scientific reports

Check for updates

OPEN

Generative and discriminative training of Boltzmann machine through quantum annealing

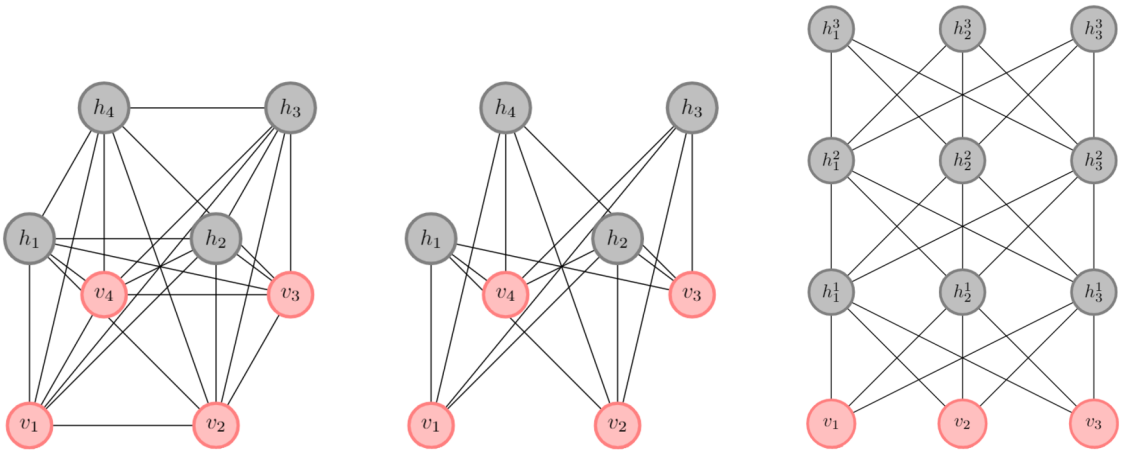
Siddhartha Srivastava[1]✉ & Veera Sundararaghavan[2]

A hybrid quantum-classical method for learning Boltzmann machines (BM) for a generative and discriminative task is presented. BM are undirected graphs with a network of visible and hidden nodes where the former is used as the reading site. In contrast, the latter is used to manipulate visible states' probability. In Generative BM, the samples of visible data imitate the probability distribution of a given data set. In contrast, the visible sites of discriminative BM are treated as Input/Output (I/O) reading sites where the conditional probability of output state is optimized for a given set of input states. The cost function for learning BM is defined as a weighted sum of Kullback-Leibler (KL) divergence and Negative conditional Log-likelihood (NCLL), adjusted using a hyper-parameter. Here, the KL Divergence is the cost for generative learning, and NCLL is the cost for discriminative learning. A Stochastic Newton-Raphson optimization scheme is presented. The gradients and the Hessians are approximated using direct samples of BM obtained through quantum annealing. Quantum annealers are hardware representing the physics of the Ising model that operates on low but finite temperatures. This temperature affects the probability distribution of the BM; however, its value is unknown. Previous efforts have focused on estimating this unknown temperature through regression of theoretical Boltzmann energies of sampled states with the probability of states sampled by the actual hardware. These approaches assume that the control parameter change does not affect the system temperature; however, this is usually untrue. Instead of using energies, the probability distribution of samples is employed to estimate the optimal parameter set, ensuring that the optimal set can be obtained from a single set of samples. The KL divergence and NCLL are optimized for the system temperature, and the result is used to rescale the control parameter set. The performance of this approach, as tested against the theoretically expected distributions, shows promising results for Boltzmann training on quantum annealers.

Boltzmann machine (BM) is an energy-based model defined on an undirected graph and is used for unsupervised learning. The graph vertices are segregated into a set of visible and hidden nodes. The probability of each state is dependent on the total energy of the graph for that state. Moreover, only the state of a visible node is "visible" to the user. Therefore, these visible states' marginalized probabilities are a non-linear function of the energy parameters and can be used to model complicated distributions. These BMs can be trained either using Maximum-likelihood (ML) learning or Contrastive Divergence (CD) learning techniques. It is well known that ML learning of Markov random fields (MRF) is a challenging task due to the large state space. Due to this complexity, Markov Chain Monte Carlo (MCMC) methods typically take a long time to converge on unbiased estimates. CD learning, on the other hand, provides a computationally inexpensive way of training MRFs. However, in general, it provides biased estimates[1].

A subclass of BM (see Fig. 1a) called the Restricted Boltzmann Machine (RBM) (see Fig. 1b) was proposed by Hinton (2002)[2] where the hidden and visible nodes had a bipartite structure. This structure allows an independent update of visible states, conditioned on the hidden states' knowledge and vice-versa. This property makes training of RBM very efficient on a classical computer. Boltzmann machines have received much attention as building blocks of multi-layer learning architectures for speech and image recognition[3,4]. The idea is that features from one RBM can serve as input to another RBM. By stacking RBMs in this way, one can construct the architecture of a Deep Boltzmann machine (see Fig. 1c). It is known that approximate inference in deep Boltzmann machines can handle uncertainty better and deal with ambiguous data[5].

[1]Department of Mechanical Engineering, University of Michigan, Ann Arbor, USA. [2]Department of Aerospace Engineering, University of Michigan, Ann Arbor, USA. ✉email: sidsriva@umich.edu



(a) General Boltzmann machine (b) Restricted Boltzmann machine (c) Deep Boltzmann machine

Figure 1. Nomenclature of Boltzmann machines from[5].

A comparison between the ML and the CD-based training of RBM is presented in[1]. The authors suggested that initial CD-based training and a final ML-based fine-tuning of RBM is the most computationally efficient way of training RBMs with less bias. This bias issue was further studied in[6], where the Persistent Contrastive Divergence (PCD) was developed. In this approach, the Markov Chain is not reset between parameter updates. This step brings the approximate gradient closer to the exact estimate in the limit of a small learning step. This method performs better on the testing data than the classical approach; however, it suffers from slow learning rates. A relatively faster approach was provided in[7] using the Fast Persistent Contrastive Divergence (FPCD) method. A tutorial on different training strategies is given in[8].

It is intuitive to see that General BM has more representative power than RBM and its derivatives. However, the efficiency of the above-mentioned training methods is not expected to translate to the general case as the data-dependent expectations are not easy to compute, at least using classical techniques. Quantum annealers (QA) have provided a promising way forward to tackle this problem of expectation estimation[9]. QA are physical devices that operate on quantum mechanical laws and are designed to minimize the Ising model's energy[10]. These devices are physical realizations of adiabatic computation, and their robustness and fault tolerance with respect to the different parameters, such as the annealing schedule[11]. These devices operate on finite temperatures, therefore, the simulations on QA result in sampling from the Boltzmann distribution of the Ising energies[12]. Researchers have recently employed this property of QA to train BMs with a slightly more complicated structure than RBMs. For instance,[13] trained a Limited Boltzmann machine (LBM) to recover missing data in the images of chemical vapor deposition (CVD) growth for a MoS_2 monolayer. LBM allows sparse connectivity among the hidden units, and due to this complexity, it is not easy to deal with in a classical setting.

Another direction that researchers have taken is the training of specialized RBMs that can be better represented on the QA architecture, e.g., the chimera RBM which emulates the topology of DWave quantum annealers[14]. This allows the model expectations to be estimated as a single sampling step instead of the k-step CD method. Meanwhile, the data-dependent expectations are estimated as the 1-step CD method due to the RBM's favorable topology. The result of this progress can be seen in the outburst of new applications of RBM in modern machine learning architectures, for instance, sampling latent space in Quantum variational autoencoders (QVAE)[15], RBM as an associative memory between generative and the discriminative part of the Associative Adversarial Network Model (AAN)[16,17] and Hybrid-Autoencoder-RBM approach to learn reduced dimension space[18].

