

Unsupervised Generative Modeling Using Matrix Product States

Zhao-Yu Han,^{1,*} Jun Wang,^{1,*} Heng Fan,² Lei Wang,^{2,†} and Pan Zhang^{3,‡}

¹*School of Physics, Peking University, Beijing 100871, China*

²*Institute of Physics, Chinese Academy of Sciences, Beijing 100190, China*

³*Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China*

Generative modeling, which learns joint probability distribution from training data and generates samples according to it, is an important task in machine learning and artificial intelligence. Inspired by probabilistic interpretation of quantum physics, we propose a generative model using *matrix product states*, which is a tensor network originally proposed for describing (particularly one-dimensional) entangled quantum states. Our model enjoys efficient learning by utilizing the *density matrix renormalization group* method which allows dynamic adjusting dimensions of the tensors, and offers an efficient direct sampling approach, *Zipper*, for generative tasks. We apply our method to generative modeling of several standard datasets including the principled *Bars and Stripes*, random binary patterns and the *MNIST* handwritten digits, to illustrate ability of our model, and discuss features as well as drawbacks of our model over popular generative models such as Hopfield model, Boltzmann machines and generative adversarial networks. Our work shed light on many interesting directions for future exploration on the development of quantum-inspired algorithms for unsupervised machine learning, which is of possibility of being realized by a quantum device.

I. INTRODUCTION

Generative modeling, a typical unsupervised learning that makes use of huge amount of unlabeled data, lies in the heart of rapid development of modern machine learning techniques [1]. Different from discriminative tasks such as pattern recognition, the goal of generative modeling is to model the probability distribution of input data and thus be able to generate *new* samples according to the distribution. At the research frontier of generative modeling, it was used for finding good data representation and dealing with tasks with missing data. Popular generative machine learning models include the Boltzmann Machines (BM) [2, 3] and their generalizations [4], variational autoencoders (VAE) [5], autoregressive models [6, 7], nonlinear density estimations [8–10], and the generative adversarial networks (GAN) [11]. For generative model design, one tries to balance the representational power and efficiency of learning and sampling.

There is a long history of relation between generative modeling and physics, especially statistical physics. Some celebrated models, such as Hopfield model [12], and Boltzmann machine [2, 3], are closely related to the Ising model in statistical physics, and its inverse version which learns couplings in the Ising model based on given training configurations [13, 14].

The task of generative modeling also shares many similarities with quantum physics research in the sense that both of them try to model probability distributions in an enormously large space. In the past decades, tensor network (TN) states and algorithms have been shown to be an incredibly potent tool set for studying many-body quantum physics with its power in expressing quantum states relevant to realistic situations [15, 16]. The success of TN description of quantum

states can be theoretically justified from a quantum information point of view [17, 18]. Tensor decomposition and tensor networks have also been applied in a broader context by the machine learning community for feature extraction, dimensionality reduction and analyzing the expressibility of deep neural nets [19–24]. In particular, matrix product states (MPS) consist of a subset of TN where the tensors are arranged in a one-dimensional geometry [25]. The same representation is referred as tensor train decomposition in the applied math community [26]. Despite of its simple structure, MPS can represent a large number of quantum states extremely well. MPS representation of ground states has been proven to be efficient to find for one-dimensional gapped local system [27]. In practice, optimization schemes for MPS such as density-matrix renormalization group (DMRG) [28] have been successful even for some quantum systems in higher dimension [29]. Recently, References [30, 31] extended the application of MPS to pattern recognition and classification tasks in machine learning. The authors of Reference [32] drew connection between Boltzmann Machines and tensor networks, and reference [33] used tensor networks for language modeling.

In this paper, building on the connection between unsupervised generative modeling and quantum physics, we employ MPS as an unsupervised generative model to learn probability distribution of given data by resembling the DMRG algorithm [28]. We also employ a direct sampling approach [34], dubbed *Zipper*, for efficient generation of samples from the trained MPS. Compared with statistical-physics based models such as the Hopfield model [12] and inverse Ising model, the MPS model exhibits much larger capability of learning, as the capability increases by increasing bond dimension of the MPS. Our model also enjoys a much more efficient direct samplings than the Boltzmann machines, which require Markov Chain Monte Carlo (MCMC) approach for data generation. When compared with popular generative models such as GAN, our model offers easier and efficient reconstruction and denoising from an initial (noisy) input using the Zipper algorithm, as opposed to GAN where mapping a noisy image to its input is not straight forward.

* These two authors contributed equally

† wanglei@iphy.ac.cn

‡ panzhang@itp.ac.cn

The rest of the paper is organized as follows. In Sec. II we present our model, show how to train the model with DMRG, and how to generate new samples using the Zipper algorithm. In Sec. III we apply our model to three datasets: Bars-and-Strips for a proof-of-principle demonstration, random binary patterns for demonstrating capability of the MPS probabilistic model, and finally the MNIST handwritten digits for illustrating the ability of the MPS model in accomplishing unsupervised tasks such as reconstruction and denoising of images. Finally, Sec. IV discusses future prospects of the generative modeling using more general tensor networks.

II. MPS FOR UNSUPERVISED LEARNING

The goal of unsupervised generative modeling is to model the joint probability distribution of given training data. With the trained model, one can then generate new samples from the learned probability distribution. Generative modeling finds wide applications such as dimensional reduction, feature detection, clustering and recommender systems [35]. In this paper, we consider a data set \mathcal{T} consisting of binary strings $\mathbf{v} \in \mathcal{V} = \{0, 1\}^{\otimes N}$ that are potentially repeated in \mathcal{T} . We adopt the negative log-likelihood (NLL) averaged on the training set as the cost function

$$\mathcal{L} = -\frac{1}{|\mathcal{T}|} \sum_{\mathbf{v} \in \mathcal{T}} \ln [\mathbb{P}(\mathbf{v})], \quad (1)$$

where $|\mathcal{T}|$ denotes the size of the training set. Minimizing the NLL reduces the dissimilarity between the model probability distribution $\mathbb{P}(\mathbf{v})$ and the empirical distribution defined by the training set. It is well-known that minimizing \mathcal{L} is equivalent to minimizing the Kullback-Leibler divergence between the two distributions [36].

A. MPS Representation of Probability Distribution

Probabilistic interpretation of quantum mechanics [37] naturally suggests a generative modeling approach. Suppose we encode the probability distribution into a quantum wavefunction $\Psi(\mathbf{v})$, measurement will collapse it and generate a result $\mathbf{v} = (v_1, v_2, \dots, v_N)$ with a probability proportional to $|\Psi(\mathbf{v})|^2$. Inspired by the generative aspects of quantum mechanics, we represent the model probability distribution by

$$\mathbb{P}(\mathbf{v}) = \frac{|\Psi(\mathbf{v})|^2}{Z}, \quad (2)$$

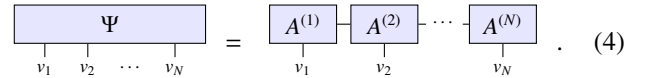
where $Z = \sum_{\mathbf{v} \in \mathcal{V}} |\Psi(\mathbf{v})|^2$ is the normalization factor. We also refer it as the *partition function* to draw an analogy with the energy based models [38]. In general the wavefunction $\Psi(\mathbf{v})$ can be complex valued, but in this work we restricted it to be real valued. Representing probability density using square of a function was also put forward by [39, 40]. These approach ensures the positivity of probability and naturally admit a quantum mechanical interpretation.

