Quantum-Assisted Learning of Hardware-Embedded Probabilistic Graphical Models

Marcello Benedetti,[1,2,7] John Realpe-Gómez,[1,3,4] Rupak Biswas,[1,5] and Alejandro Perdomo-Ortiz[1,2,6,7,*]

[1]*Quantum Artificial Intelligence Laboratory, NASA Ames Research Center,*
Moffett Field, California 94035, USA
[2]*Department of Computer Science, University College London, WC1E 6BT London, United Kingdom*
[3]*SGT Inc., Greenbelt, Maryland 20770, USA*
[4]*Instituto de Matemáticas Aplicadas, Universidad de Cartagena, Bolívar 130001, Colombia*
[5]*Exploration Technology Directorate, NASA Ames Research Center, Moffett Field, California 94035, USA*
[6]*USRA Research Institute for Advanced Computer Science,*
Mountain View, California 94043, USA
[7]*Cambridge Quantum Computing Limited, CB2 1UB, Cambridge, United Kingdom*

Mainstream machine-learning techniques such as deep learning and probabilistic programming rely heavily on sampling from generally intractable probability distributions. There is increasing interest in the potential advantages of using quantum computing technologies as sampling engines to speed up these tasks or to make them more effective. However, some pressing challenges in state-of-the-art quantum annealers have to be overcome before we can assess their actual performance. The sparse connectivity, resulting from the local interaction between quantum bits in physical hardware implementations, is considered the most severe limitation to the quality of constructing powerful generative unsupervised machine-learning models. Here, we use embedding techniques to add redundancy to data sets, allowing us to increase the modeling capacity of quantum annealers. We illustrate our findings by training hardware-embedded graphical models on a binarized data set of handwritten digits and two synthetic data sets in experiments with up to 940 quantum bits. Our model can be trained in quantum hardware without full knowledge of the effective parameters specifying the corresponding quantum Gibbs-like distribution; therefore, this approach avoids the need to infer the effective temperature at each iteration, speeding up learning; it also mitigates the effect of noise in the control parameters, making it robust to deviations from the reference Gibbs distribution. Our approach demonstrates the feasibility of using quantum annealers for implementing generative models, and it provides a suitable framework for benchmarking these quantum technologies on machine-learning-related tasks.

Subject Areas: Interdisciplinary Physics,
Quantum Information

I. INTRODUCTION

Sampling from high-dimensional probability distributions is at the core of a wide spectrum of machine-learning techniques with important applications across science, engineering, and society; deep learning [1] and probabilistic programming [2] are some notable examples. While much of the record-breaking performance of machine-learning algorithms regularly reported in the literature pertains to task-specific supervised learning algorithms [1,3], the development of the more humanlike unsupervised learning algorithms has been lagging behind. An approach to unsupervised learning is to model the joint probability distribution of all the variables of interest. This is known as the generative approach because it allows us to generate synthetic data by sampling from the joint distribution. Generative models find application in anomaly detection, reinforcement learning, handling of missing values, and visual arts, to name a few [4]. Even in some supervised contexts, it may be useful to treat the targets as standard input and attempt to model the joint distribution [5]. Generative models rely on a sampling engine that is used for both inference and learning. Because of the intractability of traditional sampling techniques like the Markov chain Monte Carlo (MCMC) method, finding good generative models is among the hardest problems in machine learning [1,3,6–8].

Recently, there has been increasing interest in the potential that quantum computing technologies have for speeding up machine learning [9–38] or implementing more effective models [39]. This goes beyond the original focus of the quantum annealing computational paradigm [40–42], which was to solve discrete optimization problems [43–51]. Empirical results suggest that, under certain conditions, quantum annealing hardware samples from a Gibbs or a Boltzmann distribution [21,25,52–54]. In

[*]alejandro.perdomoortiz@nasa.gov
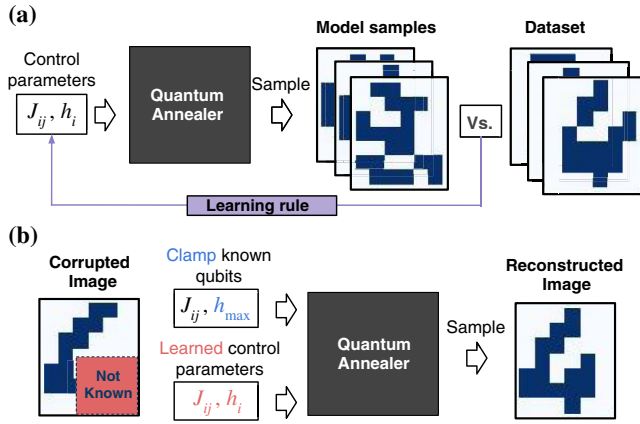
(a)

(b)

FIG. 1.   *Quantum-assisted unsupervised learning.* (a) During the training phase, samples generated by the quantum annealer are compared with samples from the data set of, say, black-and-white images. The control parameters are then modified according to a learning rule (see Sec. III). This process is iterated a given number of times, also known as epochs. (b) After being trained, we can use the quantum annealer, for instance, to reconstruct missing information in a data point, e.g., unknown values of some pixels (red region). To do this, we program the quantum annealer with the control parameters learned, except for the fields of those qubits that represent known pixels. These fields are instead set to large values h_{max}, so the qubits are clamped to the known values of the corresponding pixels. We then generate samples to infer the values of the unknown pixels.

principle, the user can adjust the control parameters so that the device implements the desired distribution. Figure 1 shows an example of how, ideally, one could use a quantum annealer for the unsupervised task of learning handwritten digits. In practice, however, there exist device-dependent limitations that complicate this process. The most pressing ones are as follows [10,11,21,52,55]: (i) The effective temperature is parameter dependent and unknown, (ii) the interaction graph is sparse, (iii) the parameters are noisy, and (iv) the dynamic range of the parameters is finite. Suitable strategies to tackle all of these issues need to be developed before we can assess whether or not quantum annealers can indeed sample more efficiently than traditional techniques on conventional computers, or whether they can implement more effective models. A relatively simple technique for the estimation of parameter-dependent effective temperature was developed in Ref. [21] and shown to perform well for training restricted Boltzmann machines. More recently, generalizations and alternative techniques have been introduced in Ref. [52]. In the context of machine learning, these techniques need to estimate temperature at each iteration, implying a computational overhead.

Here, we put forward an approach that completely sidesteps limitation (i), i.e., the need to estimate temperature at each iteration of the learning process. Furthermore, we propose a graphical model embedded in hardware that *effectively* implements all pairwise interactions between

logical variables representing the data set and that learns the parameter setting from data, improving on limitation (ii). Since the essential components for estimating the gradient needed in the learning process take place on quantum hardware, our approach is more robust to the noise in the parameters, also improving on limitation (iii).

Our work here is based on a quantum maximum-entropy model: a quantum Boltzmann machine with no hidden variables, whose learning in the classical limit is also known as the inverse Ising problem [56,57]. We show that the resulting models embedded in quantum hardware can model well both a coarse-grained binarized version of the optical recognition of handwritten digits (OptDigits) [58] and the synthetic bars-and-stripes (BAS) data set [59]. Moreover, using data sets of configurations extracted from random instances of the Sherrington-Kirkpatrick model [60–62], we show that our model's generative performance improves with training and converges to the ideal value. These results provide strong evidence that quantum annealers can indeed be effectively used as samplers and that their domain of application extends well beyond what was originally intended.

We emphasize that the objective of this work is not to address the question of quantum speedup in sampling applications but rather to provide the first clear experimental evidence that quantum annealers can be trained robustly and used in generative models for unsupervised machine-learning tasks. We use available techniques to transform the data set of interest into another data set with higher and redundant resolution, which is subsequently used to train models natively embedded in quantum hardware. We then use a gray-box model approach, which does not require us to estimate the effective temperature nor the effective transverse field; this approach has the potential to correct for errors due to nonequilibrium deviations [53], noise in the programmable parameters [63], and sampling biases in available state-of-the-art devices [64]. Hence, while the derivation of our quantum-assisted algorithm relies on the assumption that the quantum annealer is sampling from a Gibbs distribution, we do not expect that this assumption must be strictly valid for our algorithm to work well. Because we are optimizing a hard-to-evaluate convex function, the generative performance depends mostly on the quality and efficiency of the sampling required to estimate such a function, an ideal situation for the purpose of benchmarking. Recently, our model and training methodology have been used to make progress in benchmarking quantum annealers for sampling [65], in contrast with the broadly explored topic of benchmarking combinatorial optimization.

The outline of this article is as follows. In Sec. II, we describe how graphical models with *effectively* arbitrary pairwise connectivity can be embedded and realized in quantum hardware. Here, we emphasize the parameter-setting problem, which is essential for any implementation in hardware. In Sec. III, we derive an

algorithm that tackles the parameter-setting problem while learning the model from data. In Sec. IV, we discuss the implementation details. In Sec. V, we describe the experiments performed on two synthetic data sets and a coarse-grained binarized version of the OptDigits data set; we show that the model introduced here, trained by using the D-Wave 2X (DW2X) hosted at NASA Ames Research Center, displays good generative properties and can reconstruct and classify data with good accuracy. In Sec. VI, we report the conclusions of our work, discuss the implementation of our approach in other hardware architectures such as the Lechner-Hauke-Zoller (LHZ) scheme [66], and present potential research directions.

II. HARDWARE-EMBEDDED MODELS

A. Quantum annealing and quantum models

The dynamics of a quantum annealer are characterized by the time-dependent Hamiltonian

$$\mathcal{H}(\tau) = -A(\tau)\sum_{i\in\mathcal{V}}\hat{X}_i - B(\tau)\mathcal{H}_P, \qquad (1)$$

where $\tau = t/t_a$ is the ratio between time t and annealing time t_a, while $A(\tau)$ and $B(\tau)$ are monotonic functions satisfying $A(0) \gg B(0) \approx 0$ and $B(1) \gg A(1) \approx 0$. The first term in Eq. (1) above corresponds to the transverse field in the x direction, characterized by the Pauli operators $\hat{X}_i$ for each qubit i. The second term in Eq. (1) corresponds to the problem-encoding Hamiltonian

$$\mathcal{H}_P = -\sum_{(i,j)\in\mathcal{E}} J_{ij}\hat{Z}_i\hat{Z}_j - \sum_{i\in\mathcal{V}}h_i\hat{Z}_i, \qquad (2)$$

where $\hat{Z}_i$ refers to the ith qubit in the z direction, which is defined on an interaction graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Here, $\mathcal{V}$ and $\mathcal{E}$ refer to the corresponding set of vertices and edges, respectively.

