PH5842: Advanced topics in Quantum Computing and Quantum Information
Quantum Boltzmann Machines

Neelkanth Rawat (PH20C026)[a] and Rajarsi Pal (PH20C032) [a]

[a]Department of Physics, Indian Institute of Technology Madras, Chennai
May 6, 2022

Abstract

One of the major tasks in the looming field of Quantum machine learning is to develop quantum versions of the conventional classical models and to look for possible quantum advantage that one can leverage. In this term paper, we summarise the essential aspects of a Quantum model for Boltzmann machine proposed by Amin et al. in which the quantum nature of the processor is exploited in both the model as well as in the training process. We present results for some toy problems that we tried to solve numerically using this approach. We also present an interesting new insight of groundstate training highlighted by M.Rudolph in his work.

Keywords— Quantum Boltzmann Machines (QBM), Bound-based QBM, groundstate training

1 Introduction

One of the tasks in machine learning is to learn the probability density from the input data and generate new samples from the distribution so learned. Boltzmann machines (BM) is one of the models that is commonly used to achieve this task. It is a generative machine learning models, capable of learning the probability distribution over a multivariate binary vectors from given data. They belong to the class of energy based-unsupervised-generative models [Hin07].
A BM consists of a probabilistic network of binary units. The structure of this model is same as that of disordered Spin-Glass systems (Sherrington-Kirkpatrick model), with a set of bipolar units x_i i.e $x_i \in \{\pm 1\}$ and a quadratic energy function of the form:

$$E(\{x_i\}) = \sum_{ij} x_i W_{ij} x_j + \sum_i x_i h_i \tag{1}$$

. At equilibrium, the probability of any particular configuration $\vec{x}$ [1] is given by the *Boltzmann distribution*:

$$P^m_{\vec{x}} = \frac{1}{\mathcal{Z}_c} e^{-E(\vec{x})} \tag{2}$$

where $\mathcal{Z}_c$ denotes the *partition model* of the model given as $\mathcal{Z}_c = \sum_{\mathcal{C}} e^{-E(\{x_i\})}$ where $\mathcal{C}$ runs over all possible configurations. The aim is to obtain suitable the parameters W_{ij} and h_j so that the the probability distribution generated by the model P_m matches the probability distribution P_{data} of the data.
To accomplish this one recast the task as an optimisation problem, where the aim is to optimise a *cost function* $\mathcal{L}$ over all *trainable parameters* $\{\theta_i\}$[2]. In case of BMs, one uses *KL Divergence* of the P_m wrt. to P_{data} as the cost function:

$$\mathcal{L}(\theta) = -\sum_{\vec{x}\in\mathcal{C}} P^{data}_{\vec{x}} log\Big[\frac{P^{data}_{\vec{x}}}{P^m_{\vec{x}}}\Big] \tag{3}$$

[1]Here we will use the expressions $\{x_i\}$ and $\vec{x}$ interchangeably to represent the configuration of the model.

[2]$\{\theta_i\}$ jointly represents all the trainable parameters in our problem i.e W_{ij} and h_j

The training proceeds by optimising the parameters θ_i using *Gradient-Descent* method as follows:

$$\theta_i = \theta_i - \eta \frac{\partial \mathcal{L}}{\partial \theta_i} \tag{4}$$

where η is the *learning-rate*. The gradient in [4] simplifies to the following elegant expression, often called the *contrastive-divergence* [Mar21]

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = \left\langle \frac{\partial \mathcal{L}}{\partial \theta_i} \right\rangle_{data} - \left\langle \frac{\partial \mathcal{L}}{\partial \theta_i} \right\rangle_{model} \tag{5}$$

Which gives the update rule for our parameters W, h as

$$\frac{\partial \mathcal{L}}{\partial W_{ij}} = \langle x_i x_j \rangle_{data} - \langle x_i x_j \rangle_{model} \tag{6a}$$

$$\frac{\partial \mathcal{L}}{\partial h_i} = \langle x_i \rangle_{data} - \langle x_i \rangle_{model} \tag{6b}$$