In this paper, an ML-based approach is studied for a General BM. As discussed earlier, the topology of a highly connected graph is not conducive to CD-based approaches. The major hurdle of generating independent samples of BM is circumvented using QA. At present, the two popular QA devices are the "DWave 2000Q" system with approximately 2000 qubits connected in a Chimera topology, and the "DWave Advantage" system with approximately 5000 qubits connected in a Pegasus topology. In a recent review article[19],the authors have benchmarked the efficiency of this hardware with standard Quantum Monte Carlo approaches. Considering the physical devices' sparsity, the largest complete graph that can be simulated on these systems has a size of 64 on the 2000Q and 128 on the Advantage system. The past growth in these systems' computational power suggests the prospect of solving a large-scale problem in the near future. Taking the prospect for granted, large and arbitrarily connected BM can benefit from unbiased estimation via QA. The method developed in this work does not use the graph's topology and is numerically sub-optimal for the cases when such structures are present (e.g., a bipartite graph). For such cases, the readers are encouraged to pursue the literature listed above and the bibliography therein.

This paper aims to demonstrate the use of quantum annealers for discriminative and generative tasks involving Boltzmann machines. Generative tasks involve sampling a state from a probability distribution. At the same time, a discriminative BM acts as a classifier for a given dataset. A BM trained for generative and discriminative purposes can be used to sample a labeled dataset from a probability distribution. For example,[20] developed a

generative BM for sampling vertical and horizontal stripe patterned images. The theoretical aspects of generative and discriminative training in the context of RBM can be found in[21]. The second focus of this work is to analyze the effect of annealing temperature on training. The Boltzmann distribution is dependent on a sampling temperature, and the sampling temperature in QA is shown to be instance-dependent[14]. An overview of a few temperature-estimation methods for QA is provided in[22]. In a training strategy proposed by[23], the problem of temperature estimation is circumvented by assuming certain restrictions on the model and data distributions. However, their approach is machine-specific in the sense that the knowledge of annealing temperature is required to use the trained model on a new QA machine. A temperature estimation strategy similar to the one presented in[24] is adopted here. However, the proposed approach works on the probability distribution of samples instead of the energies. This ensures that the optimal set can be obtained from a single run. A new technique to identify control parameters with better performance in training cost is also proposed, where the KL divergence and NCLL are optimized with respect to the inverse system temperature. This method is employed to estimate the behavior of generative and discriminative costs and further refine the Ising model parameters.

Notations and mathematical prerequisites

A Boltzmann Machine is a probabilistic graphical model defined on a complete graph which is partitioned into "visible" nodes taking up values observed during training denoted by the vector, $\boldsymbol{v}$, and "hidden" nodes where values must be inferred taking up values denoted by the vector, $\boldsymbol{h}$. These states collectively define the energy and, consequently, the probability of each state. Next, the definition of a graph is stated to introduce useful terminology and notations.

Graph. A graph, G, is a pair of sets $(\mathcal{V}, \mathcal{C})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{C}$ is the set of edges/connections. For each element $e \in \mathcal{C}$ there is a corresponding ordered pair (x, y); $x, y \in \mathcal{V}$ i.e. $\mathcal{C} \subseteq V \times V$. A Graph, $G = (\mathcal{V}, \mathcal{C})$ is undirected if an edge does not have any directionality i.e. $(x, y) \equiv (y, x)$. A graph is simple if $(x, x) \notin \mathcal{C}$ for all $x \in \mathcal{V}$. The number of vertices is denoted by $N_V = |\mathcal{V}|$, and the number of edges is denoted by $N_C = |\mathcal{C}|$. The indices of connections and vertices are related using the maps, π_1 and π_2 such that for a connection with index, $k \in \{1, .., N_C\}$, the index of the corresponding vertices are $\pi_1(k)$ and $\pi_2(k)$ with $1 \leq \pi_1(k) < \pi_2(k) \leq N_V$. This essentially means $e_k \equiv (v_{\pi_1(k)}, v_{\pi_2(k)})$.

This work additionally requires the graph to be finite, i.e., $N_V < \infty$. Next, the definition of Ising energy is introduced.

Ising model. Ising model is a type of discrete pairwise energy model[25] on an undirected simple graph, $G(\mathcal{V}, \mathcal{C})$. Each vertex, $V_i \in V$ is assigned a state $s_i \in \{0, 1\}$ for all $i \in 1, \ldots, N_V$. This determines the complete state of the graph as an ordered tuple $\boldsymbol{S} = (s_1, \ldots, s_i, \ldots, s_{N_V}) \in \{0, 1\}^{N_V}$. The set of all possible states is referred to as $\mathcal{S} = \{0, 1\}^{N_V}$ with the total number of states denoted by $N_{TS} = |\mathcal{S}| = 2^{N_V}$. The Ising energy, E, for a particular state, $\boldsymbol{S}$ can be evaluated as follows:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_V} H_i s_i + \sum_{k=1}^{N_C} J_k s_{\pi_1(k)} s_{\pi_2(k)} \tag{1}$$

where the first term represents the energy of labeling a vertex with label s_i, and the second term is the energy of labeling two connected vertices as s_i and s_j. The parameters H_i and J_k are referred to as the field strength and interaction strength, respectively.

The parameter set is represented as a vector, $\boldsymbol{\theta} = \left[\theta_1, \ldots, \theta_{N_v+N_C}\right]^T$. In this work, it is specialized in the following form:

$$\boldsymbol{\theta} = \left[H_1, \ldots, H_{N_V}, J_1, \ldots, J_{N_C}\right]^T$$

The trainable parameters are denotes as $N_{\boldsymbol{\theta}} = |\boldsymbol{\theta}|$, noting that the set of parameters may be a subset of the above-mentioned set of field and interaction parameters. This notation allows describing energy as a matrix-product evaluated as $E(\boldsymbol{S}|\boldsymbol{\theta}) = \boldsymbol{\varepsilon}(\boldsymbol{S})\boldsymbol{\theta}$ where $\boldsymbol{\varepsilon}(\boldsymbol{S})$

$$\boldsymbol{\varepsilon}(\boldsymbol{S}) = \left[s_1, \ldots, s_{N_V}, s_{\pi_1(1)} s_{\pi_2(1)}, \ldots, s_{\pi_1(N_C)} s_{\pi_2(N_C)}\right]$$

The distribution of equilibrated states can be modeled, at least approximately, as a Boltzmann distribution:

$$p(\boldsymbol{S}; \boldsymbol{\theta}, \beta) = \frac{1}{Z} e^{-E(\boldsymbol{S}|\boldsymbol{\theta})/k_B T} \equiv \frac{1}{Z} e^{-\beta E(\boldsymbol{S})} \tag{2}$$

Here, Z denotes the partition function and is estimated as $Z = \sum_{\boldsymbol{S}} e^{-\beta E(\boldsymbol{S})}$.

Generative Boltzmann machines. The key idea behind a Boltzmann machine is the segregation of the vertices into visible and hidden states. This allows writing any state $\boldsymbol{S}$ of the graph as the following concatenation:

$$\boldsymbol{S} = [\boldsymbol{v}, \boldsymbol{h}]$$

where $\boldsymbol{v}$ denotes the state of the visible nodes and $\boldsymbol{h}$ denotes the states of the hidden nodes. Only visible states are observed by the user and their probability can be estimated by marginalizing over all hidden states. Therefore, the probability of a particular visible state, $\boldsymbol{v}$, is given as,

$$p(\boldsymbol{v}) = \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h}) = \frac{1}{Z} \sum_{\boldsymbol{h}} e^{-\beta E(\boldsymbol{v}, \boldsymbol{h})} \tag{3}$$

This marginalization allows the BM to represent complex probability distributions. Consider a data set of N_{DS} visible states, $\mathcal{D} = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_{N_{DS}}\}$. Each data state occurs with a probability $q(\boldsymbol{v}_k)$ for all $k \in \{1, ..., N_{DS}\}$, referred to as the true probability of the distribution. The performance of a BM can be judged by comparing the model distribution, $p(\boldsymbol{v})$ with the true distribution. This comparison can be carried out using the Kullback-Leibler divergence $D_{KL}(q||p)$ defined as,

$$D_{KL}(q||p; \boldsymbol{\theta}, \beta) = - \sum_{\boldsymbol{v} \in \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_{N_{DS}}\}} q(\boldsymbol{v}) \ln \frac{p(\boldsymbol{v}; \boldsymbol{\theta}, \beta)}{q(\boldsymbol{v})}$$

The KL divergence is always non-negative with $D_{KL}(q||p) = 0$ if and only if $q = p$ almost everywhere. For this property, D_{KL} is chosen to be the cost function for training generative BMs.