As discussed in Ref. [53], the dynamics of a quantum annealer are expected to remain close to equilibrium until they slow down and start deviating away from equilibrium to finally freeze out. If the time between such dynamical slow-down and freeze-out is small enough, the final state of the quantum annealer is expected to be close to the quantum Gibbs distribution

$$\rho = \frac{e^{-\beta_{\text{QA}}\mathcal{H}(\tau^*)}}{\mathcal{Z}}, \qquad (3)$$

corresponding to the Hamiltonian in Eq. (1) at a given point $\tau = \tau^*$, called freeze-out time. Here, β_{QA} is the physical temperature of the quantum annealer, and $\mathcal{Z}$ is the normalization constant. The density matrix in Eq. (3) is fully specified by the effective parameters $W_{ij} = \beta J_{ij}$, $b_i = \beta h_i$, and $c = \beta\Gamma$, where $\beta = \beta_{\text{QA}}B(\tau^*)$ is the effective

inverse temperature [21,53] and $\Gamma = A(\tau^*)/B(\tau^*)$ is the effective transverse field.

If $A(\tau^*) \ll B(\tau^*)$, the final state of the quantum annealer is close to a *classical* Boltzmann distribution over a vector of binary variables $\mathbf{z} \in \{-1, +1\}^N$,

$$P(\mathbf{z}) = \frac{e^{-\beta E(\mathbf{z})}}{\mathcal{Z}}, \qquad (4)$$

where

$$E(\mathbf{z}) = -\sum_{(i,j)\in\mathcal{E}} J_{ij}z_iz_j - \sum_{i\in\mathcal{V}}h_iz_i \qquad (5)$$

is the energy function given by the eigenvalues of $\mathcal{H}_P$ [see Eq. (2)].

The case where $A(\tau^*)$ cannot be neglected is less explored in the literature and allows the implementation of quantum Boltzmann machines [25]. All conditions described above, as well as the freeze-out time, depend on the specific instance of control parameters J_{ij} and h_i that are programmed. As shown in Sec. III, our algorithm can also train hardware-embedded models despite these unknown dependencies. The potential to train quantum models [25,26,65] opens new exciting opportunities for quantum annealing. These efforts resonate with foundational research interested in quantifying or identifying the particular computational resources that could be associated with quantum models [39,67].

B. Enhancing modeling capacity

In this section, we define the general setting to train a hardware-embedded probabilistic graphical model capable of representing graphs with arbitrary connectivity. Although we implement the general case of all-to-all connectivity as the most complex topology with pairwise interactions, working with models with simpler topologies can be easily represented with less numbers of qubits within this hardware-embedded setting.

In combinatorial optimization, one seeks a configuration of binary variables $\mathbf{z}$ associated with the lowest energy in Eq. (5). The typical strategy to embed dense graphs in quantum hardware is to represent logical binary variables by subgraphs of the interaction graph of physical qubits. The value of all control parameters should be fine-tuned such that the ground state of the original problem is preserved and therefore still favored in the physical implementation of the quantum annealing algorithm; this is known as the parameter-setting problem (see Sec. II C and Refs. [49,68,69]).

In machine learning, the scenario is different. When learning a model such as the one in Eq. (3) or Eq. (4), one seeks the configuration of control parameters J_{ij} and h_i that maximizes a suitable function of the data set. Notice that in combinatorial optimization problems, it is desirable to have

the optimal configuration or ground state with probability close to one. In machine learning, however, all configurations are significant, as are their corresponding probabilities. By mapping the original problem to quantum hardware as routinely done in combinatorial optimization applications, we may end up implementing a distribution that differs from the intended one.

An additional complication is that finding optimal parameters for a physical device is hampered by lack of precision and noise and by having to infer an instance-dependent effective temperature at each step of the learning. To avoid computing such an effective temperature at each learning iteration and to mitigate the effects of persistent biases [63], lack of precision, and noise in the control parameters, we take a gray-box model approach. In other words, although we assume that samples generated by the quantum annealer are distributed according to Eq. (3), we do not need complete knowledge of the actual parameters being implemented. This leads to the condition that the first- and second-order moments of the model and data distributions should be equal for the parameters to be optimal. The resulting model is nevertheless tied to the specific machine being used.

Using a generic logical graph as scaffolding, we associate each of its logical variables with a subgraph of the hardware interaction graph. This can be done by using existing minor embedding techniques; however, the parameter-setting problem remains. As an example, Figs. 2(a) and 2(b) show a simple graph that cannot be directly implemented in DW2X hardware and a possible minor embedding, respectively. The additional couplings inside a subgraph are part of the hardware-embedded graphical model and have to be learned along with the model parameters that couple different subgraphs. In other words, the fine-tuning of all the couplings is done by the learning algorithm, which has the potential to learn corrections to the noise affecting the physical components, under the assumption that these defects still respect the direction of the gradient driving the learning algorithm. The embedding also allows us to map the original data set into an extended data set with higher resolution, where some of the original variables are represented redundantly [see Figs. 2(c) and 2(d)]. Then, the learning algorithm runs entirely on such extended space by training from scratch the whole hardware-embedded model on the extended data set. We now discuss, in more detail, the parameter-setting problem and how it is tackled in the type of machine-learning applications studied here.

C. Parameter-setting problem

Let us define the parameter-setting problem as follows: Find values of control parameters to embed problems in hardware such that the performance of the device is "optimal." The meaning of optimal depends on the task of interest. In optimization problems, parameters are optimal if they provide the largest probability of finding a ground
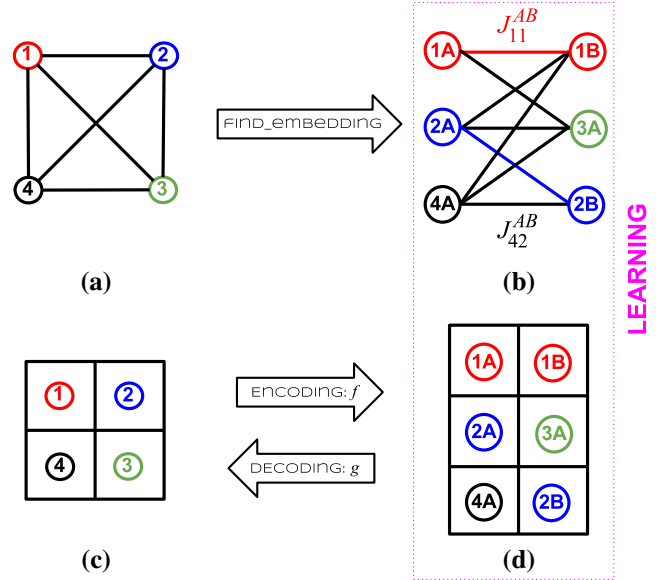

FIG. 2. *Hardware-embedded models.* (a) We first define a graph with arbitrary connectivity between the logical variables that directly encode the data set to be modeled; here, we show a fully connected graph on four variables as an example. Such a graph serves as a scaffolding to build hardware-embedded models with enhanced modeling capacity. (b) We then embed the scaffolding graph in quantum hardware by using minor embedding techniques; this requires the introduction of auxiliary qubits, couplings, and fields. In the example shown here, the logical variable 1 (red) is encoded using two qubits, $1A$ and $1B$, which are connected by an auxiliary coupler J_{11}^{AB}; the same is true for variable 2 (blue). The black links correspond to couplings between qubits representing different logical variables. In optimization problems, we are given values for the couplings and fields on the graph of logical variables, as in diagram (a). To solve such optimization problems on a quantum annealer, we first have to pick values for all control parameters such that the ground state of the physical system coincides with the optimal solution of the problem being solved. The selection of control parameters can be done via handcrafted rules when information is available about the model [69] or via heuristic approaches in the more general scenario [49]. The optimal choice of control parameters, i.e., the one that maximizes the probability of finding the ground state, is known as the parameter-setting problem, and it is an open research question. In machine-learning applications, instead, the control parameters are not given but have to be found; they are the variables of the problem. In this case, embedding techniques are used both to transform the original data set [as shown in diagram (c)] into a data set with higher resolution due to redundant variables [as shown in diagram (d)] and to find a representation of such an extended data set in hardware. This allows us to define encoding and decoding maps f and g, respectively, to transform between the two data representations. Then, we forget about the scaffolding graph and train the hardware-embedded model on the extended data set. Thus, although the final hardware-embedded model might be interpreted as an embedding of a logical model, this is certainly not the case during learning, which automatically tackles the parameter-setting problem; in a sense, machine learning *is* parameter setting. While here we use standard embedding techniques to define the maps f and g, such functions could, in principle, be learned from data, effectively automating the embedding problem, too.

state [69]. In the type of machine-learning applications considered here, parameters are optimal if samples generated by the device capture, as much as possible, the statistics of the data set. In a sense, machine learning *is* parameter setting. We discuss how this is quantified in Sec. III.

Previous research [69] suggests a possible mechanism underlying the parameter-setting problem. The authors investigated the Sherrington-Kirkpatrick model and found that the optimal choice of the additional parameters could be obtained by forcing both the spin glass and the ferromagnetic structures to cross the quantum critical point together during the annealing. Roughly speaking, the quantum phase transition happens when the energies associated with the problem-encoding system and the transverse field Γ are of the same order of magnitude. This implies that the optimal embedding parameters are $O(J_{SG}\sqrt{N})$, where N is the number of spins and J_{SG} is the typical value of the couplings, that is, the standard deviation (see, e.g., Fig. 2 in Ref. [69]). The intuition provided by this study does not necessarily apply to more realistic problems. In machine learning, even when starting from a fully connected model, the learning could still lead to a sparse final model.

As we discuss in Sec. III, our approach lets the data guide the process by treating the whole quantum annealer as a neural network. In this case, both the inter- and intra-subgraph parameters are modified according to the statistical errors made by the quantum annealer in producing samples that resemble the data. This process implicitly corrects for noise and defects on the parameters, problems that are expected to affect any near-term quantum technology. The price to pay is a relatively small overhead, as discussed in Sec. III.

In the following, we focus on hardware-embedded graphical models with *effective* all-to-all connectivity, as that is the most general case. Therefore, our derivations and the model proposed here include any topology with pairwise connectivity.

D. Fully connected inspired models

The sparse interaction topology of state-of-the-art quantum annealers strongly limits their capacity to model complex data. For this reason, we use embedding strategies based on utilizing several qubits to represent a given variable in the data set. This amounts to transforming the data set of interest into a higher-resolution data set, with redundant variables, and modeling it with a hardware-embedded model.

