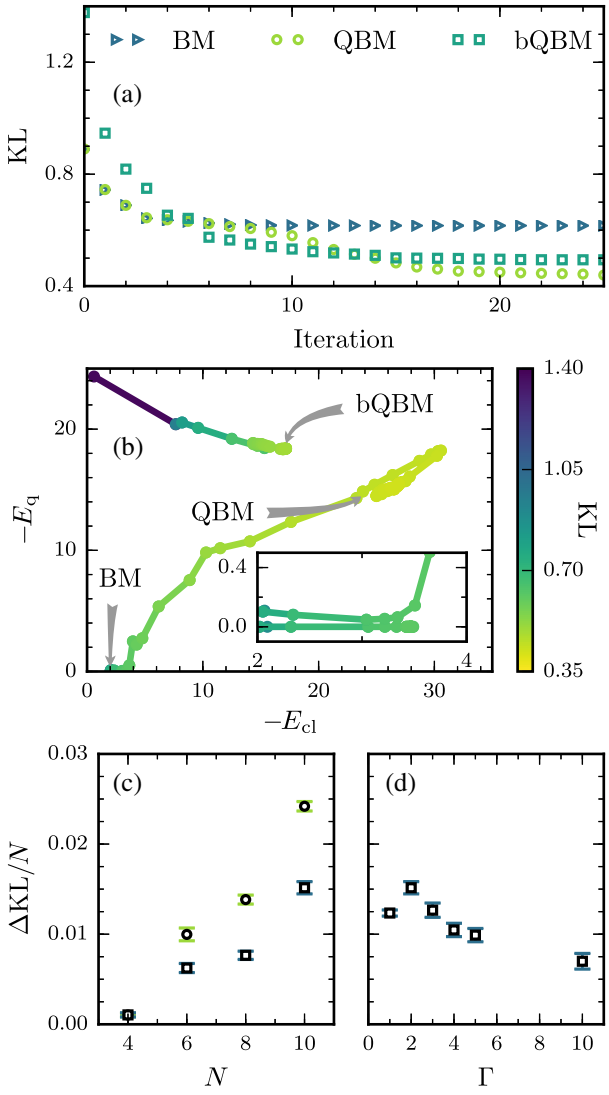

FIG. 2. Training of a fully visible fully connected model with $N = 10$ qubits on artificial data from Bernoulli mixture model Eq. (53) with the noise parameter $p = 0.9$ and the number of modes $M = 8$. Training is done using the second-order optimization routine BFGS. (a) KL divergence Eq. (54) of BM, QBM, bQBM models during training process. Both QBM and bQBM learn to KL values that are lower than that for BM. (b) Classical and quantum average energies Eq. (55) during training process. The inset details the same data set on a finer scale close to the classical regime. The bottom row of plots shows the advantage in the learning ability per spin of a quantum BM relative to the classical BM, $(\text{KL} - \text{KL}_{\text{BM}})/N$ with circles (squares) corresponding to a QBM (bQBM). This advantage is considered (c) as function of the system size with $\Gamma = 2$ and (d) as function of the transverse-field strength with $N = 10$. For those plots, the number of modes in the Bernoulli mixture data model was kept equal to the number of qubits $N = M$ while the noise parameter was always set to $p = 0.9$. In order to collect the statistics, the training was done for 100 different data sets. The error bars correspond to 1 standard deviation in the averages.

Figure 2(b) shows the learning trajectories in the space $(|E_{\rm cl}|, |E_{\rm q}|)$. BM learns a model with average energy ≈ 3.5, and KL ≈ 0.62. One can see that QBM, which starts off with $\Gamma = 0.1$, initially lowers Γ and learns (b_a, w_{ab}) that are close to the best classical result (see the inset). Soon after, QBM increases Γ and (b_a, w_{ab}) until it converges to a point with $\Gamma = 2.5$ and KL ≈ 0.42, which is better than the classical BM value. Having a fixed transverse field, $\Gamma = 2$, bQBM starts with a large $E_{\rm q}$ and approaches the parameter learned by QBM (although it does not reach the best value at $\Gamma = 2.5$ learned by QBM).

Since we cannot train the transverse field Γ, we have to treat it as a hyperparameter. We perform a scan of values of Γ and evaluate the KL advantage of QBM and bQBM over BM based on 100 randomly generated data sets. Figure 2(d) demonstrates weak dependence of this advantage on the transverse field, which justifies treating it as a hyperparameter.