2 Quantum Boltzmann Machines

In designing a quantum version of the Boltzmann machines one proceeds as follows:
The classical bipolar units $\{x_i\}$ are replaced with qubits $|q_i\rangle$. The scalar energy function in eq. [1] gets replaced with an analogous *Hamiltonian operator* , given as:

$$\mathcal{H} = \sum_{ij} w_{ij} \sigma^z_i \sigma^z_j + \sum_i \Gamma^z_i \sigma^z_i \tag{7}$$

where w_{ij} and h_i are the trainable parameters as usual and the operator σ^z_a is defined as σ^z at the ath position and identity I at the rest ;

$$\sigma^z_a = I_0 \otimes I_1 \otimes I_2 \otimes .. \otimes \sigma^z \otimes I_{a+1} .. \otimes I_n \tag{8}$$

It is easy to follow that the operator in eq. is diagonal when expressed in computational basis. The diagonal entries would be the values of energy in eq [1]. For example, for a two-qubit system the matrix for [7] would be:

$$\begin{pmatrix} E(00) & 0 & 0 & 0 \\ 0 & E(01) & 0 & 0 \\ 0 & 0 & E(10) & 0 \\ 0 & 0 & 0 & E(11) \end{pmatrix}$$

In order to introduce quantum effects that can be leveraged later, in eq. [7] one introduces terms consisting of operators which do not commute with σ_z. Since non-commuting operators do not share eigenbasis, the resultant Hamiltonian operator now becomes non-diagonal. Commonly one introduces a *transverse field hamiltonian*, which consists of σ_x operators.

$$\mathcal{H}_m = \sum_{ij} w_{ij}\sigma_i^z\sigma_j^z + \sum_i \Gamma_i^z\sigma_i^z + \sum_i \Gamma_i^x\sigma_i^x \tag{9}$$

The eigenvectors of this quantum hamiltonian[3] now would be some complex superposition of the computational basis states.

Defining the density matrix of the ensemble of states;

$$\rho_m = \frac{1}{\mathcal{Z}_q}\exp\left(-\mathcal{H}_m\right) \tag{10}$$

where $\mathcal{Z}_q = Tr[\exp(-\mathcal{H}_m)]$ is the *partition function*. The ρ_m can be re-expressed as:

$$\rho_m = \frac{1}{\mathcal{Z}_q}\sum_{i=0}^{t=2^n-1} e^{-\lambda_i}\left|\lambda_i\right\rangle\left\langle\lambda_i\right| \tag{11}$$

where $\{|\lambda_i\rangle\}$ is the eigen-spectrum of the Hamiltonian $\mathcal{H}_m$. This density matrix describes a Boltzmann distributed ensemble.

To obtain a classical the probability distribution over configurations we measure the system in the computational basis states, $\{|b\rangle\}$ spanning the hilbert space $\mathbb{H}^n$ of the system

$$\mathcal{P}^m_{(|b\rangle)} = Tr[\rho_m\Lambda_b], \;\; where \;\; \Lambda_b = \left|b\right\rangle\left\langle b\right| \tag{12}$$

Λ_b is the projector on the computational basis states defines as

$$\Lambda_b = \left|b\right\rangle\left\langle b\right| = \prod_i\Big(\frac{1+b_i\sigma_i^z}{2}\Big), \;\; b_i \in \{\pm 1\} \tag{13}$$

where b_i are the elements of the classical state. Once we have the classical probability distribution over $\{|b\rangle\}$, we could define the cost function in the usual manner as

$$\mathcal{L}(\theta) = -\sum_{|b\rangle\in\mathbb{H}^n} P^{data}_{|b\rangle}\log\Big[\frac{P^{data}_{|b\rangle}}{\mathcal{P}^m_{|b\rangle}}\Big] \tag{14}$$

Next we can take the derivative of $\mathcal{L}(\theta)$ wrt. to the parameters θ as,

$$\partial_\theta\mathcal{L} = \sum_{|b\rangle\in\mathbb{H}^n} P^{data}_{|b\rangle}\Big[\frac{Tr[\Lambda_b\partial_\theta e^{-\mathcal{H}}]}{Tr[\Lambda_b e^{-\mathcal{H}}]} - \frac{Tr[\partial_\theta e^{-\mathcal{H}}]}{Tr[e^{-\mathcal{H}}]}\Big] \tag{15}$$