Remark: When there is no meaningful notion of the probability distribution of the data states, the true probability distribution can be substituted as $q(\boldsymbol{v}_k) = 1/N_{DS}$ for all $k \in \{1, \cdots, N_{DS}\}$. In this case, the KL Divergence is equal to the Log-likelihood of the data set normalized with the cardinality of the data set, N_{DS}.

Discriminative Boltzmann machines. It is often desired to generate a labeled data set which entails assigning a classification to each visible data point. This classifier can be included in our notation by further segregating the visible state into input-output pairs. Consequently, the state of the BM is represented as:

$$\boldsymbol{S} = [\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h}]$$

where, $\boldsymbol{v}^I$ and $\boldsymbol{v}^O$ denotes the "input" and "output" visible state. The state, $\boldsymbol{v}^O$ is used to encode the classification of state $\boldsymbol{v}^I$. Discriminative BMs also referred to as conditional BMs in literature, are trained for classification using labeled data sets. The cost function in this case is taken as the Negative Conditional Log-likelihood $\mathcal{L}$ defined as,

$$\mathcal{N}(\boldsymbol{\theta}, \beta) = - \sum_{[\boldsymbol{v}^I, \boldsymbol{v}^O] \in \{\boldsymbol{v}^1, ..., \boldsymbol{v}^D\}} \ln p(\boldsymbol{v}^O | \boldsymbol{v}^I; \boldsymbol{\theta}, \beta)$$

where, the conditional probability, $p(\boldsymbol{v}^O|\boldsymbol{v}^I)$ is estimated as:

$$p(\boldsymbol{v}^O|\boldsymbol{v}^I) = \frac{p(\boldsymbol{v}^I, \boldsymbol{v}^O)}{\sum_{\widetilde{\boldsymbol{v}}^O} p(\boldsymbol{v}^I, \widetilde{\boldsymbol{v}}^O)} \equiv \frac{\sum_{\boldsymbol{h}} p(\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h})}{\sum_{\widetilde{\boldsymbol{v}}^O, \widetilde{\boldsymbol{h}}} p(\boldsymbol{v}^I, \widetilde{\boldsymbol{v}}^O, \widetilde{h})}$$

Training method

For a general purpose training strategy, the cost is set as a weighted average of KL Divergence and Negative conditional log-likelihood as described below

$$\mathrm{C} = \alpha D_{KL} + \frac{1 - \alpha}{N_{DS}} \mathcal{N}(\boldsymbol{\theta}), \qquad \alpha \in [0, 1] \tag{4}$$

where the $\alpha = 1$ signifies a generative BM while $\alpha = 0$ signifies a discriminative BM. Gradient based techniques are used to carry out the optimization procedure. The gradient is estimated as:

$$\frac{1}{\beta} \frac{\partial C}{\partial \theta_j} = -\alpha \mathbb{E}\left(\frac{\partial E}{\partial \theta_j}\right) + \sum_{\boldsymbol{v} \in \mathcal{D}} \left(\left(\alpha q(\boldsymbol{v}) + \frac{1 - \alpha}{N_{DS}} \right) \mathbb{E}\left(\frac{\partial E}{\partial \theta_j} \middle| \boldsymbol{v} \right) - \frac{1 - \alpha}{N_{DS}} \mathbb{E}\left(\frac{\partial E}{\partial \theta_j} \middle| \boldsymbol{v}^I \right) \right) \tag{5}$$

And, the hessian is estimated as:

$$\begin{aligned} \frac{1}{\beta^2} \frac{\partial^2 C}{\partial \theta_i \partial \theta_j} = \alpha \mathrm{Cov}\left(\frac{\partial E}{\partial \theta_i}, \frac{\partial E}{\partial \theta_j}\right) - \sum_{\boldsymbol{v} \in \mathcal{D}} \left(\alpha q(\boldsymbol{v}) + \frac{1 - \alpha}{N_{DS}} \right) \mathrm{Cov}\left(\frac{\partial E}{\partial \theta_i}, \frac{\partial E}{\partial \theta_j} \middle| \boldsymbol{v} \right) \\ + \sum_{\boldsymbol{v} \in \mathcal{D}} \frac{1 - \alpha}{N_{DS}} \mathrm{Cov}\left(\frac{\partial E}{\partial \theta_i}, \frac{\partial E}{\partial \theta_j} \middle| \boldsymbol{v}^I \right) \end{aligned} \tag{6}$$

The definitions of all statistical quantities are presented in Appendix A (in Supplementary information) and the derivatives of cost functions are estimated in Appendix B (in Supplementary information).

Optimization scheme. Stochastic gradient and Newton methods have been widely employed in such problems. A comparative performance of many variants of such stochastic methods are studied in[26]. In stochastic optimization methods, it has been shown that both Newton methods and gradients-based methods have a local linear convergence rate. However,[27] developed a Newton method that is independent of the condition number in contrast to the gradient-based scheme. These developments motivate the use of Hessian in the optimization process. Such schemes are very useful in problems concerning sampling from sensitive devices like quantum annealers. Analyzing the different variations of stochastic methods is out of the scope of this work. A

mini-batch momentum-based approach is adopted. This approach can be easily substituted for one of the more sophisticated ones presented in[26,27]. The following momentum-based update rule is used:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \Delta\boldsymbol{\theta}^{(t)}, \qquad \Delta\boldsymbol{\theta}^{(t)} = \eta \boldsymbol{r}^{(t)} - \lambda\boldsymbol{\theta}^{(t)} + \nu\Delta\boldsymbol{\theta}^{(t-1)} \tag{7}$$

The parameter, η defines the learning rate of the method and ν defines a momentum rate. A higher momentum rate means that the current learning direction is closer to the learning rate in the previous time step. In general, the momentum is kept low at the beginning and slowly increased as the learning progresses. This technique is employed to reduce oscillations in the final stages of training. The parameter λ modifies the cost function to minimize the magnitude of the learned parameter. In this work. this parameter is identically set to 0 in all test cases and is mentioned only for completeness. The variable, $\boldsymbol{r}$, denotes the rate of update. In the gradient-based method, it is estimated as:

$$\boldsymbol{r}^{(t)} = -\nabla_{\boldsymbol{\theta}^{(t)}} C$$

In Newton method, rate of update is estimated as:

$$\boldsymbol{r}^{(t)} = -(\nabla^2_{\boldsymbol{\theta}^{(t)}} C)^{-1} \nabla_{\boldsymbol{\theta}^{(t)}} C$$

Remark 1: The Hessian matrix estimated from the sampling process, is usually rank deficient. The main reason is under-sampling. Therefore, the inversion of these matrices poses a challenge. In this work, Tikhonov regularization is used where the singularity of $\nabla^2_{\boldsymbol{\theta}^{(t)}} C$ is alleviated by adding positive terms to the diagonal as follows where $\mathbb{I}$ is the identity matrix.

$$\nabla^2_{\boldsymbol{\theta}^{(t)}} C \rightarrow \nabla^2_{\boldsymbol{\theta}^{(t)}} C + \epsilon^2 \mathbb{I}$$