We also study the dependence of the KL advantage as a function of the system size. Figure 2(c) shows that both QBM and bQBM not only preserve their advantage over the classical BM but are able to increase the gap in their learning ability.

B. Semirestricted QBM

We now consider a semirestricted BM discussed in Sec. II B. Our toy model has 8 visible units and 2 hidden units. We allow full connectivity within the visible layer and all-to-all connectivity between the layers. The data are again generated using Eq. (53) for the visible variables, with $p = 0.9$ and $M = 8$. We present the results of training in Fig 3. Similarly to the fully visible model, QBM outperforms BM, and bQBM represents a good proxy for learning quantum distribution.

In order to illustrate the significance of consistent usage of quantum distribution in evaluating the gradients Eqs. (30) and (31), we train bQBM using the classical expression instead of Eq. (34) for expectations of hidden units in the positive phase. The resulting machine learns worse than purely classical BM because the two terms in gradient expressions are evaluated inconsistently.

C. Generative supervised learning

We consider a supervised learning example with 8 inputs and 3 outputs with full connectivity between all units. For the training set we again used the multimodal distribution Eq. (53) over $\mathbf{x}$, with $M = 8$ and $p = 0.9$, and set the label $\mathbf{y}$ for each mode to be a 3-bit binary number from 0 to 7 [34]. Both BM and QBM are trained to learn the loss function Eq. (38). As such there is no difference in the training procedure as compared to the unsupervised case and similar results for the joint distribution learning are expected to apply. Our goal is to check whether $P^{\rm clamped}_{\mathbf{y}|\mathbf{x}} \approx P_{\mathbf{y}|\mathbf{x}}$, when training QBM in this generative setup. In Fig. 4, we plot the KL divergence based on the discriminative log-likelihoods Eq. (37), evaluated with conditional probabilities $P_{\mathbf{y}|\mathbf{x}}$ and clamped probabilities $P^{\rm clamped}_{\mathbf{y}|\mathbf{x}}$. One can see that although QBM is trained with the joint probability distribution $P_{\mathbf{x},\mathbf{y}}$, the conditional distribution $P_{\mathbf{y}|\mathbf{x}}$ is also learned better than BM. The clamped probability distribution $P^{\rm clamped}_{\mathbf{y}|\mathbf{x}}$, on the other hand, starts very close to the conditional distribution at the beginning of the iterations, when QBM and BM are close to each other (similarly to what was observed in Figs. 2 and 3). But as the transverse field in QBM starts to grow, the clamped distribution deviates from the conditional one and its KL divergence grows to a value much worse than the classical BM value. This confirms that even for such a small example, the clamped distribution can be very different from the true conditional distribution.

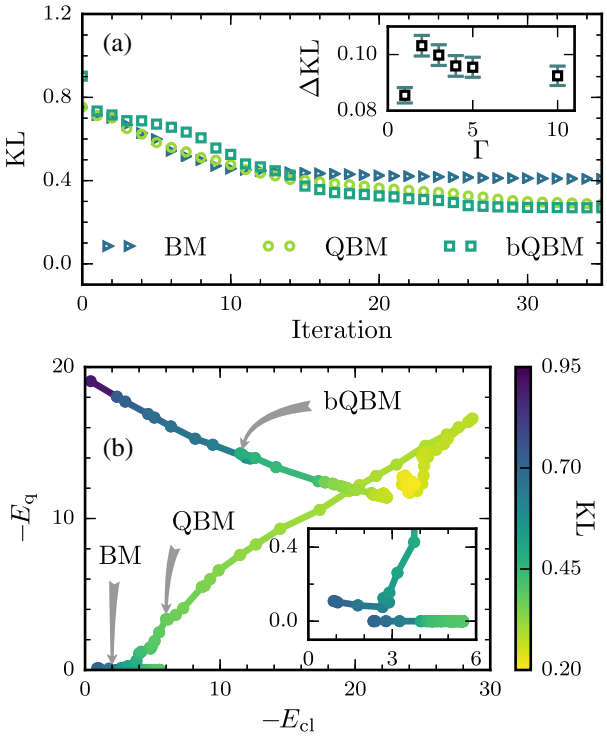

FIG. 3. Training of a RBM with 8 visible and 2 hidden units on artificial data from Bernoulli mixture model Eq. (53) using second-order optimization routine. (a) KL divergence Eq. (54) of different models during training process. Again, QBM and bQBM outperform BM. The inset shows the advantage in the learning ability of the bQBM relative to the classical BM, $\Delta{\rm KL} = {\rm KL}_{\rm bQBM} - {\rm KL}_{\rm BM}$, as function of the transverse-field strength. To collect the statistics, the training was done for 100 different data sets generated with the noise parameter $p = 0.9$ and number of modes $M = 8$. The error bars correspond to 1 standard deviation in the mean. (b) Classical and quantum average energies Eq. (55) during training process. The inset details the same data set on a finer scale close to the classical regime.