Quantum physicists and chemists have also developed many efficient classical representations of quantum wavefunctions. A number of these developed representations and algorithms can be adopted for efficient probabilistic modeling. Here, we parametrize the wave function using the MPS:

$$\Psi(v_1, v_2, \dots, v_N) = \text{Tr}(A^{(1)v_1} A^{(2)v_2} \dots A^{(N)v_N}), \quad (3)$$

where each $A^{(k)v_k}$ is a \mathcal{D}_{k-1} by \mathcal{D}_k matrix, and $\mathcal{D}_0 = \mathcal{D}_N$ is demanded to close the trace. There are $2 \sum_{k=1}^N \mathcal{D}_{k-1} \mathcal{D}_k$ parameters on the right-hand-side of Eq. (3), where the coefficient of 2 results from the binary physical indices. Representational power of the MPS is related to the Von Neumann entanglement entropy of the quantum state, which is defined as $S = -\text{Tr}(\rho_A \ln \rho_A)$. Here we divide the variables into two groups $\mathbf{v} = (\mathbf{v}_A, \mathbf{v}_B)$ and $\rho_A = \sum_{\mathbf{v}_B} \Psi(\mathbf{v}_A, \mathbf{v}_B) \Psi(\mathbf{v}_A, \mathbf{v}_B)^\dagger$ is the reduced density matrix of a subsystem. The entanglement entropy sets an lower bound for the bond dimension at the division $S \leq \ln(\mathcal{D}_k)$. Any probability distribution of a N -bit system can be described by an MPS as long as its bond dimensions are free from any restriction. Therefore as the bond dimension increases, an MPS increases its capability of parameterizing complicated functions. The inductive bias of representing quantum states using MPS with finite bond dimensions is low entanglement entropy of these states. See [15] and [16] for recent reviews on MPS and its applications on quantum many-body systems.

In practice, it is convenient to use MPS with $\mathcal{D}_0 = \mathcal{D}_N = 1$ and consequently reduce the left and right most matrices to vectors [28]. In this case, Eq. (3) reads schematically



$$\Psi = A^{(1)} A^{(2)} \dots A^{(N)}. \quad (4)$$

Here the block denotes the matrix and the connected lines indicate matrix product over virtual indices. The dangling vertical bonds denote physical indices. We refer to [15, 16] for an introduction to these graphical notations of TN. Henceforth, we shall present formulae with more intuitive graphical notations wherever possible.

The MPS representation has gauge degrees of freedom, which means that the state is invariant after inserting identity $I = MM^{-1}$ on each bond (M can be different on each bond). To remove this redundancy, one can bring the MPS into its canonical form. For example, the tensor $A^{(k)}$ is called left-canonical if it satisfies $\sum_{v_k \in \{0,1\}} [A^{(k)v_k}]^\dagger A^{(k)v_k} = I$. In diagrammatic notation, the left-canonical condition reads



$$= \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \quad (5)$$

The right-canonical condition is defined analogously. Canonicalization of each tensor can be done locally and only involves the single tensor at consideration [15, 16].

Each tensor in the MPS can be in a different canonical form. For example, given a specific site k , one can conduct gauge transformation to make all the tensors on the left,

$\{A^{(i)}|i = 1, 2, \dots, k-1\}$, left-canonical and tensors on the right, $\{A^{(i)}|i = k+1, k+2, \dots, N\}$, right-canonical, while leaving $A^{(k)}$ neither left-canonical nor right-canonical. This is called mixed-canonical form of the MPS [15]. The normalization of the MPS is particularly easy to compute in the canonical form. In the graphical notation, it reads

$$Z = \begin{array}{c} \square \cdots \square \cdots \square \\ \square \cdots \square \cdots \square \end{array} = \begin{array}{c} A^{(k)} \\ A^{(k)} \end{array}. \quad (6)$$

We note that even if the MPS is not in the canonical form, its normalization factor Z can be still computed efficiently if one pays attention to the order of contraction [15, 16].

B. Learning MPS from data

A standard way of minimization of the cost function (1) is done by performing gradient decent algorithm on the MPS tensor elements. Crucially, our method allows dynamical adjustment on the bond dimension during the optimization, thus being able to allocate resources to the spatial regions where correlations among the physical variables are stronger.

Initially, we set the MPS with random tensors with small bond dimensions. For example, all the bond dimension are set to $\mathcal{D}_k = 2$ except those on the boundaries [41]. We then carry out the canonicalization procedure so that all the tensors except the rightmost one ($A^{(N)}$) are left-canonical. Then, we sweep through the matrices back and forth to tune the elements of the tensors, i.e. the parameters of the MPS. The procedure is similar to the DMRG algorithm with two-site update where one optimizes two adjacent tensors at a time [28]. At each step, we firstly merge two adjacent tensors into an order-4 tensor,

$$\begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} A^{(k)} \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array} = \begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} A^{(k,k+1)} \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array}, \quad (7)$$

followed by adjusting its elements in order to decrease the cost function $\mathcal{L} = \ln Z - \frac{1}{|\mathcal{T}|} \sum_{\mathbf{v} \in \mathcal{T}} \ln |\Psi(\mathbf{v})|^2$. It is straight forward to check that its gradient with respect to an element of the tensor (7) reads

$$\frac{\partial \mathcal{L}}{\partial A_{i_{k-1} i_{k+1}}^{(k,k+1)}} = \frac{Z'}{Z} - \frac{2}{|\mathcal{T}|} \sum_{\mathbf{v} \in \mathcal{T}} \left[\frac{\Psi'(\mathbf{v})}{\Psi(\mathbf{v})} \right], \quad (8)$$

where $\Psi'(\mathbf{v})$ denotes the derivative of the MPS with respect to the tensor (7), and $Z' = 2 \sum_{\mathbf{v} \in \mathcal{V}} \Psi'(\mathbf{v}) \Psi(\mathbf{v})$. In diagram language, they read

$$\Psi'(\mathbf{v}) = \begin{array}{c} \square \cdots \square \begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} w_k \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array} \cdots \square \\ v_1 \quad v_{k-1} \quad v_k \quad v_{k+1} \quad v_{k+2} \quad v_N \end{array} \quad (9)$$

$$\begin{aligned} \frac{Z'}{2} &= \begin{array}{c} \square \cdots \square \begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} w_k \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array} \cdots \square \\ \square \cdots \square \begin{array}{c} w_k \\ \square \\ v_{k+1} \end{array} \begin{array}{c} w_{k+1} \\ \square \\ v_{k+1} \end{array} \cdots \square \end{array} \\ &= \begin{array}{c} w_k \quad w_{k+1} \\ \square \\ i_{k-1} \end{array} \begin{array}{c} A^{(k,k+1)} \\ \square \\ i_{k+1} \end{array} \end{aligned} \quad (10)$$

The direct vertical connections of w_k, v_k and w_{k+1}, v_{k+1} in (9) stand for Kronecker delta functions $\delta_{w_k v_k}$ and $\delta_{w_{k+1} v_{k+1}}$ respectively, meaning that only those input data with pattern $v_k v_{k+1}$ contribute to the gradient with respect to the tensor elements $A_{i_{k-1} i_{k+1}}^{(k,k+1)}$. Note that although Z and Z' involve summations over an exponentially large number of terms, they are tractable in MPS via efficient contraction schemes [15]. In particular, if the MPS is in the mixed canonical form, the computation only involves local manipulations illustrated in (10).

Next, we employ gradient descent approach to update the parameters of the 2-site tensor

$$A_{i_{k-1} i_{k+1}}^{(k,k+1) w_k w_{k+1}} = A_{i_{k-1} i_{k+1}}^{(k,k+1) w_k w_{k+1}} - \eta \frac{\partial \mathcal{L}}{\partial A_{i_{k-1} i_{k+1}}^{(k,k+1) w_k w_{k+1}}}, \quad (11)$$

where $\eta > 0$ is the learning rate. In practice, we divide the training data into n_b mini-batches randomly selected during the training, then use them to estimate the second term in the gradient [Eq.(8)] stochastically.