Remark 2: The above update rule works for unconstrained optimization. A constrained optimization problem can be considered by employing Lagrange multipliers. In this study, the constraints are much simpler, $|H_i| < H_0$ and $|J_k| < J_0$. These constraints represent the practical range of parameters for quantum annealers. These bounds are implemented by using the following scaling parameter:

$$\delta = \max\left\{ \frac{\max_{i\in\{1,\ldots N_V\}} |H_i|}{H_0}, \frac{\max_{k\in\{1,\ldots N_C\}} |J_k|}{J_0} \right\}$$

In any optimization step, if $\delta > 1$, then the parameters are scaled as: $\boldsymbol{\theta}^{(t)} \rightarrow \boldsymbol{\theta}^{(t)}/\delta$ and the corresponding $\Delta\boldsymbol{\theta}^{(t)}$ is updated. The optimization procedure is presented in Algorithm 1. The procedure uses a subroutine *EstimateDerivatives* (Algorithm 2) to estimate the gradients and hessian. This subroutine is discussed in the next section. For gradient-based approaches, the estimation of hessian can be tactically avoided from algorithm 2 by ignoring all covariance terms, and the hessian in this case is set to an Identity matrix.

Algorithm 1 Optimization Step (per epoch)

Input: M (Number of Batches), $\boldsymbol{\theta}^{(t)}$ (Current Parameters), $\{\eta, \lambda, \nu\}$ (Learning parameters)
Partition $\mathcal{D}$ into M partitions $\mathcal{D}_1, ..., \mathcal{D}_M$
For all $\{i \in 1, ..., M\}$, define $q_i : \mathcal{D}_i \rightarrow \mathbb{R}$ such that $q_i(\boldsymbol{v}_k) = q(\boldsymbol{v_k}) / \sum_{\boldsymbol{v}_j \in \mathcal{D}_i} q(\boldsymbol{v}_j)$ for all $\boldsymbol{v}_k \in \mathcal{D}_i$
$\boldsymbol{\theta}^{(t,0)} \leftarrow \boldsymbol{\theta}^{(t-1)}$
$\Delta\boldsymbol{\theta}^{(t,0)} \leftarrow \Delta\boldsymbol{\theta}^{(t-1)}$
for $i \leftarrow 1$ to M **do**
    BatchData $\leftarrow \{\mathcal{D}_i, q_i\}$
    $\nabla_{\boldsymbol{\theta}^{(t,i)}} C, \nabla^2_{\boldsymbol{\theta}^{(t,i)}} C \leftarrow$ *EstimateDerivatives*.BatchData ▷ Algorithm 2
    Estimate $\boldsymbol{r}^{(t,i)}$
    $\Delta\boldsymbol{\theta}^{(t,i)} \leftarrow \eta \boldsymbol{r}^{(t,i)} - \lambda\boldsymbol{\theta}^{(t,i)} + \nu\Delta\boldsymbol{\theta}^{(t,i-1)}$
    $\boldsymbol{\theta}^{(t,i+1)} \leftarrow \boldsymbol{\theta}^{(t,i)} + \Delta\boldsymbol{\theta}^{(t,i)}$
    Apply constraints
$\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t,M+1)}$
$\Delta\boldsymbol{\theta}^{(t)} \leftarrow \Delta\boldsymbol{\theta}^{(t,M+1)}$
return $\boldsymbol{\theta}^{(t+1)}$, $\Delta\boldsymbol{\theta}^{(t)}$

Numerical estimation of Gradient and Hessian. QA can be treated as a black box that samples the states for a given set of parameters. DWave Quantum annealer is used in this work that operates based on the $\{+1, -1\}$ states. The energy parameters for the $\{+1, -1\}$ states and the {1,0} states can be transformed using the

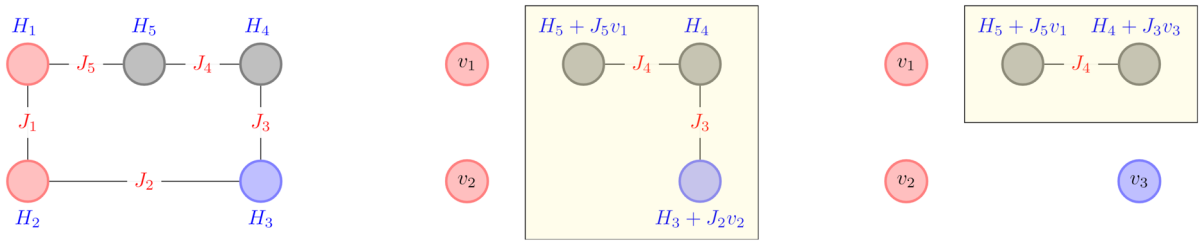
relation presented in Appendix C (in Supplementary information). Basis transformation may scale the parameters outside the allowed range of DWave. This problem can be mitigated by choosing appropriate bounds on field and interaction parameters in the optimization process. The user can specify the number of samples. The probability of a particular sample state is then estimated as the ratio of the number of occurrences of that state to the specified number of samples. It is easily noticeable that the gradients and hessian are described in terms of statistics of the energy gradient $\nabla_{\boldsymbol{\theta}} E$. The first term in the gradient and the hessian requires estimation of $\mathbb{E}(\nabla_{\boldsymbol{\theta}} E)$ and $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}} E)$. In the notation described in Sect. "Ising model", given a sample state $\boldsymbol{S}$, energy gradient is estimated as:

$$\nabla_{\boldsymbol{\theta}} E(\boldsymbol{S}) = \boldsymbol{\varepsilon}(\boldsymbol{S})$$

The latter terms in Eq(5) and Eq(6) are more complicated to compute as they require summation over each visible data. Two possible strategies can be employed in this case. In the first approach, all sampled states are grouped according to the visible/input states. The conditional probabilities are then estimated by treating each data state's respective groups as the sample set. Since the samples from this estimation is unbiased, the QA samples are independent. In theory, the accuracy of these conditional probabilities increases with sample size. However, in practice, the number of samples is finite and should be kept to a minimum possible value to reduce computational cost. This is a critical drawback of this approach. For instance, not every data state needs to appear in the samples. In such cases, the KL Divergence cannot be estimated.

The other approach is to run independent simulations for each data state on the subgraph of hidden nodes, $\mathcal{G}_H$, and the subgraph of hidden and output nodes, $\mathcal{G}_{HO}$. The energy parameters of these subgraphs depend on the visible states (for $\mathcal{G}_H$) and input states (for $\mathcal{G}_{HO}$). The field terms are augmented to include the energy for fixing the states of the removed nodes. An illustration of this procedure is presented in Fig. 2. One can observe that this process leads to a shift in energy states. For instance, same states in Fig. 2a,b have an energy difference of $H_1 v_1 + H_2 v_2 + J_1 v_1 v_2$. However, the Boltzmann distribution remains unchanged by a uniform shift in energies. The drawback of this method is that this procedure's computational cost grows with the training data size. This growth by itself is not a problem; however, the sampling step is usually the most time-consuming. In general, CD-1 steps are used to determine this term in RBMs quickly. However, this method is not extendable to General BMs. The authors are currently unaware of any scheme that circumvents this computation. This second approach of running independent simulations for each data will be adopted from here onward. The procedure for estimating gradient and hessian from the sampled data is presented in Algorithm 2. It should be noted that the estimation of RHS of Eq(5) and Eq(6) only yields the direction of gradient and hessian, respectively. The size of the update can be subsumed in the learning rates. However, the value of β influences the probability distribution and, consequently, influences the trained parameters' value. This issue is discussed in the next section.