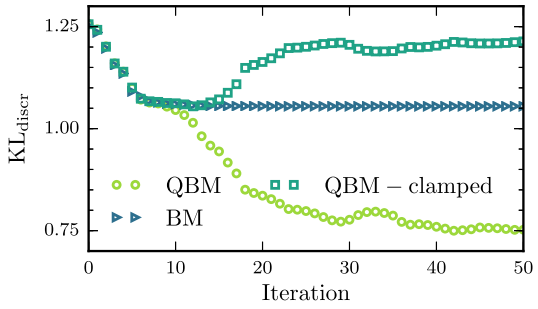

FIG. 4. Generative supervised learning using fully visible fully connected model with $N = 11$ qubits divided into 8 inputs and 3 outputs based on (38). Training is done by minimizing (38) on artificial data from Bernoulli mixture model (53) for inputs and 3-bit binary labels (0 to 7) for outputs with the noise-parameter $p = 0.9$ and the number of modes $M = 8$. We plot KL-divergence of the conditional distribution (37) and the clamped distribution (40) during the training process. The clamped QBM distribution is very different from the conditional one and gives much higher KL-divergence.

V. QBM WITH A QUANTUM ANNEALING PROCESSOR

Recent developments in manufacturing quantum annealing processors have made it possible to experimentally test some of the quantum machine-learning ideas. Up to now many experiments have confirmed the existence of quantum phenomena in such processors [15,35–38], which includes entanglement [39]. A quantum annealing processor implements the time-dependent Hamiltonian

$$\mathcal{H}(s) = -A(s)\sum_a \sigma_a^x + B(s)\left[\sum_a h_i\sigma_a^z + \sum_{a,b} J_{ab}\sigma_a^z\sigma_b^z\right], \quad (56)$$

where $s = t/t_a$, t is time, t_a is the annealing time, h_i and J_{ij} are tunable dimensionless parameters, and $A(s)$ and $B(s)$ are monotonic functions, with units of energy, such that $A(0) \gg B(0) \approx 0$ and $B(1) \gg A(1) \approx 0$. As discussed in Ref. [40], an open system quantum annealer having quasistatic evolution follows equilibrium distribution, $\rho = Z^{-1}e^{-\beta\mathcal{H}(s)}$, up to a point where the dynamics become too slow to establish equilibrium. Here, $\beta = (k_B T)^{-1}$, with T being the temperature and k_B being the Boltzmann constant. The system will then deviate from the equilibrium distribution and soon after the dynamics will freeze [see Fig. 2(c) in Ref. [40] and the related discussion].

In general, a quantum annealer with linear annealing schedule $s = t/t_a$ does not return a Boltzmann distribution. However, as argued in Ref. [40], if the dynamical slow-down and freeze-out happen within a short period of time during the annealing, then the final distribution will be close to the quantum Boltzmann distribution of Eq. (56) at a single point s^*, called the freeze-out time. In such a case, the quantum annealer with linear annealing schedule will provide approximate samples from the Boltzmann distribution corresponding to the Hamiltonian $\mathcal{H}(s^*)$. Moreover, if $A(s^*)$ happens to be small enough such that the quantum eigenstates at s^* are close to the classical eigenstates, then the resulting Boltzmann distribution will be close to the classical Boltzmann distribution. In such a case, the quantum annealer can be used as an approximate classical Boltzmann sampler for training a BM, as was done in Refs. [16,17]. Unfortunately, not all problems have a narrow freeze-out region and $A(s^*)$ is not always small. If the freeze-out does not happen in a narrow region, then the final probability distribution will depend on the history within this region and will not correspond to a Boltzmann distribution at any particular point. This would limit the applicability of using a quantum annealer for Boltzmann sampling.