More specifically, consider a binary data set $\mathcal{D} = \{\mathbf{s}^1, ..., \mathbf{s}^D\}$, where each data point can be represented as an array of Ising variables, i.e., $\mathbf{s}^d = (s_1^d, ..., s_N^d)$, with $s_i^d \in \{-1, +1\}$, for $i = 1, ..., N$. We refer to the $\mathbf{s}$ variables as logical variables. We need to define a map f from the data space to the qubit space that produces an extended binary data set $\tilde{\mathcal{D}} = \{\mathbf{z}^1, ..., \mathbf{z}^D\}$, where $\mathbf{z} = f(\mathbf{s})$. In this

work, we choose the map f to replicate the state of each logical variable s_i inside the corresponding subgraph i, i.e.,

$$z_i^{(k)} = s_i, \quad \text{for } k = 1, ..., Q_i, \tag{6}$$

where Q_i is the number of qubits in subgraph i.

The task then turns into learning the parameters of a model on the extended data set $\tilde{\mathcal{D}}$. To do this, we define a problem Hamiltonian over $M = \sum_{i=1}^N Q_i$ qubits,

$$\tilde{\mathcal{H}}_P = -\frac{1}{2} \sum_{i,j=1}^N \sum_{k,l=1}^{Q_i, Q_j} J_{ij}^{(kl)} \hat{Z}_i^{(k)} \hat{Z}_j^{(l)} - \sum_{i=1}^N \sum_{k=1}^{Q_i} h_i^{(k)} \hat{Z}_i^{(k)}. \tag{7}$$

Here, N is the number of logical variables, which equals the number of subgraphs realized in hardware; $\hat{Z}_i^{(k)}$ is the Pauli matrix in the z direction for qubit k of subgraph i; $h_i^{(k)}$ is the local field for qubit k of subgraph i; and $J_{ij}^{(kl)}$ is the coupling between qubit k of subgraph i and qubit l of subgraph j. When $i = j$, it specifies the interactions within the subgraph, while when $i \neq j$, it specifies the interactions among subgraphs; $J_{ij}^{(kl)} = 0$ if there is no available interaction between the corresponding qubits in the quantum hardware. The binary variables $z_i^{(k)}$ encoding the extended data set can be interpreted as the eigenvalues of the Pauli matrix $\hat{Z}_i^{(k)}$.

After learning the parameters of $\tilde{\mathcal{H}}_P$ in Eq. (7), we need a map g from qubit space to data space that transforms samples generated by the quantum annealer into samples that resemble the original data set. Here, we choose g to assign the state of the majority of physical variables in subgraph i to the corresponding logical variable s_i, i.e.,

$$s_i = \text{sign}\left(\sum_{k=1}^{Q_i} z_i^{(k)}\right). \tag{8}$$

The rationale behind this choice is that, ideally, samples from the trained model are expected to have all qubits $z_i^{(k)}$ in a subgraph i having exactly the same state, i.e., $z_i^{(k)} = z_i^{(l)}$ for $k, l = 1, ..., Q_i$. In this case, we could pick whichever qubit $z_i^{(k)}$ as representative of the logical variable s_i, and this choice would be equivalent to the choice in Eq. (8). However, we expect the choice in Eq. (8) to be more robust to the different sources of noise in quantum annealers by exploiting such a redundancy in the spirit of error-correction codes [61,62]. While we have *a priori* fixed mappings f and g using embedding techniques, such functions could also be learned from data, as we will discuss elsewhere.

III. LEARNING ALGORITHM

Let $\rho_{\mathcal{D}}$ be the diagonal density matrix whose diagonal elements encode the empiric data distribution. A quantum

Boltzmann machine [25], characterized by the density matrix ρ defined in Eq. (3), can be trained by minimizing the quantum relative entropy [26],

$$S(\rho_{\mathcal{D}}\|\rho) = \mathrm{Tr}\rho_{\mathcal{D}}\ln\rho_{\mathcal{D}} - \mathrm{Tr}\rho_{\mathcal{D}}\ln\rho. \qquad (9)$$

The learning rule is given by the equations

$$J_{ij}^{(kl)}(t+1) = J_{ij}^{(kl)}(t) + \eta\frac{\partial S}{\partial J_{ij}^{(kl)}}, \qquad (10)$$

$$h_i^{(k)}(t+1) = h_i^{(k)}(t) + \eta\frac{\partial S}{\partial h_i^{(k)}}, \qquad (11)$$

where t indicates the iteration and $\eta > 0$ is the learning rate. Assuming we can neglect the dependence of the time lapsed between the dynamical slow-down and freeze-out in the instance of control parameters $J_{ij}^{(kl)}$ and $h_i^{(k)}$ programmed, we obtain

$$\frac{1}{\beta}\frac{\partial S}{\partial J_{ij}^{(kl)}} = \langle\hat{Z}_i^{(k)}\hat{Z}_j^{(l)}\rangle_{\rho_{\mathcal{D}}} - \langle\hat{Z}_i^{(k)}\hat{Z}_j^{(l)}\rangle_{\rho}, \qquad (12)$$

$$\frac{1}{\beta}\frac{\partial S}{\partial h_i^{(k)}} = \langle\hat{Z}_i^{(k)}\rangle_{\rho_{\mathcal{D}}} - \langle\hat{Z}_i^{(k)}\rangle_{\rho}. \qquad (13)$$

Here, $\langle\cdot\rangle_{\rho_{\mathcal{D}}}$ denotes the ensemble average with respect to the density matrix $\rho_{\mathcal{D}}$ that involves *only* the data and is commonly referred to as the positive phase. Similarly, $\langle\cdot\rangle_{\rho}$ denotes the ensemble average with respect to the density matrix ρ that exclusively involves the model and is called the negative phase.

If $A(\tau^*) \ll B(\tau^*)$ during our experiments, we are indeed dealing with classical models. Then, the learning rule above coincides with that for maximizing the average log-likelihood of the data [70].

However, the more general quantum case we just described provides a more accurate representation of the experiments we have performed, which are described below. To provide a strong argument as to which is the case, though, we need to carry out numerical simulations of the open quantum systems dynamics undergone by quantum annealers. We leave this for future work. However, our approach would also be valid for quantum annealers capable of sampling from any desired fixed value of the transverse field, a capability that may be available in the near future.

The Hamiltonian in Eq. (7) is designed to overcome connectivity limitations of hardware-embedded graphical models. In what follows, we show that the adaptation of standard learning procedures to the quantum maximum-entropy model proposed here works very well even in the presence of unknown hardware noise on couplings $J_{ij}^{(kl)}$ and local fields $h_i^{(k)}$. Moreover, we can learn suitable intra-subgraph couplings at a rate dictated by the contrast of the strength of pairwise correlations in the model and in the data, without the need for hard-coded values.

Classically, the exact computation of the model's statistics is a computational bottleneck due to the intractability of computing the partition function and the exponential number of terms in the configuration space. An efficient approximation of the statistics is therefore required, and it is usually carried out by standard sampling techniques such as MCMC [70,71]. In this work, we instead implement an algorithm that relies on the working assumption that quantum annealers can generate samples from a Gibbs-like distribution. However, even if this assumption is not strictly valid, our approach can still work as long as the estimated gradients have a positive projection in the direction of the true gradient. Quantum annealers have the potential to improve machine-learning algorithms in two ways: (i) by enabling the exploration of a wider class of models, i.e., quantum models, which some theoretical results [39] suggest may be able to capture higher complexity, and (ii) by speeding up the generation of samples. If the transverse field at the freezing point is negligible, the samples generated by the quantum annealer are expected to approximately follow a classical Boltzmann distribution.

The learning procedure implemented by Eqs. (12) and (13) can be interpreted as quantum entropy maximization under constraints on the first- and second-order moments [72–74]. In Ref. [24], a maximum entropy approach was implemented on a D-Wave device in the context of information decoding, which is a hard optimization problem. Instead, we use quantum maximum-entropy inference for a hard machine-learning task, i.e., in unsupervised learning of generative models.

Equation (10) implies that the intra-subgraph couplings $J_{ii}^{(kl)}$ increase at a varying rate proportional to $1 - \langle\hat{Z}_i^{(k)}\hat{Z}_i^{(l)}\rangle$, which, in principle, leads to infinite values in the long term. In practice, the rate of growth decreases as the learning progresses since the statistics of the samples generated by the quantum annealer resemble the data more and more. In general, the gradient-descent learning rule tends to produce too-large values for *all* the parameters because it pushes the model as much as possible towards a distribution with all the mass concentrated on the data. This problem is known as overfitting, and it is usually approached by regularization techniques. One regularization method may consist in penalizing large parameters by adding a term to Eq. (9) accordingly. Another approach may be to employ a stopping criterion based on some measure of generalization or predictive capabilities of the model evaluated at each iteration on data not used during training. Under a proper choice of regularization, the intra-subgraph couplings utilized in our approach should not grow indefinitely anymore. However, regularization in the general setting of quantum machine learning is still an open research question.

Regarding the complexity of the algorithm, a fully connected model with N logical variables has $O(N^2)$

parameters. When embedding such a fully connected model into a sparse graph like the Chimera graph of the DW2X, we end up with $O(N^2)$ qubits, but the number of parameters is still $O(N^2)$. This result occurs because we go from a dense graph of N variables to a sparse graph of $O(N^2)$ variables. Each qubit in the DW2X interacts with, at most, six neighbors, so the number of additional parameters is a small constant factor. In our experiments, this factor is about 3 (see Table I). Because of this factor, there is a small computational overhead for learning those intra-subgraph parameters. This overhead could be neglected because the main bottleneck is still in the generation of samples which is at least as hard as any non-deterministic polynomial time problem (NP-hard). An exact analog occurs in combinatorial optimization where a quadratic overhead is expected for embedding fully connected problems in hardware. In combinatorial optimization, such overheads are usually neglected because the main bottleneck is the NP-hard problem of reaching low-energy configurations.

A few additional remarks are in order: (i) The assumption that the model is based on a Gibbs distribution is reflected in that the second moment between two variables influences only the update of the corresponding coupling between them. If such a second moment increases (decreases), so does the corresponding coupling. This leaves open the possibility for the model to effectively self-correct for relatively small deviations from equilibrium, persistent biases, noise, and lack of precision, as long as the estimated gradient has a positive projection in the right direction, in the spirit of simultaneous perturbation stochastic approximation [11,76]. (ii) The actual shape of a Gibbs distribution is instead characterized by the variables $\beta J_{ij}^{(kl)}$ and $\beta h_i^{(k)}$. Writing Eqs. (10) and (11) in terms of these new variables, we observe that the actual learning takes place at an effective learning rate that can vary since the effective temperature is instance dependent [21]. (iii) The positive phases in Eqs. (12) and (13) are constants to be estimated exclusively from the data points, as there are no hidden units in our approach. In the case of generic models with hidden variables, this term becomes difficult to compute, in general, and we have to rely on approximations, e.g., via sampling or mean-field techniques. (iv) The related problem of estimating the parameters of a *classical* Ising model is called the inverse Ising problem [56,57,77], and some of the main alternative techniques are mean-field and pseudo-likelihood methods.