After applying several steps of gradient descent Eq. (11), the order-4 tensor (7) is decomposed, by first reshaping the tensor to a matrix, then applying *singular value decomposition* (SVD), and finally reshaping obtained two matrices back to two order-3 tensors.

$$\begin{aligned} \begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} A^{(k,k+1)} \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array} &= \begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} A^{(k,k+1)} \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array} \\ &= \begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} U \\ \square \\ v_k \end{array} \begin{array}{c} \Lambda \\ \square \\ v_{k+1} \end{array} \begin{array}{c} V^\dagger \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array} \\ &\approx \begin{array}{c} i_{k-1} \\ \square \\ v_k \end{array} \begin{array}{c} A^{(k)} \\ \square \\ v_k \end{array} \begin{array}{c} A^{(k+1)} \\ \square \\ v_{k+1} \end{array} \begin{array}{c} i_{k+1} \\ \square \\ v_{k+1} \end{array}, \quad (12) \end{aligned}$$

where U, V are unitary matrices and Λ is a diagonal matrix containing singular values on the diagonal. The number of non-vanishing singular values will generally increase compared to the original value in Eq. (7) because the MPS observes correlations in the data and try to capture them. We truncate those singular values whose ratios to the largest one are smaller than a prescribed hyperparameter cutoff, along with their corresponding row vectors and column vectors deleted in U and V^\dagger .

If the next bond to train on is the $(k+1)$ -th bond on the right, take $A^{(k)} = U$ so that it is left-canonical, and consequently $A^{(k+1)} = \Lambda V^\dagger$. While if the MPS is about to be trained on the $(k-1)$ -th bond, analogously $A^{(k+1)} = V^\dagger$ will be right-canonical and $A^{(k)} = U\Lambda$. This keeps the MPS in mixed-canonical form.

The whole training process consists of many loops. In each loop the training starts from the rightmost bond (the $(N-1)$ -th) and sweeps to the leftmost (the first), then back to the rightmost.

C. Generative Sampling of MPS: The Zipper Algorithm

After training, samples can be generated independently according to Eq. (2).

In other popular generative models, especially the energy based model such as RBM [3], generating new samples is often accomplished by running MCMC from an initial configuration, due to the intractability of the partition function. In our model, one convenience is that the partition function can be computed exactly with computational complexity linear in system size. Thus our model enjoys a direct sampling method which generates a sample bit by bit from one end of the MPS to the other [34]. The detailed generating process is as follows:

It starts from one of the ends, say the N -th bit. One directly draws sample from the marginal probability $\mathbb{P}(v_N) = \sum_{v_1, v_2, \dots, v_{N-1}} \mathbb{P}(\mathbf{v})$. It is clear that this can be easily performed if we have gauged all the matrices except $A^{(N)}$ to be left-canonical because $\mathbb{P}(v_N) = \|\mathbf{x}^{v_N}\|^2/Z$, where we define $\mathbf{x}_{i_{N-1}}^{v_N} = A_{i_{N-1}}^{(N)v_N}$ and the normalization factor reads $Z = \sum_{v_N \in \{0,1\}} \|\mathbf{x}^{v_N}\|^2$. Given the value of the N -th bit, one can then move on to sample the $(N-1)$ -th bit. More generally, given the bit values v_k, v_{k+1}, \dots, v_N , the $(k-1)$ -th bit is sampled according to the conditional probability

$$\mathbb{P}(v_{k-1}|v_k, v_{k+1}, \dots, v_N) = \frac{\mathbb{P}(v_{k-1}, v_k, \dots, v_N)}{\mathbb{P}(v_k, v_{k+1}, \dots, v_N)}. \quad (13)$$

As a result of the canonical condition, the marginal probability can be simply expressed as

$$\mathbb{P}(v_k, v_{k+1}, \dots, v_N) = \|\mathbf{x}^{v_k, v_{k+1}, \dots, v_N}\|^2/Z, \quad (14)$$

where $\mathbf{x}_{i_{k-1}}^{v_k, v_{k+1}, \dots, v_N} = \sum_{i_k, i_{k+1}, \dots, i_N} A_{i_{k-1}i_k}^{(k)v_k} A_{i_k i_{k+1}}^{(k+1)v_{k+1}} \dots A_{i_{N-1}i_N}^{(N)v_N}$. In graphical notation, its squared norm reads

$$\|\mathbf{x}^{v_k, v_{k+1}, \dots, v_N}\|^2 = \begin{array}{c} \begin{array}{c} \square \quad \dots \quad \square \quad \square \\ | \quad \quad \quad | \quad | \\ v_k \quad \quad v_{N-1} \quad v_N \end{array} \\ \begin{array}{c} \square \quad \dots \quad \square \quad \square \\ | \quad \quad \quad | \quad | \\ v_k \quad \quad v_{N-1} \quad v_N \end{array} \end{array}. \quad (15)$$

Multiplying the matrix $A^{(k-1)v_{k-1}}$ from the left, and calculating the squared norm of the resulting vector $\mathbf{x}_{i_{k-2}}^{v_{k-1}, v_k, \dots, v_N} = \sum_{i_{k-1}} A_{i_{k-2}i_{k-1}}^{(k-1)v_{k-1}} \mathbf{x}_{i_{k-1}}^{v_k, v_{k+1}, \dots, v_N}$, one obtains

$$\mathbb{P}(v_{k-1}, v_k, \dots, v_N) = \|\mathbf{x}^{v_{k-1}, v_k, \dots, v_N}\|^2/Z. \quad (16)$$

Combining Eqs. (14, 16) one can compute the conditional probability Eq. (13) and sample the bit v_{k-1} accordingly. In this way, all the bit values are successively drawn from the conditional probabilities given all the bits on the right. This procedure gives a sample strictly obeying the probability distribution of the MPS within $O(N)$ operations. Since the process resembles unzipping [34], it is named *Zipper* sampling approach of an MPS.

The *Zipper* sampling approach is not limited to generation samples from scratch in a sequential order. It is also capable of inference tasks when some bits are given. In that case, the canonicalization trick may not help greatly if there is a segment of unknown bits sitting between given bits. Nevertheless, the marginal probabilities are still tractable because one can contract ladder-shaped TN efficiently [15, 16]. As what will be shown in the Sec. III, given these flexibilities of the *Zipper* sampling approach, MPS based probabilistic modeling can be applied to image reconstruction and denoising.

D. Features of the model and algorithms

We highlight several salient features of the MPS generative model and compare it to other popular generative models. Most significantly, the MPS has an explicit tractable probability density, while still allows efficient learning and inference. For a system sized N , assume one prescribes the maximal bond dimension to be \mathcal{D}_{\max} , the complexity of training on a dataset containing $|\mathcal{T}|$ samples is $O(|\mathcal{T}|N\mathcal{D}_{\max}^3)$, while the scaling for generative sampling is $O(N\mathcal{D}_{\max}^3)$.

1. Theoretical Understanding of the Expressive Power

The expressibility of MPS was intensively studied in the context of quantum physics. The bond dimensions of the MPS put an upper bound on its ability of capturing entanglement entropy. These solid theoretical understandings of the representational power of MPS [15, 16] makes it an appealing model for generative tasks.