In principle, it is possible to controllably freeze the evolution at a desired point, s^*, in the middle of the annealing and read out the qubits. One way to do this is via a nonuniform $s(t)$, which anneals slowly at the beginning up to s^* and then moves very fast (faster than all dynamics) to the end of annealing, i.e., quenching. An experimental demonstration of such controlled sampling was done in Ref. [41] for a specially designed 16-qubit problem. If s^* lies in the region where the evolution is still quasistatic, the quantum annealer will provide samples from the Boltzmann distribution of the Hamiltonian Eq. (16), with

$$\Gamma_a = \Gamma = \beta A(s^*), \quad (57)$$

$$b_a = \beta B(s^*) h_a, \quad (58)$$

$$w_{ab} = \beta B(s^*) J_{ab}. \quad (59)$$

Since h_a and J_{ab} are tunable parameters, by controlling the freeze-out point s*, all the dimensionless parameters in Eq. (16), i.e., Γ, b_a, w_{ab}, can be tuned and therefore the quantum annealer can be used for training a QBM.

The applicability of the controlled sampling technique used in Ref. [41] is limited by how fast the quench can be done, which is ultimately determined by the bandwidth of the filters that bring electrical signals to the chip. Because of this, such a technique is applicable only to specially designed problems that have very slow dynamics. With some modifications to the current hardware design, however, such techniques can become possible for general problems relevant to QBM in the near future.

VI. CONCLUSION

We examine the possibility of training a quantum Boltzmann machine, in which the Hamiltonian has nondiagonal elements. Motivated by the success of stochastic gradient descent in training classical Boltzmann machines, one may wish to use a similar technique to optimize the log-likelihood of the QBM. However, unlike the classical BM, for which the gradients of the log-likelihood can be

estimated using sampling, the existence of the off-diagonal elements makes the gradient estimation nontrivial. We introduce a lower bound on the log-likelihood, for which the gradient can be estimated using sampling. We show examples of training QBMs with a transverse field Ising Hamiltonian through maximizing both the log-likelihood and its lower bound, using exact diagonalization, and compare the results with classical BM training. Our small-size examples demonstrate that QBM can learn the data distribution better than classical BM. However, we expect that the main advantage brought by QBM will be in the access to fast sampling using a physical quantum hardware. For stoquastic Hamiltonians [42], where all the off-diagonal elements are nonpositive, quantum Monte Carlo algorithms can potentially be used to generate samples, but it becomes prohibitively slow and impractical beyond ∼100 qubits. Our scheme can also work with nonstoquastic Hamiltonians, for which a quantum Monte Carlo algorithms is not applicable because of the sign problem.

The bound-based QBM does not allow learning the transverse-field parameter. In our experiments we found that the dependence of the learned log-likelihood on the transverse field is weak; therefore, in practice it should be treated as a hyperparameter.

The similarity between BM and QBM training may not hold in all situations. For example, as we show in Sec. III A, sampling from a conditional distribution cannot be performed by clamping in QBM, as it is done in classical BM. The two models may also differ in other aspects; therefore, careful examination is needed before replacing BM with QBM in existing machine-learning techniques.

Finally, we discuss the possibility of using a quantum annealer for QBM training. Although the current commercial quantum annealers like D-Wave are not designed to provide quantum Boltzmann samples, with minor modifications to the hardware design, such a feature can become available. This would open new possibilities in both quantum information processing and machine-learning research areas.

Recently, Kieferova and Wiebe [43] proposed the relative entropy between the density matrix of the data and that of the quantum system as an objective function for training a QBM. Whereas the present paper focuses only on learning classical probability distributions, Ref. [43] provides a strategy to also learn a quantum distribution, including the ability to train the transverse field. In the case of classical data sets, when the model is fully visible, the objective function is exactly the same as the bound introduced in this paper [44]. For classical data sets training a model with hidden variables, the objective function of Ref. [43] involves a computationally difficult trace, in contrast to our bound [45]. Thus, the two approaches offer complementary strategies for the unsupervised learning of classical and quantum data distributions using quantum Boltzmann machines.

ACKNOWLEDGMENTS

We are grateful to Ali Ghodsi, Firas Hamze, William Macready, Anatoly Smirnov, and Giacomo Torlai for fruitful discussions. This research was partially supported by a Natural Sciences and Engineering Research Council of Canada (NSERC) Engage grant.