IV. IMPLEMENTATION DETAILS

A. Device and embeddings

We run experiments on the DW2X quantum annealer located at NASA Ames Research Center. The device is equipped with 1152 qubits interacting according to a graph known as Chimera connectivity. For the DW2X device hosted at NASA Ames, only 1097 qubits are functional and available to be used. Assuming all 1152 qubits were available, an efficient embedding schema [78] would allow us to implement a fully connected graph with up to 48 logical variables. Since only 1097 qubits are available, such a schema cannot be used, and the size of the largest fully connected model that can be implemented is reduced. For the embeddings of the instances studied here, we run the `find_embedding` heuristic [75] offered by D-Wave's programming interface and use the best embedding found within the 500 requested trials. We judge the quality of an embedding not only by the total number of physical qubits needed to represent the logical graph, but also by considering and preferring a smaller maximum subgraph size for the logical units. For example, in the case of the 46-variable fully connected graph, we found an embedding with 917 qubits and a maximum subgraph size of 34. We selected, instead, an embedding with a larger number of qubits, 940, but with a considerably smaller maximum subgraph size of 28. (Figure 8 in the Appendix shows the selected embedding, where each subgraph is represented by a number and a color.) Table I shows details for each of the embeddings used in our experiments. Finally, the parameter range allowed by DW2X is $J_{ij}^{(kl)} \in [-1, +1]$ and $h_i^{(k)} \in [-2, +2]$. We initialized all the parameters to small values in $[-10^{-6}, +10^{-6}]$ in order to break the symmetry.

B. Data sets and preprocessing

We tested our ideas on the real OptDigits data set [58], the synthetic BAS data set [59], and a collection of synthetic data sets generated from random Ising instances.

The OptDigits data set requires the preprocessing steps shown in Fig. 3. First, each picture is 8×8 and has a categorical variable indicating the class it belongs to. Using standard one-hot encoding for the class (i.e., $c_i^d = -1$ for $i \neq j$, $c_j^d = +1$, where j indexes the class for picture d), we would need to embed a fully connected graph of 74 variables, 64 for the pixels and 10 for the class, exceeding what we can embed in the DW2X. We removed the leftmost and rightmost columns as well as the bottom row from each picture, reducing the size to 7×6 and retaining the

TABLE I. Main characteristics of the different embeddings used here, for each of the fully connected graphs. All embeddings were generated by the `find_embedding` [75] heuristic provided by D-Wave's API. The table includes information about the minimum (Min) and maximum (Max) subgraph size, the percentage of qubits used relative to those available (Chip usage), and the total number of parameters for the logical and physical graphs.

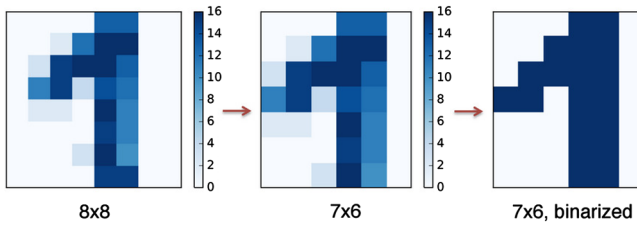| Logical variables	Physical variables	Min	Max	Chip usage	Logical parameters	Physical parameters
15	76	5	6	7%	120	252
42	739	11	25	67%	903	2644
46	940	12	28	86%	1081	3389

FIG. 3. *OptDigits preprocessing steps.* The original 8×8 pictures are cropped to 7×6 arrays by removing columns from the left and the right, as well as by deleting a row from the bottom. Finally, the four-bit gray scale is thresholded at the midpoint and binarized to $\{-1, +1\}$. Figure 4(a) shows some pictures from the test set.

readability. Second, we selected only four classes of pictures, those corresponding to digits "one" to "four," reducing the one-hot encoding to four variables. The four classes account for 1545 pictures in the training set and 723 pictures in the test set, and they are in almost equal proportion in both. Finally, the original four-bit gray scale of each pixel is thresholded at the midpoint and binarized to $\{-1, +1\}$ in order for the data to be represented by qubits in the DW2X. Figure 4(a) shows some pictures from the test set.

The BAS data set consists of $N \times M$ pictures generated by setting the pixels of each row (or column) to either black (-1) or white $(+1)$, at random. A reason to use this synthetic data set is that it can be adapted to the number of available variables in the DW2X. Having found an embedding for the 42-variable fully connected graph, we generated a 7×6 BAS data set consisting of 192 pictures of 42 binary variables each. Then, we randomly shuffled the pictures and split the data set into training and test sets of size 96 each. Figure 5(a) shows some pictures from the test set.

Finally, for the collection of synthetic data sets, we preferred to work with small-sized Ising instances that allowed us to carry out exhaustive computations. In particular, we chose 10 random instances of a Sherrington-Kirkpatrick model with $N = 15$ logical variables. Parameters J_{ij} [cf. Eqs. (4) and (5)] were sampled from a Gaussian with mean $\mu = 0$ and standard deviation $\sigma = \zeta/\sqrt{N}$, parameters h_i were set to 0, and the inverse temperature was set to $\beta = 1$. In this setting, a spin-glass transition is expected when $\zeta_c = 1$ in the thermodynamic limit, although finite-size corrections are expected to be relevant for this small size. In order to obtain interesting structures within the probability distributions, we chose $\zeta = 2$ and verified that the overlap distribution [60,61] of each instance was indeed nontrivial. Moreover,
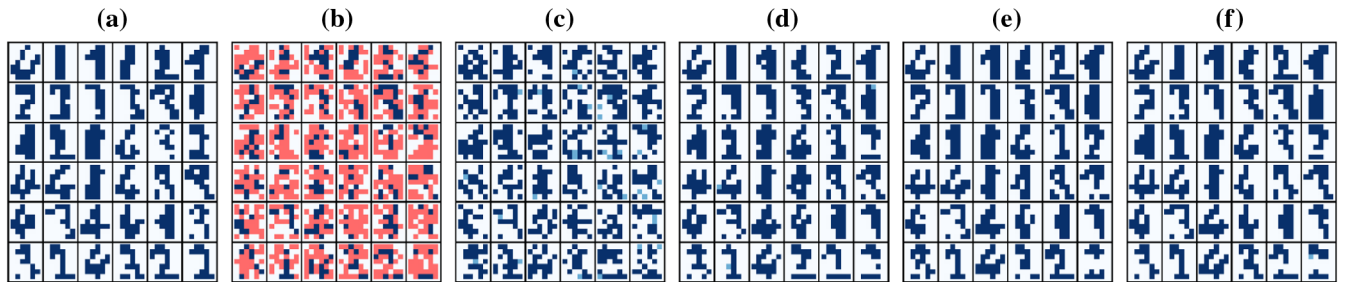

FIG. 4. *OptDigits experiment.* (a) We show 36 samples from the test set, with each pixel being either dark blue $(+1)$ or white (-1). See Fig. 3 and the main text for a description of the preprocessing steps. (b) A uniform salt-and-pepper noise shown in red corrupts each picture. The model cannot use information from the red area. (c)–(f) Reconstructions obtained after 1, 10, 1000, and 6000 learning iterations. A light blue pixel indicates a tie of the majority vote over the corresponding subgraph. We can visually verify that the model has learned to generate digits. The learning stops at iteration 6000 because further iterations would bring some parameters out of the dynamic range of the DW2X device.
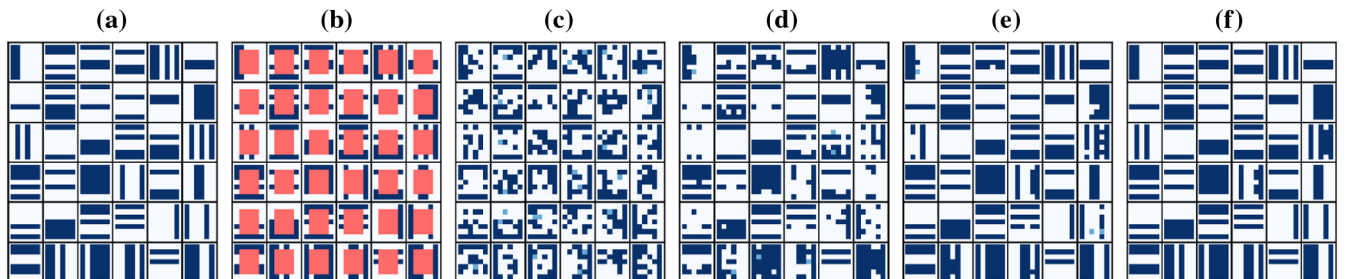

FIG. 5. *BAS experiment.* (a) We show 36 samples from the test set, with each pixel being either dark blue $(+1)$ or white (-1). (b) A 5×4 block of noise shown in red corrupts each picture. The model cannot use information from the red area, and yet the remaining pixels contain enough information to reconstruct the original picture. (c)–(f) Reconstructions obtained after 1, 10, 1000, and 3850 learning iterations. The average number of mistaken pixels is 50% in (c), 18.6% in (d), 2.95% in (e), and finally 0.65% in (f). This is an almost perfect reconstruction. The learning stops at iteration 3850 because further iterations would bring some parameters out of the dynamical range of the DW2X device.

TABLE II. Main characteristics of the data sets used here, i.e., number of variables, number of training points, and number of test points, when applicable. The * symbol indicates a synthetic data set.

Data set	Variables	Training points	Test points
OptDigits	$42 + 4$	1545	723
BAS 7×6*	42	96	96
Ising*	15	150	Not applicable

TABLE III. Settings used in all the experiments except those in Sec. V C, where gradient hyperparameters were tuned.

Domain	Hyperparameter	Value
Device	Annealing time	5 μs
	Programming thermalization	1 μs
	Readout thermalization	1 μs
	Auto scale	False
Gradient	Learning rate	0.0025
	$L2$ regularization	10^{-5}
	Momentum	0.5

we checked the performance of the closed-form solutions obtained by mean-field techniques in Ref. [57]. The mean-field method failed to produce (real-valued) solutions in seven out of the ten random instances, while it performed well in the remaining three instances, adding further evidence that these instances had nontrivial features in their energy landscape. Finally, we generated a training set of $D = 150$ samples for each instance by exact sampling from its corresponding Boltzmann distribution. Table II summarizes the characteristics of each dataset used in our experiments.