In virtue of the success of MPS for quantum systems, we expect a polynomial scaling of the computational resources for datasets with short-range correlations. Treating dataset of two dimensional images using MPS is analogous to the application of DMRG to two dimensional quantum systems [29]. Although in principle an exact representation of the image dataset may require an exponentially large bond dimensions as the image resolution increases, at computationally affordable bond dimensions the MPS may already serve as a good approximation which captures salient features of the distribution.

2. Adaptive Adjustment of Expressibility

Performing optimizations for the two-site tensor (7), rather than for each tensor individually, allows one to dynamically adjust the bond dimensions during the learning process. Since for realistic datasets the required bond dimensions are likely to be inhomogeneous, adjusting them dynamically allocates computational resources in an optimal manner. This situation will be illustrated clearly using the MNIST data set in Sec. III C, and in Fig. 5.

Adjustment of the bond dimensions follows the distribution of singular values in (12), which is related to the low entanglement inductive bias of the MPS representation. Adaptive adjustment of MPS is advantageous compared to most other generative models. Because in most cases, the architecture (which is the main limiting factor of the expressibility of the model) is fixed during the learning procedure, only the parameters are tuned. By adaptively tuning the bond dimensions, the representational power of MPS can grow as it gets more acquainted with the training data. In this sense, adaptive adjustment of expressibility is analogous to the structural learning of the probabilistic graphical models, which is, however, a challenging task due to discreteness of the structural information.

3. Efficient Computation of Exact Gradients

Another advantage of MPS compared to the standard energy based generative model is that training can be done with high efficiency. The two terms contributing to the gradient (8) are analogous to the negative and positive phases in the training of an energy based models training [38], where the visible variables are unclamped and clamped respectively. In the energy based models, such as RBM, typical evaluation of the first term requires approximated MCMC sampling [42], or sophisticated mean-field approximations e.g. Thouless-Anderson-Palmer equations [43]. Luckily, the normalization factor and its gradient can be calculated exactly and straightforwardly for MPS.

4. Efficient Direct Sampling

The *Zipper* approach introduced in Sec. II C allows direct sampling from the learnt probability distribution. This completely avoids the slowing mixing problem in the MCMC sampling of energy based models.

It is known that the inference of graphical models of one dimensional (or tree) structure can be done efficiently [44, 45]. However, these structural constraints also limit the application range of graphical models with tractable partition functions. In this respect, the MPS enjoys the advantages of efficient direct sampling and can systematically increase its expressive power by increasing the bond dimensions. Thus, although the *Zipper* algorithm is formally similar to the sampling of autoregressive models [6, 7], being able to dynamically tune its expressibility makes the MPS a more flexible generative model. Moreover, unlike GAN [11] or VAE [5], the MPS can give explicit and tractable probability density of the samples, which may enable more unsupervised learning tasks. Moreover, the sampling in MPS work with arbitrary prior information of samples, such as fixed bits, in the applications as image reconstruction and de-noising. We note that this offers an advantage over the popular GAN, which easily maps a random vector in the latent space to the image space, but having difficulties in the reverse way — mapping a vector in the images space to the latent space as a prior information to sampling.

III. APPLICATIONS

In this section, to demonstrate the ability and features of the MPS generative modeling, we apply it to several standard datasets. As a proof of principle, we first apply our method to the toy data set of *Bars and Stripes* (BS), where some properties of our model can be characterized analytically. Then we train MPS as an associative memory to learn random binary patterns to study properties such as capacity and length dependences. Finally we train our model on the *Modified National Institute of Standards and Technology database* (MNIST) for displaying ability of our model on storing, generating and reconstructing of images of handwritten digits [46].

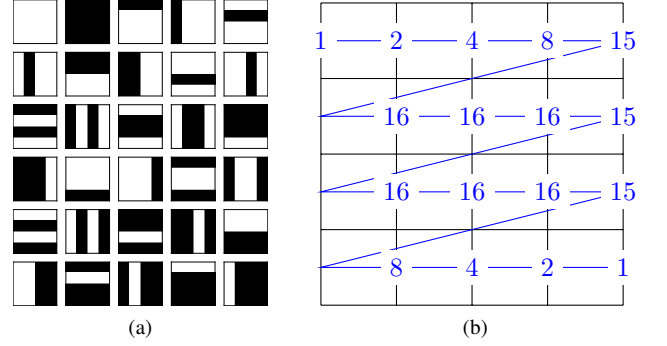


FIG. 1: (a) The Bars and Stripes dataset. (b) Ordering of the pixels when transforming the image into one dimensional vector. The numbers between pixels indicate the bond dimensions of the converged MPS.

A. Bars and Stripes

Bars and Stripes (BS) [47] is a data set containing 4×4 binary images. Each image has either four-pixel-length vertical bars or horizontal stripes, but not both. In total there are 30 different images in the dataset out of all 2^{16} possible ones, as shown in Fig. 1(a). These images appear with equal probability in the dataset. For the unsupervised generative modeling, training images do not come with labels. This toy problem allows a detailed analysis and reveals key characteristics of the MPS probabilistic model.

To use MPS for modeling, we reshape the 4×4 images into one dimensional vectors as shown in Fig. 1(b). After being trained over 4 loops, in each of which there are 10 steps of batch gradient descent training ($n_b = 1$) performed on each bond, with the hyperparameters being: cutoff = 0.0005, $\eta = 0.1$, the cost function converges to its minimum value, which equals to the Shannon entropy of the BS dataset $\ln(30)$, within an accuracy of 1×10^{-10} . Here what the MPS has accomplished is memorizing the thirty images rigidly, by increasing the probability of the instances appeared in the dataset, and suppressing the probability of not-shown instances towards zero. We have checked that the result is insensitive to the choice of hyperparameters.

The bond dimensions of the converged MPS have been annotated in Fig. 1(b). It is clear that part of the symmetry of the data set have been preserved in the learnt MPS. For instance, the 180° rotation around the center and the transposition of the second and the third rows. Open boundary condition results in the decrease of bond dimensions at both ends. In fact when conducting SVD at bond k , there are at most $2^{\min(k, N-k)}$ non-zero singular values because of the size of the merged matrix. In addition, the turnings bonds have slightly smaller bond dimension ($\mathcal{D}_4 = \mathcal{D}_8 = \mathcal{D}_{12} = 15$) than others inside the second row and the third row, which can be explained qualitatively as these bonds carrying less entanglement than the bonds in the bulk.

One can directly write down the exact “quantum wave function” of the BS dataset, which has a finite and constant am-

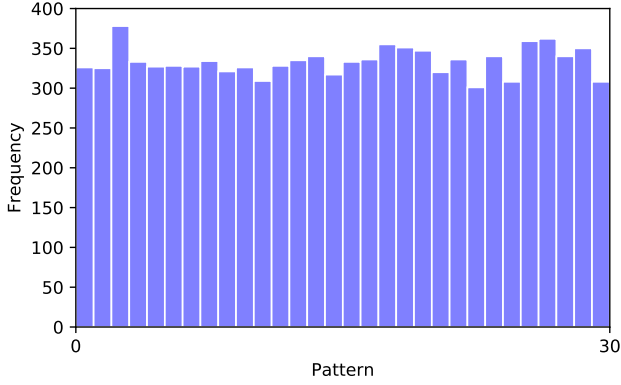


FIG. 2: Histograms of the samples generated with *Zipper* approach. The patterns are in the same order as in Fig. 1(a).

plitude for the training images and zero amplitude for other images. For each division on a bond, one can construct the reduced density matrix whose eigenvalues are the square of the singular values. Analyzed in this way, it is confirmed that the trained MPS achieve the minimal number of required bond dimension to exactly describe the BS dataset. In fact, the converged MPS takes positive values on all the training data. Combined with the fact that the distribution is actually flat for all training images, it is equivalent to think we are representing the probability distribution using MPS directly, which is related to the RBM representations of the dataset [32].