[1] M. I. Jordan and T. M. Mitchell, *Machine Learning: Trends, Perspectives, and Prospects*, Science **349**, 255 (2015).
[2] C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer, New York, 2006).
[3] S. Lloyd, M. Mohseni, and P. Rebentrost, *Quantum Algorithms for Supervised and Unsupervised Machine Learning*, arXiv:1307.0411.
[4] P. Rebentrost, M. Mohseni, and S. Lloyd, *Quantum Support Vector Machine for Big Data Classification*, Phys. Rev. Lett. **113**, 130503 (2014).
[5] N. Wiebe, A. Kapoor, and K. M. Svore, *Quantum Deep Learning*, Quantum Inf. Comput. **16**, 0541 (2016).
[6] H. Neven, G. Rose, and W. G. Macready, *Image Recognition with an Adiabatic Quantum Computer I. Mapping to Quadratic Unconstrained Binary Optimization*, arXiv:0804.4457.
[7] H. Neven, V. S. Denchev, G. Rose, and W. G. Macready, *Training a Binary Classifier with the Quantum Adiabatic Algorithm*, arXiv:0811.0416.
[8] H. Neven, V. S. Denchev, G. Rose, and W. G. Macready, *Training a Large Scale Classifier with the Quantum Adiabatic Algorithm*, arXiv:0912.0779.
[9] K. L. Pudenz and D. A. Lidar, *Quantum Adiabatic Machine Learning*, Quantum Inf. Process. **12**, 2027 (2013).
[10] M. Denil and N. de Freitas, in *Proceedinngs of the NIPS*2011 Workshop on Deep Learning and Unsupervised Feature Learning*.
[11] V. S. Denchev, N. Ding, S. V. N. Vishwanathan, and H. Neven, *Robust Classification with Adiabatic Quantum Optimization*, arXiv:1205.1148.
[12] V. Dumoulin, I. J. Goodfellow, A. Courville, and Y. Bengio, *On the Challenges of Physical Implementations of RBMs*, in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence* (AAAI Press, Quebec, 2014), pp. 1199–1205.
[13] R. Babbush, V. Denchev, N. Ding, S. Isakov, and H. Neven, *Construction of Non-Convex Polynomial Loss Functions for Training a Binary Classifier with Quantum Annealing*, arXiv:1406.4203.
[14] M. Gu, K. Wiesner, E. Rieper, and V. Vedral, *Quantum Mechanics Can Reduce the Complexity of Classical Models*, Nat. Commun. **3**, 762 (2012).
[15] M. W. Johnson, *Quantum Annealing with Manufactured Spins*, Nature (London) **473**, 194 (2011).
[16] S. H. Adachi and M. P. Henderson, *Application of Quantum Annealing to Training of Deep Neural Networks*, arXiv:1510.06356.
[17] M. Benedetti, J. Realpe-Gomez, R. Biswas, and A. Perdomo-Ortiz, *Estimation of Effective Temperatures in a Quantum Annealer and Its Impact in Sampling*

Applications: A Case Study towards Deep Learning Applications, Phys. Rev. A **94**, 022308 (2016).

[18] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz, *Quantum-Assisted Learning of Hardware-Embedded Probabilistic Graphical Models.*, Phys. Rev. X **7**, 041052 (2017).

[19] D. Korenkevych, Y. Xue, Z. Bian, F. Chudak, W. G. Macready, J. Rolfe, and E. Andriyash, *Benchmarking Quantum Hardware for Training of Fully Visible Boltzmann Machines*, arXiv:1611.04528.

[20] G. E. Hinton and T. J. Sejnowski, *Optimal Perceptual Inference*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, Washington, 1983).

[21] G. E. Hinton, S. Osindero, and Y-W. Teh, *A Fast Learning Algorithm for Deep Belief Nets*, Neural Comput. **18**, 1527 (2006).

[22] R. Salakhutdinov and G. Hinton, *Deep Boltzmann Machines*, in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics* (PMLR, Florida, 2009), Vol. 5, pp. 448–455.

[23] T. J. Sejnowski, *Higher-Order Boltzmann Machines*, AIP Conf. Proc. **151**, 398 (1986).

[24] M. Ranzato and G. E. Hinton, *Modeling Pixel Means and Covariances Using Factorized Third-Order Boltzmann Machines*, in *Computer Society Conference on Computer Vision and Pattern Recognition* (IEEE, San Francisco, 2010), pp. 2551–2558.

[25] R. Memisevic and G. E. Hinton, *Learning to Represent Spatial Transformation with Factored Higher-Order Boltzmann Machines*, Neural Comput. **22**, 1473 (2010).