Now let's first consider the second term which could be simplified further by *trotterizing* the operator $e^{-\mathcal{H}}$ as

$$e^{-\mathcal{H}} = \prod_{i=1}^{N} e^{-\mathcal{H}\delta t} \quad , \delta t = \frac{1}{N}$$

and then taking the gradient as,

$$\partial_\theta e^{-\mathcal{H}} = \sum_m \Big(e^{-\mathcal{H}\delta t}\Big)^m \partial_\theta e^{-\mathcal{H}\delta t}\Big(e^{-\mathcal{H}\delta t}\Big)^{N-m}$$

Now, taking the limit of $n\to\infty$ and taking trace on both the sides we have,

$$\begin{aligned} Tr[\partial_\theta e^{-\mathcal{H}}] &= Tr\Big[\sum_m \Big(e^{-\mathcal{H}\delta t}\Big)^m \partial_\theta e^{-\mathcal{H}\delta t}\Big(e^{-\mathcal{H}\delta t}\Big)^{N-m}\Big] \\ &\approx Tr\Big[\int_0^1 d\tau e^{-\tau\mathcal{H}}\partial_\theta\mathcal{H}e^{-(1-\tau)\mathcal{H}}\Big] \\ &= Tr\Big[e^{-\mathcal{H}}\partial_\theta\mathcal{H}\Big] \end{aligned}$$

so the second term becomes,

$$\frac{Tr[\partial_\theta e^{-\mathcal{H}}]}{Tr[e^{-\mathcal{H}}]} = \langle\partial_\theta\mathcal{H}\rangle \tag{16}$$

which can be estimated by sampling from the physical distribution. However such a simplification is not possible for the first term without which common training methods would not be applicable. As a workaround for this problem, Amin et al. proposed to minimize a properly defined upper bound on $\mathcal{L}$. This is what they called Bound-based Quantum Boltzmann Machine.

3 Bound based QBM

The basic idea comes from *Golden-Thompson inequality* [Ami+18], which states that

$$Tr\big[e^{A+B}\big] \le Tr\big[e^A e^B\big] \tag{17}$$

for any hermitian matrices A and B.

This allows us to write

$$Tr\big[e^{-\mathcal{H}}e^{ln\Lambda_b}\big] \ge Tr\big[e^{-\mathcal{H}+ln\Lambda_b}\big] \tag{18}$$

. Since Λ_b is a projector onto the computational basis state $|b\rangle$ the action of $ln\Lambda_b$ is to map any state orthogonal to the state $|b\rangle$ to $-\infty$, thus it follows that,

$$Tr\big[e^{-\mathcal{H}+ln\Lambda_b}\big] = Tr\big[e^{-\mathcal{H}_b}\big]$$

where $\mathcal{H}_b \equiv \langle b|\mathcal{H}|b\rangle$ Thus 18 becomes,

$$\frac{Tr\big[e^{-\mathcal{H}}e^{ln\Lambda_b}\big]}{Tr\big[e^{-\mathcal{H}}\big]} \ge \frac{Tr\big[e^{-\mathcal{H}_b}\big]}{Tr\big[e^{-\mathcal{H}}\big]} \tag{19}$$

So we can modify the first term in the cost function in 14 as,

$$\mathcal{L} \le \tilde{\mathcal{L}} = \sum_{|b\rangle} P^{data}_{|b\rangle} log\frac{Tr\big[e^{-\mathcal{H}_b}\big]}{Tr\big[e^{-\mathcal{H}}\big]} \tag{20}$$

Hence eq. [15] becomes

$$\partial_\theta\mathcal{L} = \sum_{|b\rangle\in\mathbb{H}^{2^n}} P^{data}_{|b\rangle}\Big[\frac{Tr[e^{-\mathcal{H}_b}\partial_\theta\mathcal{H}_b]}{Tr[e^{-\mathcal{H}_b}]} - \frac{Tr[\partial_\theta e^{-\mathcal{H}}]}{Tr[e^{-\mathcal{H}}]}\Big] \tag{21}$$

$$= \sum_{|b\rangle\in\mathbb{H}^{2^n}} P^{data}_{|b\rangle}\big[\,\langle\partial_\theta\mathcal{H}_b\rangle_{|b\rangle} - \langle\partial_\theta e^{-\mathcal{H}}\rangle\,\big] \tag{22}$$

$$= \langle\partial_\theta\bar{\mathcal{H}}_b\rangle_{|b\rangle} - \langle\partial_\theta\mathcal{H}\rangle \tag{23}$$

[3] The QML, the diagonal hamiltonian given in eq. [7] is called *a classical hamiltonian* whereas the non-diagonal one given by eq. [9] is called *a quantum hamiltonian ([SP18],[Rud20])*.