We have generated 10000 independent samples from the learned MPS using the *Zipper* approach. All the generated samples are training images as shown in Fig. 1(a). In Fig. 2 we plot the frequency of the generated images, which indeed demonstrates that the *Zipper* approach correctly samples from the uniform distribution. Note that in contrast to our model, for the energy based model one typically has to resort to MCMC method for sampling new patterns. This suffers from slow mixing problem since various patterns in the BS dataset differs substantially and it requires many MCMC steps to obtain one independent pattern.

As we will see in the next two subsections, if we constraint the bond dimension from growing to the required value to fully learn about the dataset, the MPS will try to learn about the most crucial feature of the dataset ranked by the singular values of the Eq. (12).

B. Random patterns

Capability of the model represents how well the training data are learned by the model. Usually this is evaluated using randomly generated patterns, for example in computing capacity of the classic Hopfield model [12], a pairwise interaction model with interactions given by the Hebb's rule. It has been shown [48] (using the replica method) that with $N \rightarrow \infty$ variables (i.e. at the thermodynamic limit), at a low-temperature the Hopfield model is in the retrieval phase where at most $|\mathcal{T}|_c = 0.14N$ random binary patterns can be remembered, in the sense that sample generated by the model have a

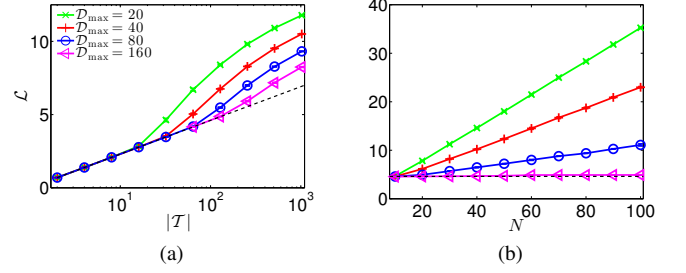


FIG. 3: NLL averaged as a function of: (a) number of random patterns used for training, with system size $N = 20$. (b) system size N , trained using $|\mathcal{T}| = 100$ random patterns. In both (a) and (b), different symbols correspond to different values of maximal bond dimension (\mathcal{D}_{\max}). Each data point is averaged over 20 random instances (i.e. sets of random patterns), error bars are also plotted, although they are much smaller than symbol size. The black dashed lines in figures denote $\mathcal{L} = \ln |\mathcal{T}|$.

large overlap with one of the training pattern. If the number of patterns in the Hopfield model is larger than $|\mathcal{T}|_c$, at a low-temperature the model would enter the spin glass state where samples generated by the model are not correlated with any training pattern.

Thanks to the tractable evaluation of the partition function Z in the MPS, we are able to evaluate exactly the likelihood, i.e. probability of a training pattern. Thus the capability of the model can be easily characterized by the mean negative log-likelihood \mathcal{L} . In this section we focus on the behavior of \mathcal{L} with varying number of training samples and varying system sizes.

In Fig. 3 (a) we plot \mathcal{L} as a function of number of patterns used for training for several maximal bond dimension \mathcal{D}_{\max} . From the figure we can see that with a small number of training patterns, we obtain $\mathcal{L} = \ln |\mathcal{T}|$. As what has been shown in the previous section, this means that all training patterns are remembered exactly, hence generation of the model is identical to one of the training samples. With number of training patterns increases, MPS with a fixed \mathcal{D}_{\max} will eventually fail in remembering exactly all the training patterns, resulting to $\mathcal{L} > \ln |\mathcal{T}|$. In this regime generations of the model usually deviate from training patterns (as illustrated in Fig. 4 on the MNIST dataset). We notice that with $|\mathcal{T}|$ increasing, the curves in the figure deviate from $\ln |\mathcal{T}|$ smoothly and continuously. We note this is very different from the Hopfield model where the overlap between the generation and training samples changes abruptly due to the first order transition from the retrieval phase to spin glass phase.

Figure 3 (a) also shows that a larger \mathcal{D}_{\max} enables MPS to remember exactly more patterns, and produce smaller \mathcal{L} with the number of patterns N fixed. This is quite natural because increasing \mathcal{D}_{\max} amounts to increasing number of parameters of the model, hence increases capability of the model. In principle if $\mathcal{D}_{\max} = \infty$ our model has infinite capability, since arbitrary quantum states can be decomposed into MPS [15].

Clearly this is an advantage of our model over the Hopfield model and inverse Ising model, whose maximum model capacity is proportional to system size.

Careful readers may complain that the inverse Ising model may be not the correct model to compare with, and the inverse Ising models with hidden variables, i.e. Boltzmann machines, do have infinite representation power. Actually this is exactly our point of using MPS as a unsupervised model: increasing bond-dimensions of tensors in MPS has similar functions as increasing number of hidden variables in other generative models.

In Fig. 3(b) we plot \mathcal{L} as a function of system size N , trained on $|\mathcal{T}| = 100$ random patterns. From the figure we can see that with \mathcal{D}_{\max} fixed \mathcal{L} increases linearly with system size N , which indicates that our model gives worse capability with a larger system size. This is due to the fact that keeping joint-distribution of variables becomes more and more difficult for MPS when the number of variables increases. A simple example is that the correlation between two arbitrary variable decays quickly as distance between two variables increases. This is actually a drawback of our model when compared with pair-wise models such as the inverse Ising model, which is the maximum-entropy model given mean and variance of the probability distribution of training samples, hence is able to capture long-distance correlations of the training data easily. Fortunately Fig. 3 (b) also shows that the decay of capability with system size can be compensated by increasing \mathcal{D}_{\max} , which reflects the universality approximation capability of MPS without any constraint on \mathcal{D}_{\max} [15].

C. MNIST data set

In this subsection we perform generative modeling experiment on the MNIST dataset [49]. Since MNIST images are in grayscale, we first turn them into binary numbers by random binarization, then flatten the images row by row into a vector as what was done for the BS dataset in Sec. III A. For the purpose of unsupervised generative modeling we do not need to use the labels of the digits.

Firstly we randomly selected $|\mathcal{T}| = 100$ samples from the MNIST dataset and trained the MPS with different maximal bond dimensions \mathcal{D}_{\max} . The results are shown in Fig. 4. The images in the insets are random generations by direct sampling from the MPS at different NLL values (denoted by the arrows). It is shown that with \mathcal{D}_{\max} increasing, \mathcal{L} converges gradually to its minimum $\ln(|\mathcal{T}|)$, and generations become more and more clear. It is interesting to see that with a relatively small maximum bond dimension, e.g. $\mathcal{D}_{\max} = 100$, although the handwriting is not as clear as with a large bond dimension, we can see that some crucial features, such as loops in image with label “6”, and the corner in image with label “7”, have already been reconstructed. When the maximum bond dimension increases to a larger value, the MPS is able to learn clear character of an image such as the pen strokes, hence we can say that the MPS has learned many “prototypes”. It has recently been reported that the feature-to-prototype transition in pattern recognitions can be observed