[26] In physical systems, Hamiltonian parameters have unit of energy. We normalize these parameters by $k_B T \equiv \beta^{-1}$, where T is temperature and k_B is the Boltzmann constant; we absorb β into the parameters.

[27] G. E. Hinton, *Training Products of Experts by Minimizing Contrastive Divergence*, Neural Comput. **14**, 1771 (2002).

[28] N. Yapage and H. Nagaoka, *An Information Geometrical Approach to the Mean-Field Approximation for Quantum Ising Spin Models*, J. Phys. A **41**, 065005 (2008).

[29] S. Golden, *Lower Bounds for the Helmholtz Function*, Phys. Rev. **137**, B1127 (1965).

[30] C. J. Thompson, *Inequality with Applications in Statistical Mechanics*, J. Math. Phys. (N.Y.) **6**, 1812 (1965).

[31] S. Osindero and G. E. Hinton, *Modeling Image Patches with a Directed Hierarchy of Markov Random Fields*, edited by J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Advances in Neural Information Processing Systems Vol. 20 (Curran Associates Inc., Vancouver, 2008), pp. 1121–1128.

[32] There are other techniques used for supervised learning, for example, when only a small fraction of the available data is labeled.

[33] J. Nocedal and S. Wright, *Numerical Optimization* (Springer-Verlag, New York, 2006), p. 136.

[34] This choice was made to keep the number of qubits small to allow for exact diagonalization.

[35] R. Harris *et al.*, *Experimental Investigation of an Eight Qubit Unit Cell in a Superconducting Optimization Processor*, Phys. Rev. B **82**, 024511 (2010).

[36] S. Boixo, T. Albash, F. M. Spedalieri, N. Chancellor, and D. A. Lidar, *Experimental Signature of Programmable Quantum Annealing*, Nat. Commun. **4**, 2067 (2013).

[37] S. Boixo, T. F. Rønnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, and M. Troyer, *Evidence for Quantum Annealing with More than One Hundred Qubits*, Nat. Phys. **10**, 218 (2014).

[38] S. Boixo, V. N. Smelyanskiy, A. Shabani, S. V. Isakov, M. Dykman, V. S. Denchev, M. Amin, A. Smirnov, M. Mohseni, and H. Neven, *Computational Role of Multiqubit Tunneling in a Quantum Annealer*, Nat. Commun. **7**, 10327 (2016); see also arXiv:1411.4036.

[39] T. Lanting *et al.*, *Entanglement in a Quantum Annealing Processor*, Phys. Rev. X **4**, 021041 (2014).

[40] M. H. Amin, *Searching for Quantum Speedup in Quasistatic Quantum Annealers*, Phys. Rev. A **92**, 052323 (2015).

[41] N. G. Dickson, *Thermally Assisted Quantum Annealing of a 16-Qubit Problem*, Nat. Commun. **4**, 1903 (2013).

[42] S. Bravyi, D. P. Divincenzo, R. Oliveira, and B. M. Terhal, *The Complexity of Stoquastic Local Hamiltonian Problems*, Quantum Inf. Comput. **8**, 361 (2008).

[43] M. Kieferova and N. Wiebe, *Tomography and Generative Data Modeling via Quantum Boltzmann Training*, Phys. Rev. A **96**, 062327 (2017).

[44] Up to a constant the relative entropy between the density matrix of the data and the quantum model is given by $-\mathrm{Tr}\{\rho_{\mathrm{data}} \log (e^{-H}/\mathrm{Tr}[e^{-H}])\} = \mathrm{Tr}[\rho_{\mathrm{data}} H] + \log \mathrm{Tr}[e^{-H}] = \sum_{\mathbf{v}\in\mathrm{data}} H_{\mathbf{v}} + \log \mathrm{Tr}[e^{-H}] = -\sum_{\mathbf{v}\in\mathrm{data}} \log (e^{-H_{\mathbf{v}}}/\mathrm{Tr}[e^{-H}]) = -\tilde{\mathcal{L}}$. As a result, minimizing relative entropy is equivalent to maximizing the lower bound $\tilde{\mathcal{L}}$.

[45] Interestingly, the interpretation of our bound as a relative entropy provides an alternative intuition for why the transverse field cannot be trained with this approach: since the density matrix of the classical data is diagonal, matching the density matrix of the quantum system to it will always force the transverse field to be zero.