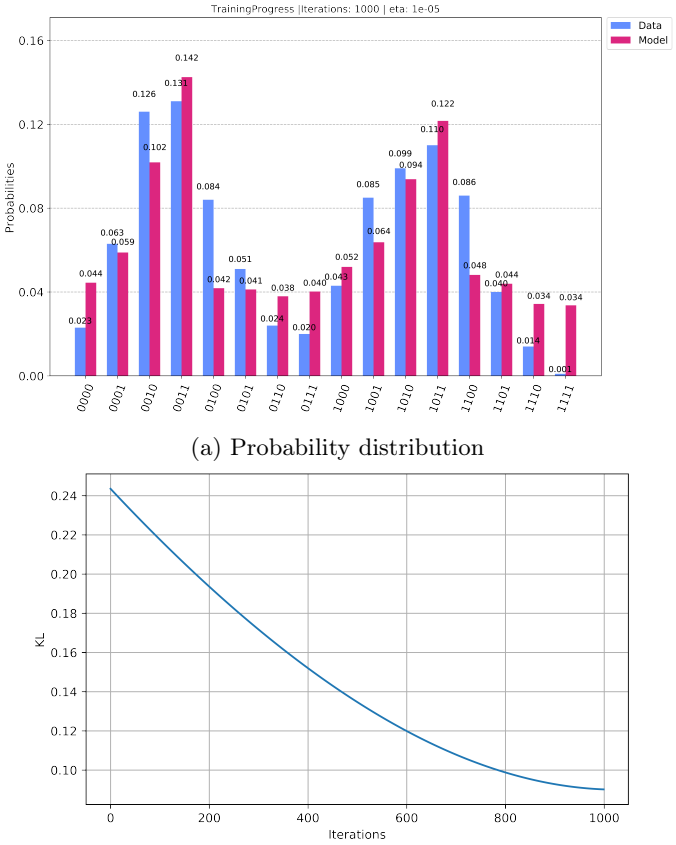
Note the similarity between equation [5] and equation [23], the first term corresponds to the sampling $\partial_\theta \mathcal{H}_b$ over the computational basis states $|b\rangle$ over the data distribution $P^{data}_{|b\rangle}$ while the second term corresponds to sampling $\partial_\theta \mathcal{H}$ over the model distribution generated by the QBM.
The Bound-based-QBM leads to a tractable gradient formula [23], and the update rules are similar to that in the classical case which is easy to implement. However one of the drawbacks of using eq. [23] is that we cannot use it to update Γ^x_j as expectation value of σ^x operators[4] in the computational basis states always yield to zero. This is an artifact of minimizing upper bound on the K.L. divergence. However one can still proceed by treating the Γ^x as hyper-parameter.

4 An interesting insight

Kieferova and Weibe [KW17] proposed *Relative entropy* $S(\rho_{data}||\sigma)$ as the objective function for training a QBM. In the case of a classical data sets, for a fully visible model, upto a constant

$$\begin{aligned} O(H) &= -Tr\left[\rho_{data} ln\left(\frac{e^{-H}}{Tr(e^{-H})}\right)\right] \\ &= Tr[\rho_{data} H] + Tr(\rho_{data} log[Tr(e^{-H})]) \\ &= \sum_{\vec{z}} P^{data}_{\vec{z}} H_{\vec{z}} + log[Tr(e^{-H})] \\ &= -\sum_{\vec{z}} P^{data}_{\vec{z}} log\left(\frac{Tr(e^{-H_{\vec{z}}})}{Tr[e^{-H}]}\right) = \tilde{\mathcal{L}} \end{aligned}$$

which is the bound proposed by Amin et al.



(a) Probability distribution

(b) KL Divergence

Figure 1: (a) Histogram to provide a visual aid for comparison between the data distribution model 1 (blue) and the probability distribution learned by the QBM (red). eta $= 1 \times e^{-05}$ is the learning rate. (b)KL divergence during the training process.