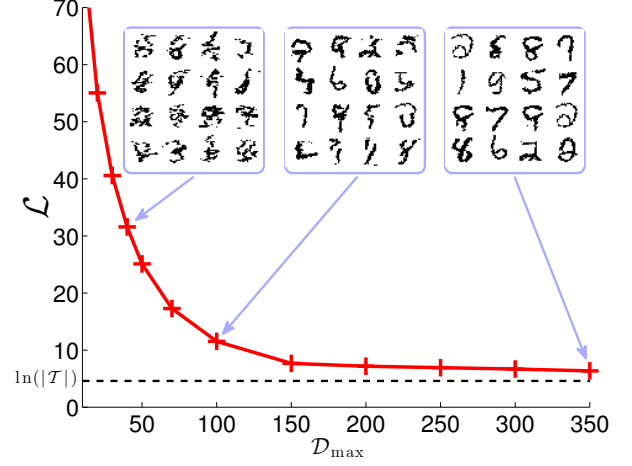


FIG. 4: The mean negative log-likelihood of a MPS trained using $|\mathcal{T}| = 100$ MNIST images of size 28×28 , with varying maximum bond dimension \mathcal{D}_{\max} . In all experiments, parameters of learning are set to cutoff = 1×10^{-6} , $n_b = 1$, $\eta = 0.001$. Each MPS is trained over at most 200 loops, the training is terminated if difference of \mathcal{L} between consecutive loops is less than 0.002. The horizontal dashed line indicates the Shannon entropy of the training dataset $\ln(|\mathcal{T}|)$, which is also the minimum value of \mathcal{L} on the training data.

by using a many-body interaction in the Hopfield model, or equivalently using an higher-order rectified polynomial activation function in the deep neural networks [50]. It is interesting to remark that in our model this can be achieved by simply adjusting the maximum bond dimension of the MPS.

Secondly we randomly selected 1000 images from the MNIST database for training. We restricted the maximum bond dimension to $\mathcal{D}_{\max} = 800$. The first loop was carried out under the hyperparameter configuration of: cutoff = 0.3, $\eta = 0.05$, $n_b = 10$, and the following 150 loops came with: cutoff = 1×10^{-7} , $\eta = 1 \times 10^{-3}$, $n_b = 20$. On each bond we performed 10 steps of stochastic gradient descent, c.f. Eq. (11). After 251 loops of training, the NLL on the training dataset reached 16.8, and many bond dimensions had reached \mathcal{D}_{\max} . Figure 5 shows the distribution of bond dimensions where we can see that large bond dimensions concentrated in the center of the image, where the variation of the pixels are the largest. The bond dimensions around the top and bottom edge of the image remained small, because those pixels are always inactivated in the training samples. They carry no information and has no correlations with the remaining part of the image. Interestingly, although the pixels on the left and right edges are also white, they have larger bond dimensions because these bonds learned to mediate the correlations between the rows of the images.

The samples directly generated after training are shown in Fig. 6(a). We also show a few original samples from the training set in Fig. 6(b) for comparison. Although many of the generated images cannot be recognized as digits, some as-

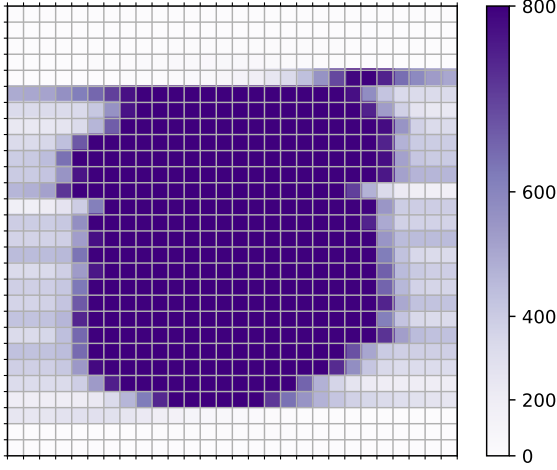


FIG. 5: Bond dimensions of the learnt MPS. Each pixel in this figure corresponds to bond dimension of the right leg of the tensor associated to the identical coordinate in the original image.

pects of the result are worth mentioning. Firstly, the MPS learnt to leave margin blank with width 4 pixels, which is the most obvious common feature in MNIST database. Secondly, the activated pixels compose kinds of pen strokes that can be extracted from the digits. Finally, a few of the samples already be recognized as digits. We expect as one increases the maximal bond dimension and keeps on training the MPS will produce more realistic images. Unlike the discriminative learning task carried out in [30], it seems we need to use much larger bond dimensions to achieve a good performance in the unsupervised learning. We think the reason is that in the classification task, local features of an image are used for predicting label of the image. Those local features, such as presence of loop for label “6”, usually span a shorter range of pixels, thus do not require MPS to remember longer-range correlation between pixels. However it is necessary for generative modeling because learning the joint distribution from the

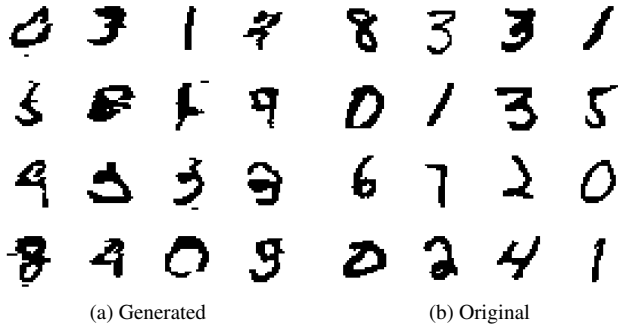


FIG. 6: (a) Images generated from a MPS trained on the 1000-image training set over 251 loops, achieving a final average NLL 16.8. (b) Original images randomly selected from the training set.

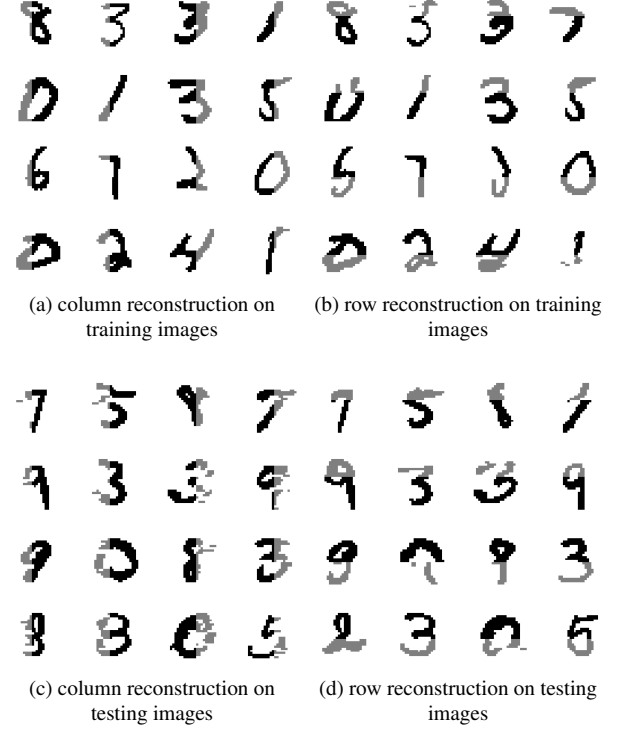


FIG. 7: Image reconstruction from partial images by direct sampling of a MPS that has been trained on the 1000-image training set over 251 loops. (a,b) Restoration of images in the training set, e.g. those shown in Fig. 6(b). (c,d) Reconstruction of 16 images chosen from the test set. The test set contains images from the MNIST database that were not used for training. The given parts are in black and the reconstructed parts are in gray. The reconstructed parts are: (a,c) 12 columns from either the left or the right; (b,d) 12 rows from either the top or the bottom.

data consists of (but not limited to) learning two-point correlations between pairs of variables that could be far from each other.