It is also worth noting that the exact scaling of the computational complexity of these problems is very hard to ascertain and most likely not well-defined. The reason is that the computational cost of estimation of gradient and hessian scales with the sample size while the optimal sample size is problem dependent[28]. However, for any graph with N_V vertices and N_S samples, the estimation of gradient requires $\mathcal{O}(N_S N_V N_{\boldsymbol{\theta}})$ computations while the estimation of Hessian requires at most $\mathcal{O}(N_S N_V^2 N_{\boldsymbol{\theta}}^2)$ computations. Due to the limited number of vertices in QA ($\sim 10^3$), this cost is not expected to be a bottleneck in the near term. Note that this cost is only for estimation of Hessian. The Newton scheme will have an additional cost for matrix inversions ($\mathcal{O}(N_{\boldsymbol{\theta}}^3)$ based on Gauss-Jordan elimination). For any graph, $N_{\boldsymbol{\theta}} \leq N_V + N_C = (N_V + N_V^2)/2$. Therefore, the complexity of inversion operation is smaller (or equal) to the estimation of the hessian.



(a) Graph, $\mathcal{G}$ (b) Subgraph, $\mathcal{G}_{HO}$ (c) Subgraph, $\mathcal{G}_H$

Figure 2. Illustration for estimating parameters of the subgraph. The Input, Output and Hidden nodes are represented with red, blue and grey colors, respectively. The field parameters and interaction parameters are written in blue and red fonts, respectively. The subgraphs are presented in a yellow box. (**a**) Cyclic graph, $\mathcal{G}$ with 2 input nodes, 1 output node and 2 hidden nodes (**b**) Subgraph of output and hidden nodes, $\mathcal{G}_{HO}$: Fixing the visible input $\boldsymbol{v}^I = [v_1, v_2]$ results in an augmented field term on the output and hidden nodes (**c**) Subgraph of hidden nodes, $\mathcal{G}_H$: Fixing the visible data $\boldsymbol{v} = [v_1, v_2, v_3]$ results in an augmented field term on the hidden nodes.

Algorithm 2 *EstimateDerivative*: Estimation of Gradient and Hessian

$\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{DS}\}, \{q(\boldsymbol{v}_1), \ldots, q(\boldsymbol{v}_D)\} \leftarrow$ Batch Data
$\{\boldsymbol{S}\} \leftarrow$ Sample state of $\mathcal{G}$
Estimate random variable $\nabla_{\boldsymbol{\theta}} E$ from $\{\boldsymbol{S}\}$
Estimate $\mathbb{E}(\nabla_{\boldsymbol{\theta}} E)$, $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}} E)$
for $i \leftarrow 1$ to DS **do**
    $[\boldsymbol{v}^I, \boldsymbol{v}^O] \leftarrow \boldsymbol{v}_i$
    Update Parameters of $\mathcal{G}_{HO}$ and $\mathcal{G}_H$
    $\{\boldsymbol{h}\} \leftarrow$ Sample state of $\mathcal{G}_H$
    Estimate random variable $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{v})$ from $\{[\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h}]\}$
    Estimate $\mathbb{E}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v})$, $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v})$
    $\{[\widetilde{\boldsymbol{v}}^O, \widetilde{\boldsymbol{h}}]\} \leftarrow$ Sample state of $\mathcal{G}_{HO}$
    Estimate random variable $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{v}^I)$ from $\{[\boldsymbol{v}^I, \widetilde{\boldsymbol{v}}^O, \widetilde{\boldsymbol{h}}]\}$
    Estimate $\mathbb{E}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v}^I)$, $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v}^I)$
Evaluate Eq(5) and Eq(6)
end

Effect of annealing temperature. Experimental evidence has shown that the apparent annealing temperature, i.e., the temperature corresponding to the Boltzmann distribution of samples, is instance-dependent[14]. The corresponding inverse temperature is referred to as β^* in this section. The consequence of this instance-dependence is that the quantum annealing systems cannot be rated for specific temperatures, and β^* has to be estimated from the samples for each instance of the graph. The knowledge of β^* is crucial in developing models capable of being implemented on different computational devices. Even in the same machine, two different embeddings of the same logical graph may lead to different annealing temperatures and consequently show disparities in performance. The key idea behind the estimation of β^* is that the Boltzmann distribution of a state can be equivalently written as follows by taking a log on both sides:

$$\log p(\boldsymbol{S}) = -\beta E(\boldsymbol{S}) - \log Z$$

β^* is estimated as the slope of this line. The exact form is as follows where $\mathbb{E}(.)$ denotes the expectation of the variable over all possible states:

$$\beta^* = -\frac{\sum_{\boldsymbol{S}} (E(\boldsymbol{S}) - \mathbb{E}(E))\big(\ln p(\boldsymbol{S}) - \mathbb{E}\big(\ln p\big)\big)}{\sum_{\boldsymbol{S}} (E(\boldsymbol{S}) - \mathbb{E}(E))^2} \tag{8}$$

A similar approach was developed by[14] that uses binning of energies. They estimated the value of β using regression of ratio of probability of two bins of different energies that are sampled for two different control parameter sets. The statistics of different energy levels can be succinctly written as:

$$p(E;\beta) = D_g(E)\frac{e^{-\beta E}}{Z}$$

where $D_g(E)$ is the degeneracy of states with energy, E. The authors used the log of ratio of probability, $l(\beta) = p(E_1;\beta)/p(E_2;\beta)$ for some fixed energy levels. They manipulated the value of β by rescaling the parameters and were able to estimate the β from the slope of $l(\beta)$ and the scaling factor. Readers should notice that although these two approaches are based on a similar argument, they differ greatly in their application. In the former approach, the intuition is that β represents the slope of the following sampled data, $(E(\boldsymbol{S}), \log p(\boldsymbol{S}))$. Meanwhile, the latter approach uses the data of the probability distribution of energy levels. Both methods have their pros and cons. The second method requires sampling at rescaled parameters assuming that the effective annealing temperature remains invariant with rescaled parameters. This is not usually the case as shown by[22] where a nonlinear relation was estimated between the rescaling of energy and the effective β, attributed to 'non-Boltzmann' nature of probability distribution. The variation in the distribution due to small changes in parameters is overshadowed by the noise in the Boltzmann machine. Meanwhile, large parameter changes may lead to empty bins leading to an inability to compute probability ratios. The first approach that is proposed here, is equivalent to creating a bin of each unique state that is sampled, and this step is computationally more expensive than binning energy levels, especially in the limit of large graphs. On the favorable side, it requires only one set of samples as rescaling parameters is not required. Note that the probability of samples may be obtained as a direct output from current quantum annealing hardware, as is the case here in the case of the D–wave annealer making this approach fairly easy to implement compared to the energy binning method.

Examples

The gradients for the cost as used in the training algorithm remain invariant to the system temperature up to scaling as seen in Eq. (5). Assuming that distribution of states in the optimal solution can be effectively modeled as a Boltzmann distribution for some β, one can extract some useful statistics that can help further refine the model parameters for better performance (in terms of training cost).

The key idea is that the variation of cost components (D_{KL} and $\mathcal{N}$) with respect to β (close to β^*) can be approximated using the statistics of samples obtained from the optimal parameter set. This information can elucidate in an approximate sense, the β at which the BM's performance is most optimal, say β^o, for a given choice of parameters (θ^*). The user cannot control the annealing temperature but the parameters can be rescaled to achieve the overall effect. It can be seen via Eq. (2) that scaling $\beta \to c\beta^*$ and $\boldsymbol{\theta} \to c\boldsymbol{\theta^*}$ has the same effect on the probability distribution. Since the hardware temperature cannot be modified, β^o can then be used to rescale the parameters as $\boldsymbol{\theta} \to \boldsymbol{\theta} \cdot \beta^o/\beta^*$. to further reduce the cost.