5 Implementation

5.1 Methodology

We numerically implemented the proposed Quantum Boltzmann Machine learning model for a small toy problem of the size $N = 4$ qubits and a completely visible model. Moreover we considered 2 cases of Hamiltonian: The classical diagonal one and the quantum non-diagonal one. The eigen-vectors and corresponding eigenvalues of the Hamiltonian operator were obtained using QuSpin, an open source python package which utilises the exact-diagonalisation procedure to achieve the task. For training, we implemented the naïve gradient descent.

5.2 Results

In figure [1] and [2], we present result for two of the toy problems we considered. In both cases, the learning rate η was chosen to be $1 \times e^{-05}$.

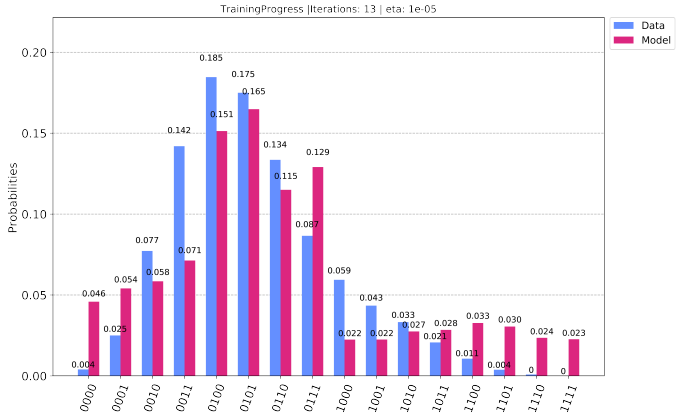
5.2.1 Tuning the Hyper-parameter Γ^x

As pointed out earlier, Γ^x cannot be trained in the framework of bound based QBM model and needs to be treated as a hyper-parameter if one wants to use the gradient-descent algorithm for training. One of the major issues that we faced was setting its value. We observed that (cite figure here) that the rate of convergence of the model depends significantly on the choice of Γ^x.
We observed that the minimum *KL divergence* (KL_{min}) obtained is almost the same when Γ^x is varied keeping the order same ($\sim 10^1$). However the number of iterations it requires to reach KL_{min} is different in each case. We also observed that after a certain number of iterations, the *KL divergence* was increasing drastically, which is contrary to our expectations of getting a monotonous decrease in the value of the cost function under the gradient descent algorithm. This indicates that either there is some numerical error in our implementation, or this might have to do with the inefficient nature of the gradient descent while training a QBM. For the want of time we could not pursue this further but will definitely look into it in the future.

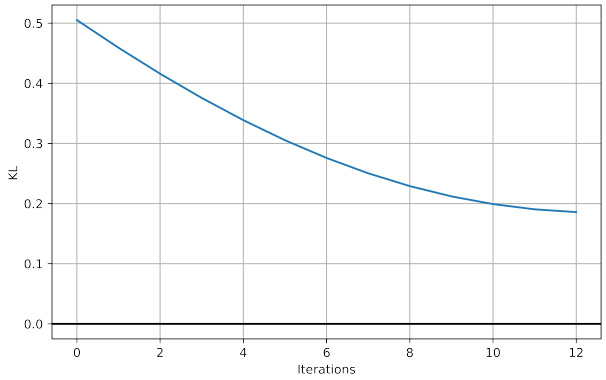
5.2.2 Tuning the learning rate η

We also varied the learning rate of the model. Here also we observed a similar behaviour in *KL divergence* as we did for the case of different $\Gamma^x s$. We are still investigating this issue.

[4] here, $\partial_{\Gamma^x_j}(\mathcal{H}) = \sigma^x_j$



(a) Probability distribution



(b) KL Divergence

Figure 2: (a) Histogram to provide a visual aid for comparison between the data distribution model 2 (blue) and the probability distribution learned by the QBM (red). eta $= 1 \times e^{-05}$ is the learning rate. (b)KL divergence during the training process.