Thirdly we carried out image restoration experiment on the MPS trained on 1000 images. As shown in Fig. 7 we first remove part of images from the samples in Fig. 6 (b) then reconstruct ungiven pixels (in gray) using direct sampling conditioning on given parts. For column reconstruction, its performance is remarkable. The reconstructed images in Fig. 7(a) are almost identical with the original ones in Fig. 6(b). On the other hand, for row reconstruction in Fig. 7(b), it made interesting but reasonable deviations. For instance, the rightmost on the first row, an “1” has been bent to “7”. We also checked its ability to reconstruct MNIST images other than the training images, as shown in Fig. 7(c,d). These indicates that the MPS has learned crucial features of the dataset, rather than merely memorizing the training instances. In fact, even as early as when trained over only 11 loops, the MPS could perform column reconstruction with similar image quality, but its row reconstruction performance was much worse than later when

trained over 251 loops. It is reflected that the MPS has learned about short range patterns within each row earlier than those with long range correlations between different rows, since the images have been flattened into a one dimensional vector row by row.

IV. SUMMARY AND OUTLOOK

We have presented a tensor-network-based unsupervised model aims at modeling the probability distribution of samples in given unlabeled data. The representation of probability is structured on matrix product state, which brings a number of advantages in learning and sampling.

We assess performance by the NLL averaged on the training set since we focus on the associative memory application of the MPS generative model in this paper. In more general unsupervised applications, considerations such as overfitting and regularization certainly apply to the MPS model as well. One can employ either the early stopping or using a regularization term to prevent from overfitting. In this sense, the maximal bond dimension plays a role of regularization as it prevents the MPS learn about too intricate correlations [30].

Similar to the study of quantum states using TN, the required resources to model a probability distribution of the dataset is related to the complexity of the dataset. In this way, ideas and insights from the study of quantum many-body physics can be transferred to generative modeling together with those techniques.

A crucial issue is the sign redundancy of the tensor network states when used as a probabilistic model. Noted that besides the usual gauge redundancy of the MPS, the sign of the MPS is also redundant in the probabilistic modeling. Since we did not fix the sign of MPS, it is likely that during the optimization it develops different signs for different configurations. This sign fluctuation may unnecessarily increase the entanglement in the MPS and therefore the bond dimension [51]. In the future it worth exploring approaches to restrict the sign of MPS during the optimization. If that is possible, one may alternatively model the probability using MPS, other than the MPS square. In this way, the MPS representation of the probability has stronger and more direct connection to the RBM [32]. Alternatively, to select the representation towards the low entanglement one, one can add a penalty term in the target function, for instance, a term proportional to Rényi entanglement entropy.

The binary data links closely with the quantum spin-1/2 system. To model continuous variables one can follow [30], using a local feature map to lift the continuous variable to a

spinor space. The capacity and efficiency of this approach may also depend on the specific way of performing the mapping, so in terms of continuous input there are still a lot to be explored on this algorithm. Moreover, for colored images one can envision to encode the RGB values as three physical legs of the MPS tensor.

Similar to using MPS for studying two-dimensional quantum lattice problems [29], modeling images with MPS faces the problem of introducing long range correlations for some neighboring pixels in two dimension. An obvious generalization of the present approach is to use more expressive TN with more complex structures. In particular, the projected entangled pair states (PEPS) [52] is particularly suitable for images, because it takes care of correlation between pixels in two dimensional space. Similar to studies of quantum systems in 2D, however, this advantage of PEPS is partially compensated by the difficulty of contracting the network and losing the convenient canonical forms. Exact contraction of a PEPS is #P hard [53]. Nevertheless, one can employ tensor renormalization group methods for approximated contraction of PEPS [54–57]. Thus, it remains to be seen whether judicious combination of these techniques really deliver a better performance as a generative model.

After all, we would like to remark that perhaps the most exciting feature of the quantum-inspired generative models is the possibility of being implemented by a quantum device [58], rather than being simulated in the classic computers. In that way, neither the large bond dimension nor the high computational complexity of tensor contraction, would be a problem. The tensor network representation of probability densities may facilitate this quantum generative modeling because some of the tensor network states can be prepared efficiently on a quantum computer [59].

ACKNOWLEDGMENTS

We thank Liang-Zhu Mu, Ze-Yang Li, Hong-Ye Hu, Dian Wu, Zi-Zhao Han, Song Cheng, Jing Chen, Wei Li and E. Miles Stoudenmire for inspiring discussions. P.Z. acknowledges Swarna Club workshop on “Geometry, Complex Network and Machine Learning” sponsored by Kai Feng Foundation in 2016. L.W. is supported by the Ministry of Science and Technology of China under the Grant No. 2016YFA0300603 and National Natural Science Foundation of China under the Grant No. 11774398. J.W. is supported by National Training Program of Innovation for Undergraduates. Part of the computation is carried out at the High Performance Computational Cluster of ITP, CAS.

-
- [1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, “Deep learning,” *Nature* **521**, 436–444 (2015).
 - [2] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski, “A Learning Algorithm for Boltzmann Machines,” *Cognitive Science* **9**, 147–169 (1985).
 - [3] P Smolensky, “Information processing in dynamical systems:

- foundations of harmony theory,” in *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1* (1986) pp. 194–281.
- [4] Ruslan Salakhutdinov, “Learning deep generative models,” *Annual Review of Statistics and Its Application* **2**, 361–385 (2015).
- [5] Diederik P Kingma and Max Welling, “Auto-encoding varia-