C. Choice of hyperparameters

We distinguish two kinds of hyperparameters: those associated with the device and those referring to the gradient. Device hyperparameters affect the time needed to obtain samples. We set them to their corresponding minimum values in order to obtain samples as fast as possible. Gradient hyperparameters come from advanced techniques known to improve generalization and speed up learning. We adopt standard $L2$ regularization for the pairwise interactions and momentum for all the parameters, hence introducing two hyperparameters in Eqs. (10) and (11) (see Ref. [71] for discussion about implementation details and best practices). For these hyperparameters, we tried a small grid of values and chose the value that would allow the quantum-assisted algorithm to produce visually appealing samples. All the experiments were performed using the hyperparameters shown in Table III.

V. RESULTS

A. Reconstruction of pictures

The first task we address is verifying that the model is indeed able to learn the joint probability distribution of variables given a data set. One way to do this is to check whether the learned model can reconstruct corrupted pictures. To generate a reconstruction, we first need to enforce the value of each *correct* pixel to all qubits of the corresponding subgraphs, as illustrated in Fig. 1(b). The qubits can be clamped to the desired value by using a strong local field in the corresponding direction. Notice that clamping variables in quantum annealers is somewhat different from its classical counterpart. Applying a strong local field to a qubit can substantially bias it towards a given value, but it still remains a dynamical variable. In classical computation, clamping a variable completely freezes it. We then generated samples from the learned model and assigned values to each corrupted pixel s_i using the majority-vote map [Eq. (8)] for all qubits in the corresponding subgraph i. To further mitigate the noise associated with this, we generated multiple reconstructions, 100 for each corrupted picture, and took a second majority vote over them. This approach is very robust as we did not observe any mismatch between the desired clamped variables and the corresponding readouts. We chose to interrupt the training of the models as soon as any of the parameters left the dynamic range of the device. Since the intra-subgraph couplings always increase, we expect these to be the first to get out of range, and we observed this result in the experiments described below. We use two data sets, OptDigits and BAS.

1. Optical recognition of handwritten digits

We trained a model on the real data set OptDigits, a sample of which is shown in Fig. 4(a). Since the training set contains a relatively large number of data points, we opted for a minibatch learning approach [71], where 200 data points were used at each iteration to compute the positive phase of the gradient. The negative phase is computed on 200 samples from DW2X. We trained for 6000 iterations, after which an intra-subgraph coupling went outside the dynamic range of the device.

To evaluate the model, we added a 50% uniformly distributed "salt-and-pepper" noise [Fig. 4(b), red pixels] to each picture of the test set and used the model to reconstruct it. Notice that, given a corrupted picture, it is not always possible to obtain perfect reconstruction as multiple solutions could be correct. Therefore, we do not compute any error measure but rather visually inspect the reconstructions. Figures 4(c)–4(f) show some reconstructions obtained by models learned after 1, 100, 1000, and 6000 iterations, respectively. We can observe that qualitatively good reconstructions are already available from early stages of training. However, the large degree of corruption in the original image gives rise to things such as thicker reconstructions [Fig. 4(f), third row, fourth column], thinner reconstructions [Fig. 4(e), fourth row, second column], change of digit "three" to "one" [Fig. 4(e), third row, fifth column], among others.

2. Bars and stripes

We performed a similar test on the 7×6 BAS, a sample of which is shown in Fig. 5(a). We computed the positive

phase once using all 96 training data points. Then, we ran the learning procedure, and for each iteration, we computed the negative phase out of 96 samples obtained from the DW2X. The learning process stopped at iteration 3850, after which an intra-subgraph coupling exceeded the maximum value allowed.

To evaluate the model, we blacked-out a 5×4 block [Fig. 5(b), red pixels corresponding to 47.6% of the image] from each of the 96 test pictures and used the model to reconstruct it. We can observe from Fig. 5(e) that reconstructed pictures are qualitatively similar to the original ones. To have a quantitative estimate of the quality of the reconstruction, we computed the expected number of incorrect pixel values (or mistakes) per reconstruction. After one iteration [Fig. 5(c)], we obtained a rate of 10.45 mistakes out of 20 corrupted pixels, corresponding to about 50% performance as expected. The number of mistakes decreased to 3.73 (18.6%) after 100 iterations [Fig. 5(d)], 0.59 (2.95%) after 1000 [Fig. 5(f)], and finally 0.13 (0.65%) at the end of training [Fig. 5(e)]. The latter result corresponds to almost perfect reconstruction. Notice that, by definition, pictures from the test set are never used during training. Hence, these results provide evidence that the joint probability distribution of the pixels has been correctly modeled, and we can most likely rule out a simple memorization of the patterns.

B. Generation and classification of pictures

To investigate the generative and classification capabilities of the model, we introduced a one-hot encoding of the four classes of the OptDigits data set, therefore introducing four additional logical variables, for a total of 46. We trained this larger model on the OptDigits data set, also including the classes.

We performed a simple classification task that does not require turning the generative model into a discriminative one by additional post-training. We classify each test picture as $c^* = \arg\max_c P(\mathbf{c}|\mathbf{s})$, where $\mathbf{s}$ is the vector encoding the pixels and $\mathbf{c}$ is the vector encoding the classes. To approximate the probability, we clamped the subgraphs, by applying strong local fields, to the pixel values corresponding to the picture to be classified and sampled the four class variables from DW2X. We generated 100 samples for each picture and assigned the picture to the most frequent class. After 6000 learning iterations, this simple procedure led to an accuracy of 90% on the test pictures. This is a significant result, given that a random guess achieves 25% accuracy. However, it is to be expected that a fine-tuned discriminative model can achieve better accuracy.

Finally, Fig. 6 shows samples obtained from the DW2X by first setting the class variables, by applying strong local fields, to classes one to four (one class per column), along with human-generated pictures from the test set. Rows correspond to either human-generated pictures from the test set or machine-generated pictures. We defer the details of this visual Turing test to Ref. [79]. Machine-generated pictures are remarkably similar, though not identical, to those drawn by humans. Notice that human-generated digits may be ambiguous because of a variety of calligraphy styles encoded in low-resolution pictures. This ambiguity has been captured by the model, as shown by the machine-generated pictures.

C. Learning of an Ising model

In the previous section, we showed that quantum annealing can be used to successfully train the hardware-embedded models introduced here on both real and synthetic data sets of pictures. Here, we compare physical and logical
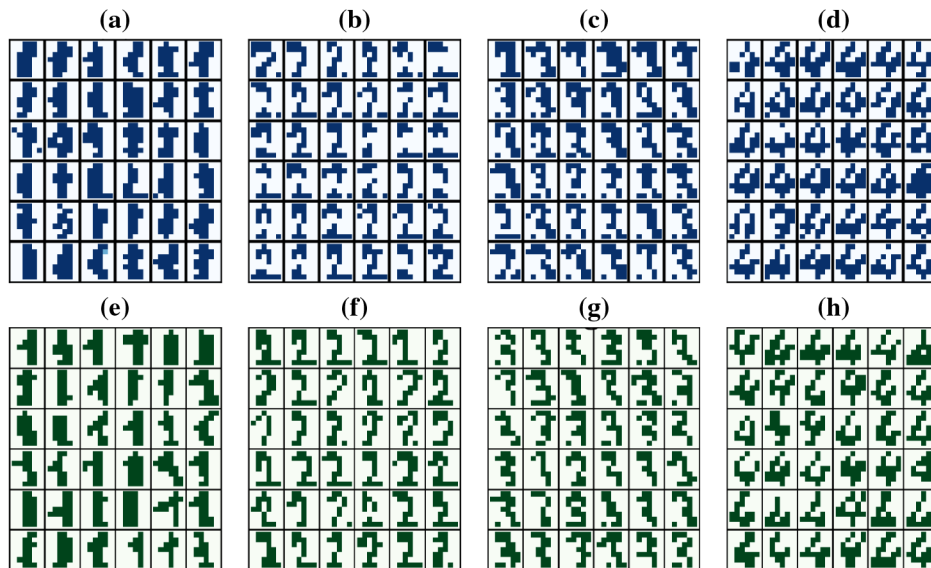

FIG. 6.   *Visual Turing test(a)–(h)*. The reader is invited to distinguish between human-generated pictures from the test set and machine-generated pictures sampled from the model. Columns identify classes one to four; rows identify the source—human or machine. The solution is given in Ref. [79].

models trained by quantum annealing (QA), simulated thermal annealing (SA) [80], and exact gradient. To simplify this task, we now deal with synthetic data sets composed of $D = 150$ samples generated exhaustively from small-sized Boltzmann distributions as described in Sec. IV B. This is similar in spirit to the approach usually taken in the literature on the inverse Ising problem [56,57]. However, we do not quantify the quality of the trained model by the quadratic error between the parameters of the original model and those obtained by the learning algorithms, as it is usually done, for three reasons: (i) The physical model implemented in quantum hardware has a larger number of parameters than the logical model from which the data are generated, and a direct comparison is not straightforward. (ii) In our gray-box model approach, we do not have direct access to the effective parameters implemented in the quantum annealer, so we have to estimate the effective temperature that can introduce errors. (iii) To our knowledge, there is no direct connection between generic distances in parameter space, as measured by the quadratic error, and distances in probability space, which are those that have actual operational meaning, except perhaps when the parameters are close enough. Indeed, to measure distances in parameter space that correspond to distances in probability space, it is necessary to use the Fisher information metric. For instance, it is known that, close to a critical point, a slight variation in the parameters can lead to drastically different probability distributions [81].

Instead, our evaluation strategy exploits the fact that we have full knowledge of the probability distribution $Q(\mathbf{s})$

that generated the data. At each learning iteration, or epoch, we sample a set $\mathcal{S} = \{\mathbf{s}^{(1)}, ..., \mathbf{s}^{(L)}\}$ of L points from the model $P(\mathbf{s})$ and evaluate the average log-likelihood that such samples were generated by $Q(\mathbf{s})$,

$$\Lambda_{\mathrm{av}}(\mathcal{S}) = \frac{1}{L} \sum_{\ell=1}^{L} \log Q(\mathbf{s}^{(\ell)})$$

$$= -\beta \frac{1}{L} \sum_{\ell=1}^{L} E(\mathbf{s}^{(\ell)}) - \log Z(\beta); \quad (14)$$

for simplicity, we chose $L = D = 150$.