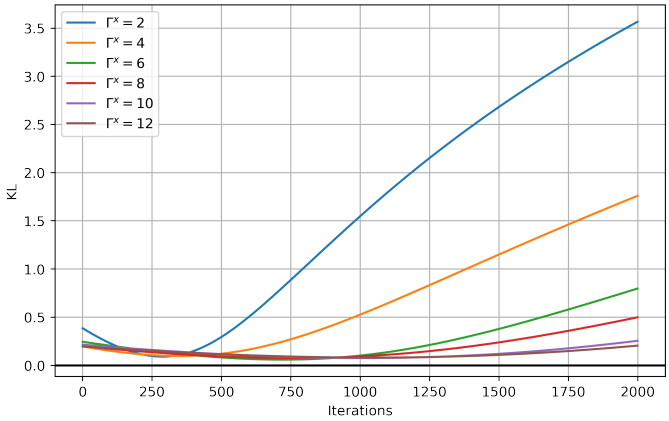
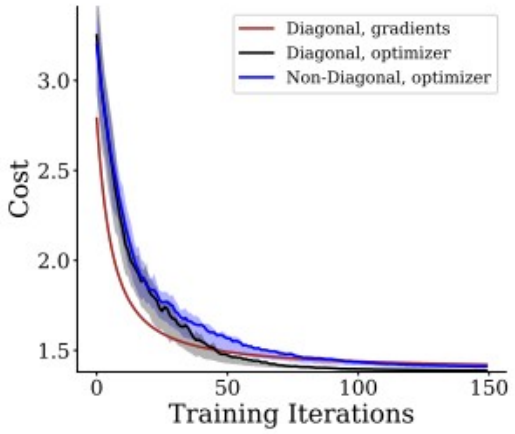


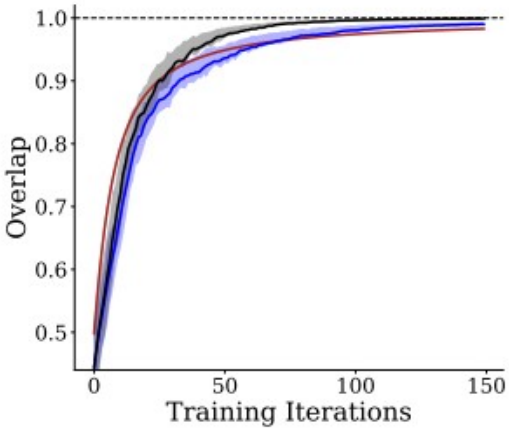
Figure 3: KL divergence for different choices of the hyperparameter Γ_x.

6 Training Quantum model of Boltzmann machine by a classical optimizer

The reason Amin et al. had to introduce a bound was to obtain samplable terms in the expression for the training through gradient descent. However an obvious question which gets raised at this point is that could one work with the quantum model for Boltzmann machine whose parameters are trained using a some other classical tool as we know that gradient descent is not the only way one can train a model? The answer to this question is that it is indeed possible and this work was pursued by Manuel Rudolf [Rud20]. In this work, the parameters were trained using *Covariance matrix adaptation-encoding strategy (cma-es)* optimizer. The advantage of this is that one can now also train the transient field terms Γ^x. Figure [4] shows the result obtained on the Bars-and-Stripes data-set.



(a) KL divergence during training process



(b) Overlap during training process

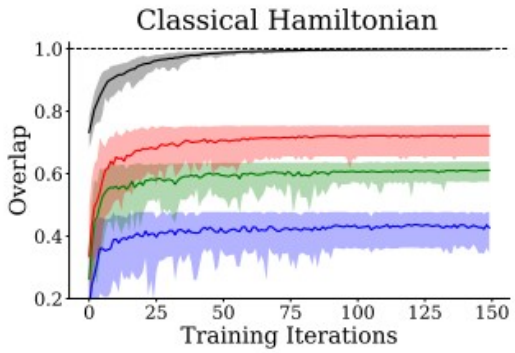
Figure 4: Evaluation of the average training progress for three Quantum Boltzmann Machines on the Bars-and-Stripes training set over 100 repetitions. These plots have been taken from [Rud20]

7 How good is QBM?

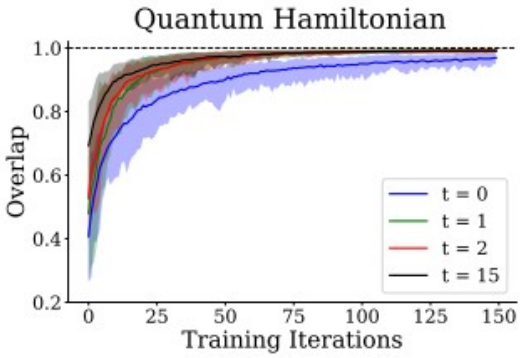
Two important questions one must ask is (a) Why do we need Quantum Boltzmann Machines and (b) What new insight this model brings to the table that can prove to be advantageous? These questions are addressed below:

7.1 Groundstate training

Let's first try to address the second question. Consider the expression for the density matrix describing the Boltzmann distributed ensembles (eq. 11), which includes all 2^n eigen-states ($t = 2^n - 1$). Constructing such a complete Boltzmann density matrix on a near-term quantum computer is, however, quite impractical. Question that gets



(a) Training using classical hamiltonian



(b) Training using Quantum hamiltonian

Figure 5: Training process of QBMs with (a) classical and (b) quantum hamiltonian with truncated density matrix for a system of n=4 qubits on random training sets. One can observe that even with truncated density matrix, BM with Quantum hamiltonian (5b) is able to learn the probability distribution of the data quite well. These plots have been taken from [Rud20].

.

raised at this point is can the truncated density matrix be considered in QBM model ($t < 2^n - 1$)? M.Rudolph [Rud20] carried out this study for an n=4 qubit system with t=15,2,1 and even 0 for both classical and quantum hamiltonian based QBM models. The expressivity was benchmarked by training models on random training set consisting of 10 random binary samples each and the training process was averaged over 40 repetitions with a different random training set for each repetition. The result of this study are presented in fig [5]. It shows that a QBM with classical hamiltonian cannot be trained properly with a small number of eigenstates contributing to the Boltzmann distribution. This is because each eigenstate of a diagonal hamiltonian contains exactly one basis state with amplitude 1 and thus all 2^n eigenstates are required to learn a random distribution. The reason the non-diagonal hamiltonian is able to perform so well with any level of truncation have to do with the fact that its eigenvectors are the complex superposition of computational basis states. This opens an avenue for *ground state training* as even a single quantum ground state can be trained to approximate any arbitrary distribution. Inclusion of additional eigenstates would just increase the training performance per iteration.

7.2 Expressivity of the Quantum models

The question "Why Quantum model for Boltzmann Machines?" can be addressed using the notion of expressivity of a model. One can note that when the QBM with diagonal and non-diagonal hamiltonian, is given access to all the basis states, then the final overlap of the model is almost 1 for some random training sets considered. This hints towards the high expressivity of the Quantum Boltzmann Machines.

8 Conclusion

We summarised main features of an interesting extension of classical Boltzmann machines to a quantum framework proposed by Amin et al . We also implemented the scheme numerically to some small sized toy problems. One thing we would like to mention is that the dependence of the learned log-likelihood on the transverse field is not so weak, at-least for the toy problems we considered, which is quite in contrast with Amin et al's claim.

References

[Hin07] Geoffrey Hinton. "Boltzmann machine". In: *Scholarpedia* 2.5 (Jan. 2007), p. 1668. DOI: 10.4249/scholarpedia.1668.

[KW17] Mária Kieferová and Nathan Wiebe. "Tomography and generative training with quantum Boltzmann machines". In: *Phys. Rev. A* 96 (6 Dec. 2017), p. 062327. DOI: 10.1103/PhysRevA.96.062327. URL: https://link.aps.org/doi/10.1103/PhysRevA.96.062327.

[Ami+18] Mohammad H. Amin et al. "Quantum Boltzmann Machine". In: *Phys. Rev. X* 8 (2 May 2018), p. 021050. DOI: 10.1103/PhysRevX.8.021050. URL: https://link.aps.org/doi/10.1103/PhysRevX.8.021050.

[SP18] Maria Schuld and Francesco Petruccione. *Supervised Learning with Quantum Computers*. 1st. Springer Publishing Company, Incorporated, 2018. ISBN: 3319964232. DOI: https://doi.org/10.1007/978-3-319-96424-9. URL: https://link.springer.com/book/10.1007/978-3-319-96424-9.

[Rud20] Manuel Rudolph. "Exploring and Benchmarking Quantum-assisted Neural Networks with Qubit Layers". Masterarbeit. Universität Heidelberg, 2020. URL: https://jendrzejewski.synqs.org/publication/thesis/rudolph-2020/.

[Mar21] Florian Marquardt. "Machine Learning and Quantum Devices". In: *SciPost Phys. Lect. Notes* (2021), p. 29. DOI: 10.21468/SciPostPhysLectNotes.29. URL: https://scipost.org/10.21468/SciPostPhysLectNotes.29.