Example: temperature-based scaling of training parameters. This effect is experimentally evaluated in Fig. 3, where the costs are computed for a fully connected graph with 10 vertices (7 visible and 3 hidden) trained with data representing the truth table of the 2-bit adder circuit. The net cost of optimization is estimated for $\alpha = 0.5$. Here the scaling factor of 1 represents the initial parameter θ^*. The individual cost components simultaneously decrease till the scaling factor of 3. These rescaled optimal parameters may not lie in the hardware-specified bounds, this may in fact be the reason why these parameters are not estimated by the training procedure in the first place. Next, we demonstrate a method that allows evaluating this optimal scaling with only the statistics of samples at θ^*. Unlike previous studies, this method assumes Boltzmann behavior only in the vicinity of β^* while circumventing the problem of multiple evaluations of samples at different but close parameters.

Here, the Taylor expansions of KL Divergence and the Negative Conditional Log-likelihood as a function of β are employed to compute the optimal temperature. The advantage is that all the coefficients of β in the following expression can be estimated from the sampled states at β^*. The Taylor expansion till the second term is as follows:

$$\begin{aligned} D_{KL}^{\text{app}}(\beta) &= D_{KL}^* + \left.\frac{\partial D_{KL}}{\partial \beta}\right|_{\beta^*} (\beta - \beta^*) + \frac{1}{2}\left.\frac{\partial^2 D_{KL}}{\partial \beta^2}\right|_{\beta^*} (\beta - \beta^*)^2 + \cdots \\ \mathcal{N}^{\text{app}}(\beta) &= \mathcal{N}^* + \left.\frac{\partial \mathcal{N}}{\partial \beta}\right|_{\beta^*} (\beta - \beta^*) + \frac{1}{2}\left.\frac{\partial^2 \mathcal{N}}{\partial \beta^2}\right|_{\beta^*} (\beta - \beta^*)^2 + \cdots \end{aligned} \tag{9}$$

where

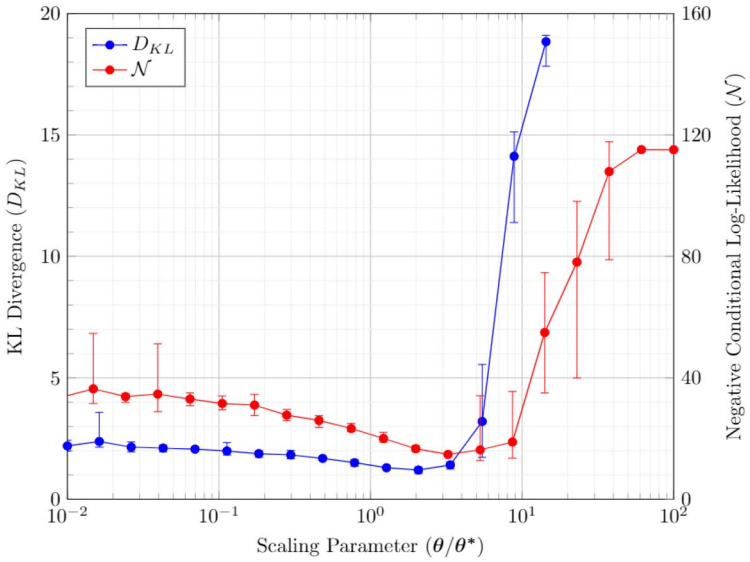


Figure 3. Cost components for different scaling parameters. The initial set of parameters ($\boldsymbol{\theta^*}$) are for a BM with 10 vertices, trained for a 2-bit added circuit. The cost corresponds to the mean cost of 20 sample runs. The errorbar represents the minimum and maximum values in the sample set for each scaling parameter.

$$\frac{\partial D_{KL}}{\partial \beta} = -\mathbb{E}_{\boldsymbol{v},\boldsymbol{h}}(E) + \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v}) \sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h}) p(\boldsymbol{h}|\boldsymbol{v})$$

$$\frac{\partial^2 D_{KL}}{\partial \beta^2} = \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v})(-\mathrm{Var}(E|\boldsymbol{v}) + \mathrm{Var}(E))$$

$$\frac{\partial \mathcal{N}}{\partial \beta} = \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \mathbb{E}(E|\boldsymbol{v}) - \mathbb{E}(E|\boldsymbol{v}^I)$$

$$\frac{\partial^2 \mathcal{N}}{\partial \beta^2} = \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} -\mathrm{Var}(E|\boldsymbol{v}) + \mathrm{Var}(E|\boldsymbol{v}^I)$$

The exact behavior is estimated by evaluating the Boltzmann distribution, Eq(2). The value of β^o is estimated as the minimizer of the approximated quadratic cost function, $C^{\mathrm{app}} = \alpha D_{KL}^{\mathrm{app}} + (1-\alpha)\mathcal{N}^{\mathrm{app}}/N_{DS}$. Therefore, when approximated cost is convex, the β^o is estimated as:

$$\beta^o = \beta^* - \left(\frac{\partial^2 C}{\partial \beta^2}\right)^{-1} \frac{\partial C}{\partial \beta}$$

Figure 4, shows the individual cost components for the 2-bit adder example with respect to the value of β. The analytical model (dashed line) is estimated by enumerating all possible states of the Ising model and estimating the Boltzmann distribution. The approximated model is estimated using Eq. (9). The estimated coefficients are: $\beta^* = 2.5251$, $\beta^o = 5.2321$, $D_{KL} = 1.4182$, $\mathcal{N} = 21.2122$, $\frac{\partial D_{KL}}{\partial \beta} = -0.1245$, $\frac{\partial^2 D_{KL}}{\partial \beta^2} = 0.0559$, $\frac{\partial \mathcal{N}}{\partial \beta} = -1.9677$, $\frac{\partial^2 \mathcal{N}}{\partial \beta^2} = 0.7170$. The figure verifies the expected behavior that both the components of the costs are simultaneously lowered at $\beta = \beta^o$. The cost components for rescaled parameters are $D_{KL} = 1.2193$, $\mathcal{N} = 16.5163$. The scaling factor for optimal parameters is approximated to be $\beta^*/\beta^o \approx 2.1$. This value is very close to the experimentally evaluated value of ~ 3. This procedure requires only single set of samples in contrast to sampling at all possible β.

Example: generative and discriminative training. As a toy problem, the data illustrated in Fig. 5 is used to train the BM models. The first data set (Fig. 5a) is unlabelled (Fig. 5b) and has 11 data points. The second data set has 40 data points with 11 labeled as '0' and 29 labeled as '1'.

In all the cases, the training parameters of Eq. (7) have a constant value of $\eta = 0.1$, $\nu = 0.7$, and $\lambda = 0$. The Hessian is inverted using Tikhonov regularization with $\epsilon = 10^{-3}$. The training data (see Fig. 6) shows that the Newton approach performs better than the gradient-based approach. The fluctuations in the deterministic training case (NumBatch = 1) are due to the DWave sampling step's stochastic nature. It should also be remarked that the parameters were not optimized for individual cases and were arbitrarily picked from a suitable range.