- tional bayes,” [arXiv:1312.6114](#) (2013).
- [6] Benigno Uribe, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo Larochelle, “Neural autoregressive distribution estimation,” *Journal of Machine Learning Research* **17**, 1–37 (2016).
- [7] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu, “Pixel recurrent neural networks,” in *Proceedings of The 33rd International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 48 (PMLR, New York, New York, USA, 2016) pp. 1747–1756.
- [8] Laurent Dinh, David Krueger, and Yoshua Bengio, “NICE: non-linear independent components estimation,” [arXiv:1410.8516](#) (2014).
- [9] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio, “Density estimation using real NVP,” [arXiv:1605.08803](#) (2016).
- [10] Danilo Rezende and Shakir Mohamed, “Variational inference with normalizing flows,” in *Proceedings of the 32nd International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 37 (PMLR, Lille, France, 2015) pp. 1530–1538.
- [11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems 27* (Curran Associates, Inc., 2014) pp. 2672–2680.
- [12] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities,” (1982) p. 2554.
- [13] Hilbert J. Kappen and Francisco de Borja Rodríguez Ortiz, “Boltzmann machine learning using mean field theory and linear response correction,” in *Advances in Neural Information Processing Systems 10* (MIT Press, 1998) pp. 280–286.
- [14] Y. Roudi, J. Tyrcha, and J. Hertz, “Ising model for neural data: Model quality and approximate methods for extracting functional connectivity,” *Phys. Rev. E* **79**, 051915 (2009).
- [15] Ulrich Schollwöck, “The density-matrix renormalization group in the age of matrix product states,” *Annals of Physics* **326**, 96–192 (2011).
- [16] Román Orús, “A practical introduction to tensor networks: Matrix product states and projected entangled pair states,” *Annals of Physics* **349**, 117 – 158 (2014).
- [17] M B Hastings, “An area law for one-dimensional quantum systems,” *Journal of Statistical Mechanics: Theory and Experiment* **2007**, P08024–P08024 (2007).
- [18] J. Eisert, M. Cramer, and M. B. Plenio, “Colloquium: Area laws for the entanglement entropy,” *Reviews of Modern Physics* **82**, 277–306 (2010), [arXiv:0808.3773](#).
- [19] Johann A. Bengua, Ho N. Phien, and Hoang Duong Tuan, “Optimal feature extraction and classification of tensors via matrix product state decomposition,” [arXiv:1503.00516](#) (2015).
- [20] Alexander Novikov, Anton Rodomanov, Anton Osokin, and Dmitry Vetrov, “Putting mrfs on a tensor train,” in *International Conference on Machine Learning* (2014) pp. 811–819.
- [21] Alexander Novikov, Dmitrii Podoprikin, Anton Osokin, and Dmitry P Vetrov, “Tensorizing neural networks,” in *Advances in Neural Information Processing Systems* (2015) pp. 442–450.
- [22] Nadav Cohen, Or Sharir, and Amnon Shashua, “On the expressive power of deep learning: A tensor analysis,” [arXiv:1509.05009](#) (2015).
- [23] Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao, Danilo P Mandic, *et al.*, “Tensor networks for dimensionality reduction and large-scale optimization: Part I low-rank tensor decompositions,” *Foundations and Trends® in Machine Learning* **9**, 249–429 (2016).
- [24] Y. Levine, D. Yakira, N. Cohen, and A. Shashua, “Deep Learning and Quantum Entanglement: Fundamental Connections with Implications to Network Design,” [arXiv:1704.01552](#) (2017).
- [25] D Perez-Garcia, F Verstraete, M M Wolf, and J I Cirac, “Matrix Product State Representations,” *Quantum Info. Comput.* **7**, 401–430 (2007).
- [26] Ivan V Oseledets, “Tensor-train decomposition,” *SIAM Journal on Scientific Computing* **33**, 2295–2317 (2011).
- [27] Zeph Landau, Umesh Vazirani, and Thomas Vidick, “A polynomial time algorithm for the ground state of 1d gapped local hamiltonians,” *Nature Physics* **11**, 566 (2015).
- [28] Steven R. White, “Density matrix formulation for quantum renormalization groups,” *Phys. Rev. Lett.* **69**, 2863–2866 (1992).
- [29] E.M. Stoudenmire and Steven R. White, “Studying two-dimensional systems with the density matrix renormalization group,” *Annual Review of Condensed Matter Physics* **3**, 111–128 (2012).
- [30] E. Miles Stoudenmire and David J. Schwab, “Supervised Learning with Quantum-Inspired Tensor Networks,” *Advances in Neural Information Processing Systems* **29**, 4799 (2016), [arXiv:1605.05775](#).
- [31] Alexander Novikov, Mikhail Trofimov, and Ivan Oseledets, “Exponential machines,” [arXiv:1605.05775](#) (2016).
- [32] Jing Chen, Song Cheng, Haidong Xie, Lei Wang, and Tao Xiang, “On the Equivalence of Restricted Boltzmann Machines and Tensor Network States,” [arXiv:1701.04831](#) (2017).
- [33] Angel J Gallego and Roman Orus, “The physical structure of grammatical correlations: equivalences, formalizations and consequences,” [arXiv:1708.01525](#) (2017).
- [34] Andrew J. Ferris and Guifre Vidal, “Perfect sampling with unitary tensor networks,” *Phys. Rev. B* **85**, 165146 (2012).
- [35] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning* (MIT Press, 2016) <http://www.deeplearningbook.org>.
- [36] S. Kullback and R. A. Leibler, “On Information and Sufficiency,” *The Annals of Mathematical Statistics* **22**, 79–86 (1951).
- [37] M. Born, “Zur Quantenmechanik der Stoßvorgänge,” *Zeitschrift für Physik* **37**, 863–867 (1926).
- [38] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang, “A tutorial on energy-based learning,” (2006), <http://yann.lecun.com/exdb/publis/orig/lecun-06.pdf> (Accessed on 2-September-2017).
- [39] Ming-Jie Zhao and Herbert Jaeger, “Norm-observable operator models,” *Neural Computation* **22**, 1927–1959 (2010), pMID: 20141473, <https://doi.org/10.1162/neco.2010.03-09-983>.
- [40] Raphael Bailly, “Quadratic weighted automata: spectral algorithm and likelihood maximization,” in *Proceedings of the Asian Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 20, edited by Chun-Nan Hsu and Wee Sun Lee (PMLR, South Garden Hotels and Resorts, Taoyuan, Taiwan, 2011) pp. 147–163.
- [41] Setting $\mathcal{D}_k = 1$ for all bonds makes the bond dimension difficult to grow in the initial training phase. Since the rank of two site tensor is $1 \times 2 \times 2 \times 1$ and the number of nonzero singular value is at most 2, which is likely to be truncated back to $\mathcal{D}_k = 1$ with small cutoff.
- [42] Geoffrey E Hinton, “A practical guide to training restricted boltzmann machines,” in *Neural networks: Tricks of the trade* (Springer, 2012) pp. 599–619.
- [43] Marylou Gabrie, Eric W Tramel, and Florent Krzakala, “Training restricted boltzmann machine via the thouless-anderson-palmer free energy,” in *Advances in Neural Information Pro-*

- cessing Systems* 28, edited by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Curran Associates, Inc., 2015) pp. 640–648.
- [44] Martin J Wainwright and Michael I Jordan, “Graphical models, exponential families, and variational inference,” *Foundations and Trends® in Machine Learning* **1**, 1–305 (2008).
 - [45] Daphne Koller and Nir Friedman, *Probabilistic Graphical Models, Principles and Techniques* (The MIT Press, 2009).
 - [46] The code of these experiments have been posted at <https://github.com/congzlwag/UnsupGenModbyMPS>.
 - [47] David JC MacKay, *Information theory, inference and learning algorithms* (Cambridge University Press, 2003).
 - [48] D.J. Amit, H. Gutfreund, and H. Sompolinsky, “Spin-glass models of neural networks,” *Phys. Rev. A* **32**, 1007 (1985).
 - [49] Yann LeCun, Corinnai Cortes, and Christopher J.C. Burges, “The mnist database of handwritten digits,” <http://yann.lecun.com/exdb/mnist> (Accessed on 2-September-2017).
 - [50] Dmitry Krotov and John J Hopfield, “Dense associative memory for pattern recognition,” in *Advances in Neural Information Processing Systems* (2016) pp. 1172–1180.
 - [51] Yi Zhang, Tarun Grover, and Ashvin Vishwanath, “Entanglement entropy of critical spin liquids,” *Phys. Rev. Lett.* **107**, 067202 (2011).
 - [52] Frank Verstraete and J Ignacio Cirac, “Renormalization algorithms for quantum-many body systems in two and higher dimensions,” *arXiv preprint cond-mat/0407066* (2004).
 - [53] Norbert Schuch, Michael M. Wolf, Frank Verstraete, and J. Ignacio Cirac, “Computational complexity of projected entangled pair states,” *Phys. Rev. Lett.* **98**, 140506 (2007).
 - [54] Michael Levin and Cody P. Nave, “Tensor renormalization group approach to two-dimensional classical lattice models,” *Phys. Rev. Lett.* **99**, 120601 (2007).
 - [55] Z. Y. Xie, H. C. Jiang, Q. N. Chen, Z. Y. Weng, and T. Xiang, “Second renormalization of tensor-network states,” *Phys. Rev. Lett.* **103**, 160601 (2009).
 - [56] Chuang Wang, Shao-Meng Qin, and Hai-Jun Zhou, “Topologically invariant tensor renormalization group method for the edwards-anderson spin glasses model,” *Phys. Rev. B* **90**, 174201 (2014).
 - [57] G. Evenbly and G. Vidal, “Tensor network renormalization,” *Phys. Rev. Lett.* **115**, 180405 (2015).
 - [58] 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 e-prints* (2017), [arXiv:1708.09757](https://arxiv.org/abs/1708.09757).
 - [59] Martin Schwarz, Kristan Temme, and Frank Verstraete, “Preparing projected entangled pair states on a quantum computer,” *Phys. Rev. Lett.* **108**, 110502 (2012).