Notice that Eq. (14) requires full knowledge of the distribution that generated the data. This is unfeasible for real data sets since the whole point of learning a model is precisely that we do not know the true underlying distribution. However, this proxy is related to the generalization properties of the trained model since it corresponds to the likelihood that new arbitrary samples generated by the model were actually generated by the true underlying distribution. We expect this to be a faithful proxy since achieving good generalization performance is the main objective of machine-learning techniques. However, we should take into account that $\Lambda_{\mathrm{av}}(\mathcal{S})$ is not expected to be maximized by the generated samples but rather to match the value $\Lambda_{\mathrm{av}}(\mathcal{D})$ of the original data set.

In this set of experiments, we performed 500 learning iterations and did not use gradient enhancements such as momentum and regularization in order to simplify the
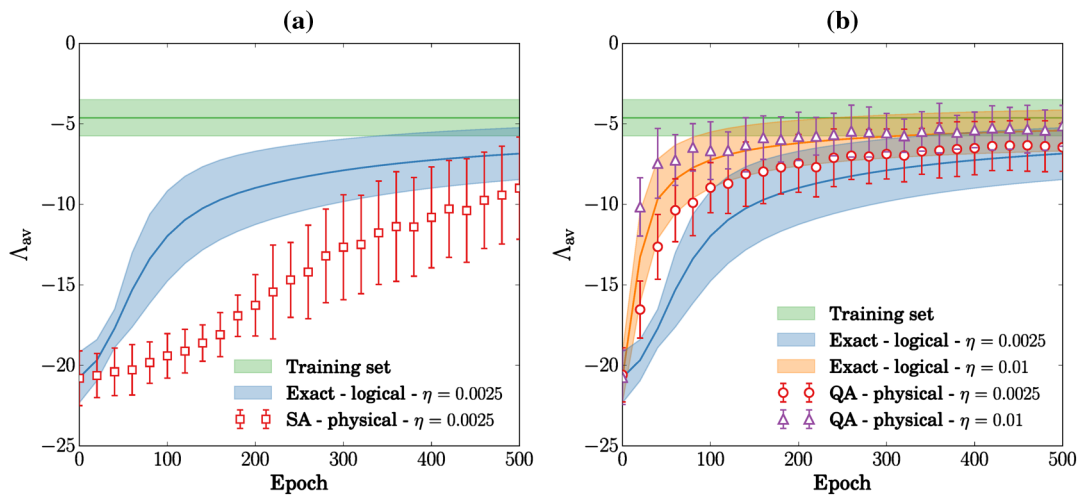

FIG. 7.   *Comparison of different learning settings.* The plots show mean and 1 standard deviation of the proxy Λ_{av} for 10 random instances and for different learning procedures. We use exact gradient for the 15-variable logical graph and quantum annealing (QA) or simulated thermal annealing (SA) for the corresponding 76-qubit physical graph. A learning procedure is considered successful if it can generalize, that is, if it matches the proxy of the training set (green band). (a) The logical model (blue band) matches faster than the physical model (red squares) when the same learning rate is used. This suggests that the larger number of parameters does not help the physical model. (b) Quantum annealing on the physical graph (red circles) enables faster matching than exact gradient on the logical graph (blue band) when the same learning rate $\eta = 0.0025$ is used. However, the exact-gradient procedure equipped with a larger learning rate $\eta = 0.01$ (orange band) outperforms the quantum-assisted algorithm. In turn, the quantum-assisted algorithm outperforms all other learning procedures when equipped with the larger learning rate $\eta = 0.01$ (purple triangles). Notice that neither the computation of Λ_{av} nor the exact-gradient learning is tractable, in general.

quantitative analysis. First, we verified whether the larger number of parameters in the physical graph provides a practical advantage against the logical models. While exact gradient calculations are feasible in the 15-variable logical graph, they are infeasible for the 76-qubit physical graph considered here (see details in Table I). We opted for a sampling procedure based on SA where each sample follows its own independent linear schedule, therefore avoiding the problem of autocorrelation among samples. We used a linear schedule $\beta(t) = t/t_{\max}$ for the inverse temperature and performed a preliminary study in order to set the optimal number of Monte Carlo spin flips per sample, $t_{\max}$. We incrementally increased this number and observed the change in learning performance via the proxy Λ_{av}. We choose $t_{\max} = 15200$ Monte Carlo spin flips, as multiples of this number did not result in improved learning speed nor in better values of Λ_{av}. We expect this procedure to be essentially equivalent to exact gradient within the 500 learning iteration considered here. Figure 7 shows mean and 1 standard deviation of the performance indicator for the 10 synthetic instances considered here. Figure 7(a) indicates that SA-based learning on the physical graph (red squares) is slower than exact gradient learning on the logical graph (blue band) when the same learning rate is used. Even though both methods approach the optimal Λ_{av} of the data set (green band), the larger number of parameters does not speed up learning. Despite this, Fig. 7(b) shows that quantum-assisted learning with $\eta = 0.0025$ (red circles) outperforms exact gradient. This indicates that a varying effective learning rate could be induced by the instance-dependent effective temperature [21] at which a quantum annealer samples. Indeed, by increasing the learning rate of the exact gradient method to $\eta = 0.01$ (orange band), we were able to outperform quantum-assisted learning. In turn, however, quantum-assisted learning can outperform exact gradient if the same larger learning rate is used (purple triangles). The fast initial learning could also be caused by a nonvanishing transverse field at the freeze-out point (see Sec. III above for a discussion). Because of the interplay between effective temperature and learning rate, the experiments presented here cannot confirm nor rule out the presence of these quantum effects. Open-quantum-systems simulations on small systems might give us greater control and allow us to have further insights into the interplay of the mechanisms proposed here. We leave this task for future work.

VI. CONCLUSIONS

Whether quantum annealing can improve algorithms that rely on sampling from complex high-dimensional probability distributions or whether they can provide more effective models are important open research questions. However, quantum annealers face several challenges that need to be overcome before we can address such a question from an experimental perspective. Besides the problem of proper

temperature estimation, which has been addressed recently [21,52], some of the most pressing challenges are sparse connectivity which limits the capacity of the models that can be implemented, low precision, and limited range of the control parameters, as well as different sources of noise that affect the performance of state-of-the-art quantum annealers [55].

By combining standard embedding techniques with the data-driven automatic setting of embedding parameters, we substantially improve the robustness and the complexity of machine-learning models that can be modeled with quantum annealers. By working on a gray-box model framework, which requires only partial information about the actual distribution from which the quantum annealer is sampling, this approach also avoids the need for estimating temperature during the learning of the models and has the potential to help mitigate the different sources of noise on the device. The resulting model can be interpreted as a visible-only quantum Boltzmann machine with all pairwise interactions among logical variables. We validated our ideas qualitatively by training the fully connected hardware-embedded model for reconstruction and generation of pictures, and quantitatively by computing a proxy on data sets extracted from randomly generated Boltzmann distributions.

Another advantage of our approach is that the learning rules are embedding-agnostic. More precisely, the underlying hardware embedding for the scaffolding logical model can be found by either heuristic algorithms [75] or by known efficient schemes [78,82], and the learning strategy is the same. While we have *a priori* fixed mappings f and g using embedding techniques, such functions could also be learned from data, as we will discuss elsewhere.

Furthermore, the strategy for training can be straightforwardly extended to other proposed hardware architectures, such as the LHZ scheme [66]. More specifically, the data from the machine-learning task can be easily mapped to the physical qubits of that scheme by following the equivalent of our Eq. (6). One difference is that the gradient updates [see Eqs. (12) and (13)] for the programmable parameters in this case will involve the updates of bias terms and the penalties for the quartic terms, under that choice of hardware implementation. This does not pose any challenges with our approach either, and the final results of the same iterative learning procedure detailed here would be a trained quantum or classical model. By using this gray-box model, one can also get samples from a LHZ-type device and use it for useful tasks such as the digit reconstruction or generation as illustrated in this work. The question of whether there is any advantage of either implementation for the machine-learning tasks proposed here is a question that would need to be addressed in future work.

Natural extensions of the model will be inclusion of latent variables, also known as hidden units, support for continuous variables, and the development of techniques for the quantum annealer to also learn the embedding from data.

Hidden units are needed, for example, if visible patterns require constraints that cannot be enforced by pairwise interactions alone [70]. Continuous variables are needed for a correct modeling of real data sets. This has been the focus of recent work [37,38], where we used the same gray-box model developed here but on a fully connected graph of 60 hidden variables (instead of visible units). We performed experiments on a hardware-embedded model with 1644 qubits, further supporting the robustness-to-noise claims in this work and the value of this approach as a template for other quantum-assisted frameworks.

Another possible direction for future work is the extension of our learning algorithm to more general, possibly non-equilibrium, distributions. As we discussed, our learning algorithm might still work when there are relatively small deviations from the thermal distribution we assumed, as long as the estimated gradient has a positive projection on the direction of the true gradient. Indeed, if we had no information on the state reached by the quantum annealer, we would have to rely on model-free (e.g., black-box) techniques based, for instance, on randomly choosing a direction to update the control parameters, which may be highly inefficient [11,76]. On the other hand, if we had complete knowledge of the final state reached by the quantum annealer, we could benefit from model-based techniques, as we have done here. A possible hybrid algorithm may be based on a model, e.g., a Gibbs distribution, that captures the most relevant features of the quantum annealer state and some unknown corrections. In this way, the algorithm may choose a direction in parameter space informed by the model, instead of just randomly, and use the black-box techniques to correct for mistakes.

From a more fundamental perspective, several key questions remain open: When and why could the quantum annealer do better than classical MCMC approaches, or when and why could it provide more effective models? Our results show that the quantum-assisted learning algorithm has a faster learning during the initial stage, in the scenario where both classical (exact gradient estimation and SA) and our hybrid quantum-classical approach are set under the same conditions in terms of hyperparameters. Given that an instance-dependent effective temperature can imply a varying learning rate, this faster learning is probably due to the quantum-assisted algorithm automatically adjusting its learning rate. In this respect, it is important to investigate if such a learning schedule is optimal and, if so, whether it can be effectively simulated by classical means. Still, we cannot discard that some nontrivial quantum effects play a role here. Indeed, as pointed out in Ref. [26], and as we further discussed above, the learning rules for classical and quantum Boltzmann machines coincide when there are no hidden variables. The potential to train quantum models [25,26,65] opens new exciting opportunities for quantum annealing. These efforts resonate with foundational research interested in quantifying or identifying the computational resources that could be associated with quantum models [39,67].