The stochastic training method with 2 batches was employed for the first data set (Generative learning). Two BM's with 3 hidden nodes were considered, the first with complete connectivity and the second with Restricted

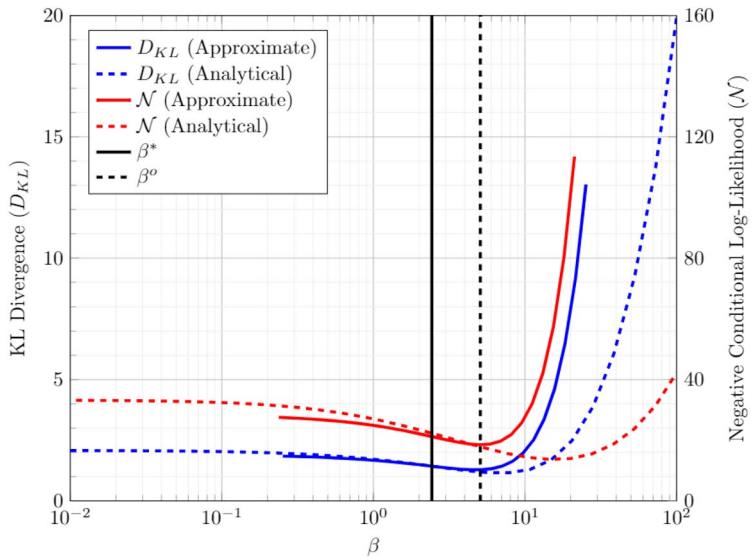
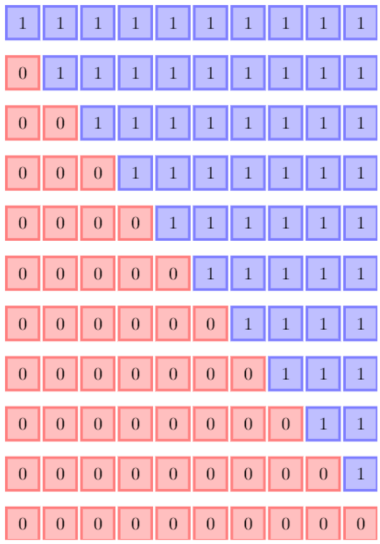
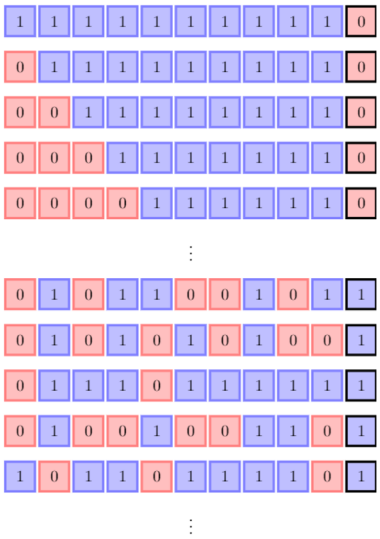


Figure 4. Comparison of approximate and exact values of the training cost of BM with 10 vertices, trained for a 2bit added circuit.



(a) Data points for generative training



(b) Data points for discriminative training

Figure 5. Visible data states, $\mathcal{D}$, for training Boltzmann machines. Each row represents a single data point. (**a**) Each data point represents the phase of a state at 10 spatial points. The '0' phase is accumulated to the left, and the '1' phase is accumulated to the right with at most 1 boundary. (**b**) Labeled data sets with the data points described in part(**a**) are labeled as '0', and data points with random spatial distribution are labeled as '1'. The label is appended at the end of each data point in a black box.

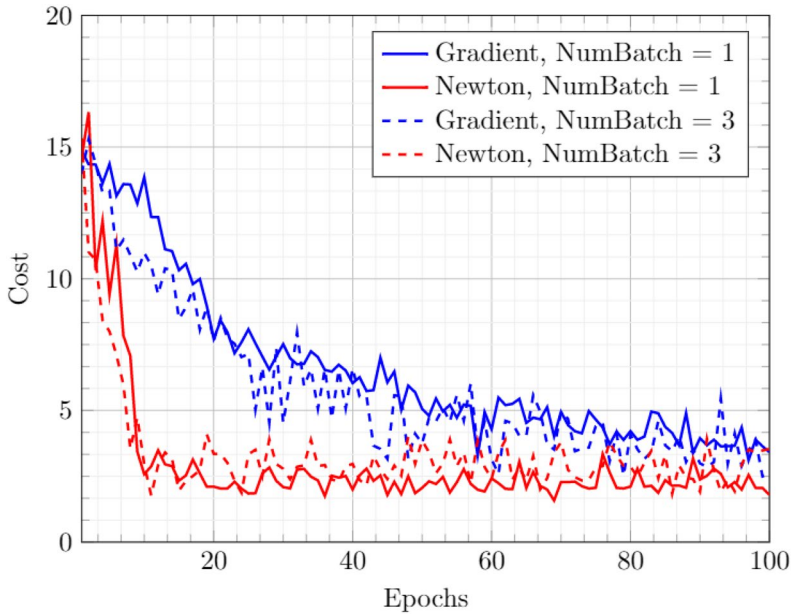


Figure 6. Training data for a General BM with 3 Hidden nodes. The cost of training is defined by Eq. (4) with $\alpha = 0.5$.

BM architecture. The variation of KL Divergence (training Cost) with the annealing temperature is shown in Fig. 7. It is observed that trained BM of the general type performs better than the Restricted type. This is an intuitive result as RBM is a specialized case of the General BM and has less representation capability in comparison.

The effect of the cost parameter, α was studied for training General BM with the second dataset. The results are presented in Fig. 8. The training is carried out with 70/30 split into training and testing data. A large reduction in KL Divergence is observed, even for a small value of α. Moreover, there was no substantial change in the conditional likelihood. This result suggests that the performance of Conditional/Discriminative BM can be enhanced by adding a small KL Divergence component to the training cost.

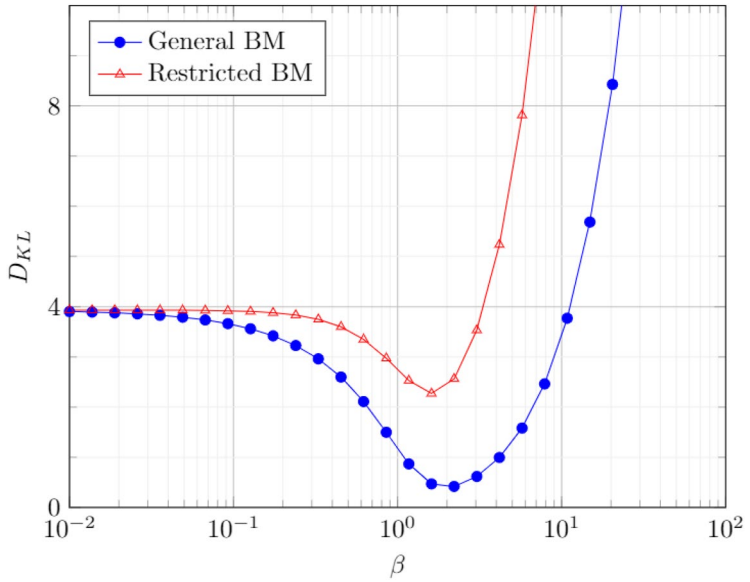


Figure 7. Trained Generative BM with 3 hidden nodes.

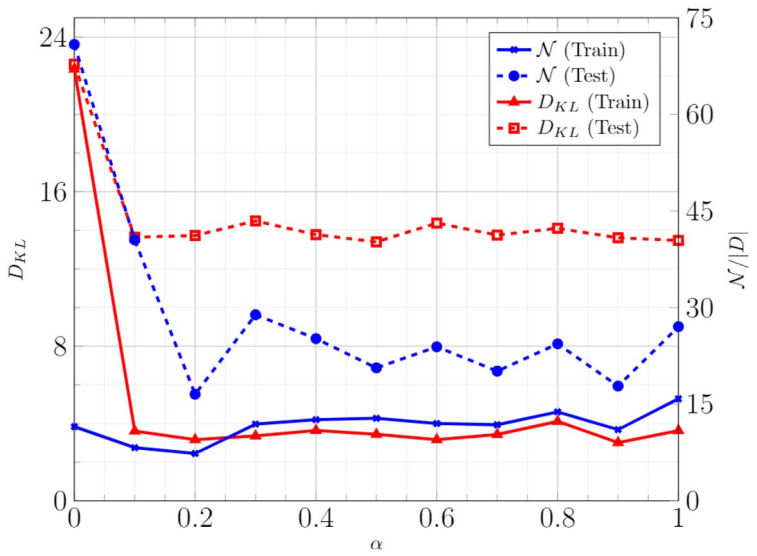


Figure 8. Performance of trained BM with 4 hidden nodes with respect to the cost parameter, α.

Conclusion

Quantum annealing has the potential to significantly improve the training of General BM. The stochastic Newton and gradient-based training methods can be employed using direct sampling from quantum states. In the proposed approach, the inclusion of Hessian in training increases the computation cost from $\mathcal{O}(N_S N_V N_{\boldsymbol{\theta}})$ to $\mathcal{O}(N_S N_V^2 N_{\boldsymbol{\theta}}^2)$. This scaling does not pose any practical limitation for current QA devices due to a limited number of qubits. This procedure can accelerate the training of a General Boltzmann machine with higher representation capability. The use of QA is promising for quantum/classical training since many qubits are available, and the training takes advantage of measurements on the physical realization of the BM[15,29]. Unlike the other popular methods like the Contrastive Divergence, this method does not utilize the suggested BM's special topology. However, in practice, having a sparse connection is desirable to embed larger graphs in the DWave architecture. These methods were employed to carry out generative and discriminative training in toy problems. The numerical results suggested that stochastic Newton optimization performs better than gradient-based optimization. It was also observed that adding a small contribution of KL Divergence in discriminative cost greatly improves BM's performance.

Typically, the annealing temperature of this hardware is experimentally calibrated and is not accessible to the user. The effective temperature also depends on the graph embedding on the physical hardware leading to a sub-optimal prescription of parameters when simulating the BM using a new embedding. A significant contribution

of this paper is the procedure to approximate the behavior of BM in these different temperatures. Our procedure is helpful in approximating a refined set of rescaled parameters for BM for a given embedding using the statistics of a single sample set of annealed states. In addition, once the hardware parameters are changed and optimized for a given generative or discriminative task using a gradient algorithm, the final parameter set can be further improved using a rescaling strategy. Here, the cost is additionally optimized with respect to the system temperature. Comparison of the results against exact analytical distributions has shown that this approach improves training. In the future, we will extend this work to Potts model[25] (generalization of Ising model with multiple spins) . This work will be tested for other practically relevant benchmark problems along with a rigorous analysis of training parameters as a function of hardware embedding.

Data availability

All codes and data are provided at https://github.com/sidsriva/GenDisBM.

Received: 11 January 2023; Accepted: 4 May 2023
Published online: 16 May 2023

References

1. Miguel, A. C.-P. and Geoffrey, E. H. On contrastive divergence learning. In *Aistats*, volume 10, pages 33–40. Citeseer, (2005).
2. Geoffrey, E. H. Training products of experts by minimizing contrastive divergence. *Neural comput.* **14**(8), 1771–1800 (2002).
3. Navdeep, J. and Geoffrey, H. Learning a better representation of speech soundwaves using restricted boltzmann machines. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5887. IEEE, 2011.
4. Eslami, S. M. A., Heess, N., Christopher, K. I. W. & John, W. The shape Boltzmann machine: A strong model of object shape. *Int. J. Comput. Vis.* **107**(2), 155–176 (2014).
5. Salakhutdinov, R. and Hinton G. Deep boltzmann machines. In *Artificial intelligence and statistics*, 448–455 (2009).
6. Tijmen, T. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, 1064–1071 (2008).
7. Tijmen T. and Geoffrey H. Using fast weights to improve persistent contrastive divergence. In *Proceedings of the 26th Annual International Conference on Machine Learning,* 1033–1040, (2009).
8. Asja Fischer and Christian Igel. An introduction to restricted boltzmann machines. In Iberoamerican Congress on Pattern Recognition, pages 14–36. Springer, 2012.
9. Steven, H. A. and Maxwell, P. H. Application of quantum annealing to training of deep neural networks. arXiv preprint arXiv:1510.06356, (2015).
10. Kadowaki, T. & Nishimori, H. Quantum annealing in the transverse ising model. *Phys. Rev. E* **58**(5), 5355 (1998).
11. Andrew, M. C., Edward, F. & John, P. Robustness of adiabatic quantum computation. *Phys. Rev. A* **65**(1), 012322 (2001).
12. Mohammad, H. A. Searching for quantum speedup in quasistatic quantum annealers. *Phys. Rev. A* **92**(5), 052323 (2015).
13. Jeremy, L. *et al.* Boltzmann machine modeling of layered mos2 synthesis on a quantum annealer. *Comput. Mater. Sci.* **173**, 109429 (2020).
14. Benedetti, M., Realpe-Gómez, J., Biswas, R. & Perdomo-Ortiz, A. Estimation of effective temperatures in quantum annealers for sampling applications: A case study with possible applications in deep learning. *Phys. Rev. A* **94**(2), 022308 (2016).
15. Amir, K. *et al.* Quantum variational autoencoder. *Quantum Sci. Technol.* **4**(1), 014001 (2018).
16. Tarik, A. and Asli, C. Associative adversarial networks. arXiv preprint arXiv:1611.06953, 2016.
17. Max, W., Thomas, V., Tad, H., and Eleanor, R. Quantum-assisted associative adversarial network: Applying quantum annealing in deep learning. arXiv preprint arXiv:1904.10573, (2019).
18. Jennifer, S., John, D., and Milton, H. A hybrid quantum enabled rbm advantage: Convolutional autoencoders for quantum image compression and generative learning. In *Quantum Information Science, Sensing, and Computation XII*, volume 11391, page 113910B. International Society for Optics and Photonics, (2020).
19. Atanu, R., Sei, S., Amit, D. & Bikas, K. C. Quantum annealing: An overview. *Philos. Trans. R. Soc. A* **381**(2241), 20210417 (2023).
20. Vivek, D., Raja, S., Muhammad, A. A., Travis, S. H., and Sabre, K. Training and classification using a restricted boltzmann machine on the d-wave 2000q. arXiv preprint arXiv:2005.03247 (2020).
21. Larochelle, H., Mandel, M., Pascanu, R. & Bengio, Y. Learning algorithms for the classification restricted boltzmann machine. *J. Mach. Learn. Res.* **13**(1), 643–669 (2012).
22. Raymond, J., Yarkoni, S. & Andriyash, E. Global warming: Temperature estimation in annealers. *Front. ICT* **3**, 23 (2016).
23. Benedetti, M., Realpe-Gómez, J., Biswas, R. & Perdomo-Ortiz, A. Quantum-assisted learning of hardware-embedded probabilistic graphical models. *Phys. Rev. X* **7**(4), 041052 (2017).
24. Dmytro, K., Yanbo, X., Zhengbing, B., Fabian, C., William, G. M., Jason, R., , and Evgeny, A. Benchmarking quantum hardware for training of fully visible boltzmann machines. arXiv preprint arXiv:1611.04528, 2016.
25. Siddhartha, S.and Veera, S. Bandgap optimization in combinatorial graphs with tailored ground states: Application in quantum annealing. *Opt Eng.* 1–19, (2022).
26. Nicolas, L. and Peter, R. Momentum and stochastic momentum for stochastic gradient, newton, proximal point and subspace descent methods. *Comput. Opt. Appl.* 1–58, (2020).
27. Dmitry, K., Konstantin, M., and Peter, R. Stochastic newton and cubic newton methods with simple local linear-quadratic rates. arXiv preprint arXiv:1912.01597, (2019).
28. Ayanzadeh, R., Halem, M. & Finin, T. Reinforcement quantum annealing: A hybrid quantum learning automata. *Sci. Rep.* **10**(1), 1–11 (2020).
29. Walter, W. *et al.* A path towards quantum advantage in training deep generative models with quantum annealers. *Mach Learn Sci Technol* **1**(4), 045028 (2020).

Author contributions

S.S. and V.S. conceptualized the work and performed the analysis of methods.S.S. developed the codes and wrote the manuscript. S.S. and V.S reviewed the manuscript

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-023-34652-4.

Correspondence and requests for materials should be addressed to S.S.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.