Arguably, this question of whether or not there is quantum speedup in sampling applications is one of the most important questions propelling our research. Years of experience accumulated with the use of quantum annealers for combinatorial optimization suggest that the answer may not be straightforward [83,84], with the first comprehensive benchmarking study on an industrial application performed only recently [51]. Benchmarking quantum annealing for machine learning can be approached by following well-established guidelines used in optimization (see Ref. [85]). However, the iterative nature of most machine-learning applications makes the task far more time-consuming. Almost all the hyperparameters (e.g., learning rate, annealing time, number of samples per iteration, etc.) can be adjusted throughout the learning, hence requiring us to find an optimal schedule for both classical and quantum algorithms. To obtain acceptable statistics, the study should be carried out on several data sets and different system sizes, where the time required to optimize the hyperparameters above grows quickly with the system size. In nonconvex problems (in parameter space), even if the samples used at each iteration are of high quality, the learning algorithm can find suboptimal solutions. In convex problems like the one we considered here, the performance of the learning algorithm mostly relies on the quality of the samples. This makes our approach appealing for the purpose of benchmarking. Still, it is required to assess the quality of the whole distribution of states and not just the ground state as in combinatorial optimization applications. In this work, we focus on providing a proof-of-principle demonstration and experimental evidence that quantum annealers can be used for complex machine-learning tasks, such as in the case of unsupervised generative modeling on fully visible, probabilistic, graphical models with arbitrary pairwise connectivity. We hope this work continues opening new opportunities for quantum annealing and, more broadly, for quantum machine-learning research.

ACKNOWLEDGMENTS

APPENDIX: EXAMPLE

Figure 8 shows the embedding of a fully connected graph with 46 logical units into 940 physical qubits.
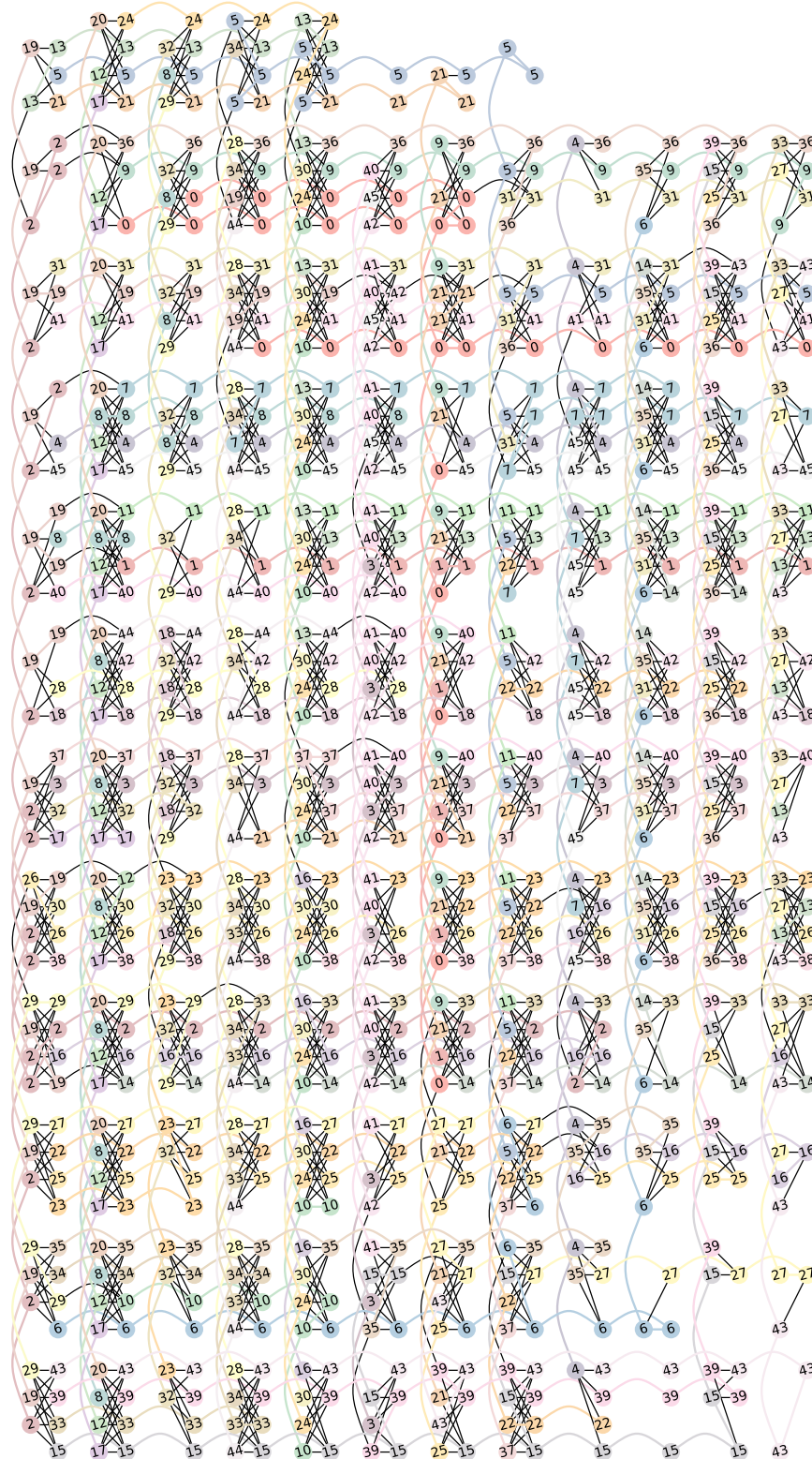

FIG. 8.   *Embedding*. We show 46 logical variables embedded into DW2X's chimera graph using 940 physical variables. Qubits belonging to a logical variable are identified by the same number and linked by edges of the same color. This embedding uses 86% of DW2X's qubits.

[1] Y. LeCun, Y. Bengio, and G. Hinton, *Deep Learning*, Nature (London) **521**, 436 (2015).

[2] Z. Ghahramani, *Probabilistic Machine Learning and Artificial Intelligence*, Nature (London) **521**, 452 (2015).

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, 2016).

[4] I. Goodfellow, arXiv:1701.00160.

[5] A. Y. Ng and M. I. Jordan, *On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes*, in *Advances in Neural Information Processing Systems*, edited by T. G. Dietterich, S. Becker, and Z. Ghahramani (MIT Press, Cambridge, 2001), pp. 841–848.

[6] R. Salakhutdinov, *Learning Deep Generative Models*, Annu. Rev. Stat. Appl. **2**, 361 (2015).

[7] A. Sinclair and M. Jerrum, *Approximate Counting, Uniform Generation and Rapidly Mixing Markov Chains*, Inf. Comput. **82**, 93 (1989).

[8] A. Frigessi, F. Martinelli, and J. Stander, *Computational Complexity of Markov Chain Monte Carlo Methods for Finite Markov Random Fields*, Biometrika **84**, 1 (1997).

[9] H. Neven, V. S. Denchev, M. Drew-Brook, J. Zhang, W. G. Macready, and G. Rose, *Binary Classification Using Hardware Implementation of Quantum Annealing*, in *Demonstrations at NIPS-09, 24th Annual Conference on Neural Information Processing Systems* (2009), pp. 1–17.

[10] Z. Bian, F. Chudak, W. G. Macready, and G. Rose, *D-Wave Systems Technical Report*, 2010 (unpublished).

[11] M. Denil and N. De Freitas, *Toward the Implementation of a Quantum RBM, NIPS Deep Learning and Unsupervised Feature Learning Workshop* (2011).

[12] N. Wiebe, D. Braun, and S. Lloyd, *Quantum Algorithm for Data Fitting*, Phys. Rev. Lett. **109**, 050505 (2012).

[13] K. L. Pudenz and D. A. Lidar, *Quantum Adiabatic Machine Learning*, Quantum Inf. Process. **12**, 2027 (2013).

[14] S. Lloyd, M. Mohseni, and P. Rebentrost, *Quantum Algorithms for Supervised and Unsupervised Machine Learning*, arXiv:1307.0411.

[15] P. Rebentrost, M. Mohseni, and S. Lloyd, *Quantum Support Vector Machine for Big Data Classification*, Phys. Rev. Lett. **113**, 130503 (2014).

[16] G. Wang, *Quantum Algorithm for Linear Regression*, Phys. Rev. A **96**, 012335 (2017).

[17] Z. Zhao, J. K. Fitzsimons, and J. F. Fitzsimons, *Quantum Assisted Gaussian Process Regression*, arXiv:1512.03929.

[18] S. Lloyd, M. Mohseni, and P. Rebentrost, *Quantum Principal Component Analysis*, Nat. Phys. **10**, 631 (2014).

[19] M. Schuld, I. Sinayskiy, and F. Petruccione, *Prediction by Linear Regression on a Quantum Computer*, Phys. Rev. A **94**, 022342 (2016).

[20] K. M. Svore, N. Wiebe, and A. Kapoor, *Quantum Deep Learning*, arXiv:1412.3489.

[21] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz, *Estimation of Effective Temperatures in Quantum Annealers for Sampling Applications: A Case Study with Possible Applications in Deep Learning*, Phys. Rev. A **94**, 022308 (2016).

[22] S. Aaronson, *Read the Fine Print*, Nat. Phys. **11**, 291 (2015).

[23] S. H. Adachi and M. P. Henderson, *Application of Quantum Annealing to Training of Deep Neural Networks*, arXiv:1510.06356.

[24] N. Chancellor, S. Szoke, W. Vinci, G. Aeppli, and P. A. Warburton, *Maximum-Entropy Inference with a Programmable Annealer*, Sci. Rep. **6**, 22318 (2016).

[25] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, *Quantum Boltzmann Machine*, arXiv:1601.02036.

[26] M. Kieferova and N. Wiebe, *Tomography and Generative Data Modeling via Quantum Boltzmann Training*, arXiv:1612.05204.

[27] I. Kerenidis and A. Prakash, *Quantum Recommendation Systems*, arXiv:1603.08675.

[28] L. Lamata, *Basic Protocols in Quantum Reinforcement Learning with Superconducting Circuits*, Sci. Rep. **7**, 1609 (2017).

[29] U. Alvarez-Rodriguez, L. Lamata, P. Escandell-Montero, J. D. Martín-Guerrero, and E. Solano, *Quantum Machine Learning without Measurements*, arXiv:1612.05535.

[30] P. Wittek and C. Gogolin, *Quantum Enhanced Inference in Markov Logic Networks*, Sci. Rep. **7**, 45672 (2017).

[31] T. E. Potok, C. Schuman, S. R. Young, R. M. Patton, F. Spedalieri, J. Liu, K.-T. Yao, G. Rose, and G. Chakma, *A Study of Complex Deep Learning Networks on High Performance, Neuromorphic, and Quantum Computers*, arXiv:1703.05364.

[32] M. Schuld, I. Sinayskiy, and F. Petruccione, *An Introduction to Quantum Machine Learning*, Contemp. Phys. **56**, 172 (2015).

[33] J. Romero, J. P. Olson, and A. Aspuru-Guzik, *Quantum Autoencoders for Efficient Compression of Quantum Data*, Quantum Sci. Technol. **2**, 045001 (2017).

[34] J. Adcock, E. Allen, M. Day, S. Frick, J. Hinchliff, M. Johnson, S. Morley-Short, S. Pallister, A. Price, and S. Stanisic, *Advances in Quantum Machine Learning*, arXiv:1512.02900.

[35] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, *Quantum Machine Learning*, Nature (London) **549**, 195 (2017).

[36] C. Ciliberto, M. Herbster, A. Davide Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, *Quantum Machine Learning: A Classical Perspective*, arXiv:1707.08561.

[37] A. Perdomo-Ortiz, M. Benedetti, J. Realpe-Gómez, and R. Biswas, *Opportunities and Challenges for Quantum-Assisted Machine Learning in Near-Term Quantum Computers*, arXiv:1708.09757.

[38] M. Benedetti, J. Realpe-Gómez, and A. Perdomo-Ortiz, *Quantum-Assisted Helmholtz Machines: A Quantum-Classical Deep Learning Framework for Industrial Datasets in Near-Term Devices*, arXiv:1708.09784.

[39] M. Gu, K. Wiesner, E. Rieper, and V. Vedral, *Quantum Mechanics Can Reduce the Complexity of Classical Models*, Nat. Commun. **3**, 762 (2012).

[40] A. B. Finnila, M. A. Gomez, C. Sebenik, C. Stenson, and J. D. Doll, *Quantum Annealing: A New Method for Minimizing Multidimensional Functions*, Chem. Phys. Lett. **219**, 343 (1994).

[41] T. Kadowaki and H. Nishimori, *Quantum Annealing in the Transverse Ising Model*, Phys. Rev. E **58**, 5355 (1998).

[42] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, *A Quantum Adiabatic Evolution Algorithm*

Applied to Random Instances of an NP-Complete Problem, Science **292**, 472 (2001).

[43] F. Gaitan and L. Clark, *Ramsey Numbers and Adiabatic Quantum Computing*, Phys. Rev. Lett. **108**, 010501 (2012).

[44] A. Perdomo-Ortiz, N. Dickson, M. Drew-Brook, G. Rose, and A. Aspuru-Guzik, *Finding Low-Energy Conformations of Lattice Protein Models by Quantum Annealing*, Sci. Rep. **2**, 571 (2012).

[45] Z. Bian, F. Chudak, R. Israel, B. Lackey, W. G. Macready, and A. Roy, *Discrete Optimization Using Quantum Annealing on Sparse Ising Models*, Front. Phys. **2**, 56 (2014).

[46] B. O'Gorman, R. Babbush, A. Perdomo-Ortiz, A. Aspuru-Guzik, and V. Smelyanskiy, *Bayesian Network Structure Learning Using Quantum Annealing*, Eur. Phys. J. Spec. Top. **224**, 163 (2015).

[47] E. G. Rieffel, D. Venturelli, B. O'Gorman, M. B. Do, E. M. Prystay, and V. N. Smelyanskiy, *A Case Study in Programming a Quantum Annealer for Hard Operational Planning Problems*, Quantum Inf. Process. **14**, 1 (2015).

[48] A. Perdomo-Ortiz, J. Fluegemann, S. Narasimhan, R. Biswas, and V. N. Smelyanskiy, *A Quantum Annealing Approach for Fault Detection and Diagnosis of Graph-Based Systems*, Eur. Phys. J. Spec. Top. **224**, 131 (2015).

[49] A. Perdomo-Ortiz, J. Fluegemann, R. Biswas, and V. N. Smelyanskiy, *A Performance Estimator for Quantum Annealers: Gauge Selection and Parameter Setting*, arXiv:1503.01083.

[50] D. Venturelli, D. J. J. Marchand, and G. Rojo, *Quantum Annealing Implementation of Job-Shop Scheduling*, arXiv:1506.08479.

[51] A. Perdomo-Ortiz, A. Feldman, A. Ozaeta, S. V. Isakov, Z. Zhu, B. O'Gorman, H. G. Katzgraber, A. Diedrich, H. Neven, J. de Kleer, B. Lackey, and R. Biswas, *On the Readiness of Quantum Optimization Machines for Industrial Applications*, arXiv:1708.09780.

[52] J. Raymond, S. Yarkoni, and E. Andriyash, *Global Warming: Temperature Estimation in Annealers*, Frontiers ICT **3**, 23 (2016).

[53] M. H. Amin, *Searching for Quantum Speedup in Quasi-static Quantum Annealers*, Phys. Rev. A **92**, 052323 (2015).

[54] M. Benedetti, Master's thesis, Université Lumière Lyon 2, France (2015).

[55] V. Dumoulin, I. J. Goodfellow, A. C. Courville, and Y. Bengio, *On the Challenges of Physical Implementations of RBMs*, in *Proceedings of the 28th AAAI Conference on Artificial Intelligence, 2014, Québec City, Québec, Canada* (2014), pp. 1199–1205.

[56] E. Schneidman, M. J. Berry, R. Segev, and W. Bialek, *Weak Pairwise Correlations Imply Strongly Correlated Network States in a Neural Population*, Nature (London) **440**, 1007 (2006).

[57] F. Ricci-Tersenghi, *The Bethe Approximation for Solving the Inverse Ising Problem: A Comparison with Other Inference Methods*, J. Stat. Phys. (2012) P08015.

[58] M. Lichman, UCI Machine Learning Repository (2013).

[59] D. J. C. MacKay, *Information Theory, Inference & Learning Algorithms* (Cambridge University Press, Cambridge, 2003).

[60] M. Mezard, G. Parisi, and M. A. Virasoro, *Spin Glass Theory and Beyond*, Lecture Notes in Physics Series (World Scientific, Singapore, 1987).

[61] M. Mezard and A. Montanari, *Information, Physics, and Computation* (Oxford University Press, New York, NY, 2009).

[62] H. Nishimori, *Statistical Physics of Spin Glasses and Information Processing: An Introduction*, International Series of Monographs on Physics (Oxford University Press, New York, 2001).

[63] A. Perdomo-Ortiz, B. O'Gorman, J. Fluegemann, R. Biswas, and V. N. Smelyanskiy, *Determination and Correction of Persistent Biases in Quantum Annealers*, Sci. Rep. **6**, 18628 (2016).

[64] S. Mandrà, Z. Zhu, and H. G. Katzgraber, *Exponentially Biased Ground-State Sampling of Quantum Annealing Machines with Transverse-Field Driving Hamiltonians*, Phys. Rev. Lett. **118**, 070502 (2017).

[65] 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.

[66] W. Lechner, P. Hauke, and P. Zoller, *A Quantum Annealing Architecture with All-to-All Connectivity from Local Interactions*, Sci. Adv. **1**, e1500838 (2015).

[67] J. Realpe-Gómez, *Quantum as Self-Reference*, arXiv:1705.04307.

[68] V. Choi, *Minor-Embedding in Adiabatic Quantum Computation: I. The Parameter Setting Problem*, Quantum Inf. Process. **7**, 193 (2008).

[69] D. Venturelli, S. Mandrà, S. Knysh, B. O'Gorman, R. Biswas, and V. Smelyanskiy, *Quantum Optimization of Fully Connected Spin Glasses*, Phys. Rev. X **5**, 031040 (2015).

[70] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, *A Learning Algorithm for Boltzmann Machines*, Cogn. Sci. **9**, 147 (1985).

[71] G. E. Hinton, *A Practical Guide to Training Restricted Boltzmann Machines*, in *Neural Networks: Tricks of the Trade*, 2nd ed., Lecture Notes in Computer Science, edited by G. Montavon, G. B. Orr, and K.-R. Miller (Springer, New York, 2012), Vol. 7700, pp. 599–619.

[72] E. T. Jaynes, *Information Theory and Statistical Mechanics. II*, Phys. Rev. **108**, 171 (1957).

[73] E. T. Jaynes, *Information Theory and Statistical Mechanics*, Phys. Rev. **106**, 620 (1957).

[74] E. T. Jaynes and G. L. Bretthorst, *Probability Theory: The Logic of Science* (Cambridge University Press, Cambridge, England, 2003).

[75] J. Cai, W. G. Macready, and A. Roy, *A Practical Heuristic for Finding Graph Minors*, arXiv:1406.2741.

[76] J. C. Spall, *Introduction to Stochastic Search and Optimization*, 1st ed. (John Wiley & Sons, New York, NY, 2003).

[77] M. Mézard and T. Mora, *Constraint Satisfaction Problems and Neural Networks: A Statistical Physics Perspective*, J. Physiol. (Paris) **103**, 107 (2009).

[78] V. Choi, *Minor-Embedding in Adiabatic Quantum Computation: II. Minor-Universal Graph Design*, Quantum Inf. Process. **10**, 343 (2011).

[79] Blocks (a)–(d) show machine-generated pictures, while blocks (e)-(h) show human-generated pictures. We have not performed the standard Turing test, where each pair of figures is shown in isolation. Ours is, in principle, a harder test for the machine, as the redundancy of having all human- and machine-generated images together enhances the probability of a human to spot differences between the two types of images. This is compensated by the low resolution of the images, which might hint at an easier test for the machine, if shown one by one, given the distortion of the images.

[80] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by Simulated Annealing*, Science **220**, 671 (1983).

[81] I. Mastromatteo and M. Marsili, *On the Criticality of Inferred Models*, J. Stat. Mech. (2011) P10012.

[82] C. Klymko, B. D. Sullivan, and T. S. Humble, *Adiabatic Quantum Programming: Minor Embedding With Hard Faults*, arXiv:1210.8395.

[83] J. Job and D. Lidar, *Test-Driving 1000 Qubits*, arXiv:1706.07124.

[84] H. G. Katzgraber, *Viewing Vanilla Quantum Annealing through Spin Glasses*, arXiv:1708.08885.

[85] T. F. Rønnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, and M. Troyer, *Defining and Detecting Quantum Speedup*, Science **345**, 420 (2014).