Distributive Pre-training of Generative Modeling Using Matrix Product States

Sheng-Hsuan Lin [*]
Department of Physics
Technische Universität München
85748 Garching, Germany

Olivier Kuijpers
Department of Physics
Technische Universität München
85748 Garching, Germany

Sebastian Peterhansl
Department of Physics
Technische Universität München
85748 Garching, Germany

Frank Pollmann
Department of Physics
Technische Universität München
85748 Garching, Germany

Abstract

Tensor networks have recently found applications in machine learning for both supervised learning and unsupervised learning. The most common approaches for training these models are gradient descent methods. In this work, we consider an alternative training scheme utilizing basic tensor network operations, e.g., summation and compression. The training algorithm is based on compressing the superposition state constructed from all the training data in product state representation. The algorithm could be parallelized easily and only iterates through the dataset once. Hence, it serves as a pre-training algorithm. We benchmark the algorithm on the MNIST dataset and show reasonable results for generating new images and classification tasks. Furthermore, we provide an interpretation of the algorithm as a compressed quantum kernel density estimation for the probability amplitude of input data.

1 Introduction

Machine learning and many-body physics have great similarity in the studies of finding low-dimensional and meaningful representations over exponentially large degrees of freedom [1, 2]. Tensor networks have been proven to be efficient representations for quantum many-body systems with low entanglement [3, 4]. With numerical algorithms, tensor network states (TNSs) are parameterized models that could be optimized variationally to solve many-body problems [5, 6], e.g. searching the ground state. As parameterized models, TNSs have attracted interest in recent years for solving supervised learning problems [7] and also unsupervised learning problems [8]. Being a promising method, tensor networks may also give us insight into machine learning problems.

In this work, we propose a new pre-training algorithm for models based on matrix-product states (MPSs) for unsupervised generative modeling and supervised learning on the MNIST dataset [9]. The algorithm is an iterative compression over the summation of quantum states [2] encoding the input data which could be parallelized as a tree-based reduction algorithm and run distributively.

[*]shenghsuan.lin@tum.de

[2]The uncompressed superposition states are considered previously by Martyn et al. [10], where they showed the uncompressed states form good models but concluded these states are not relevant for tensor network models because of the high entanglement.

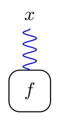
The main contributions of our article are: (i) We propose a simple pre-training algorithm for MPS models that could be parallelized and run distributively with potentially exponential speedup. (ii) We identify the combination of the orthonormal feature map [7] and the superposition state of all the quantum states encoding data as a quantum version of the kernel density estimation. (iii) We propose a sampling algorithm for continuous variables by viewing the combination of wavefunctions over discrete variables $\mathbf{s}$ and the orthonormal feature maps as quantum latent variable wavefunctions.

The remainder of the paper is organized as follows: We introduce the tensor network notation for continuous variables and review the idea of feature maps in Sec. 2.1. We discuss the MPS-based Born machines for both discrete and continuous variables in Sec. 2.2. The proposed pre-training algorithm is shown in Sec. 2.3. We show the result in Sec. 3 and discuss the implication in Sec. 4.

2 Method

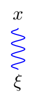
2.1 Tensor network notation and feature map

We introduce some non-conventional tensor network notation for continuous variables to facilitate the discussion. In the standard tensor network notation, a straight line is associated with a discrete index, for example s. Here, we represent a continuous degree of freedom, for example $x \in \mathbb{R}$ or $\mathbb{C}$, by a curly line. Similar to representing a vector in standard tensor network notation, the univariate function f is denoted by a rounded square box with a curly leg



$$= f(x). \tag{1}$$

By this convention, the Dirac-delta function $\delta(x-\xi)$ for $x, \xi \in \mathbb{R}$ is denoted by the curly line

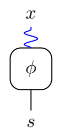


$$= \delta(x-\xi), \tag{2}$$

which is similar to the Kronecker delta represented by the straight line. Connecting the curly lines means taking the integral over the continuous degree of freedom by some measure $\mu(x)$. Similarly, connecting the straight lines represents a summation over the discrete degree of freedom.

We apply the same notation for the local feature map introduced by [7]. The local feature map is a vector-valued function defined as

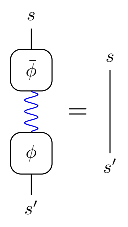
$$\phi^s(x) = \begin{pmatrix} \phi^{s=0}(x) \\ \vdots \\ \phi^{s=d-1}(x) \end{pmatrix} \in \mathbb{C}^d \qquad \text{denoted as}$$



$$= \phi^s(x). \tag{3}$$

A local feature map is orthonormal if it satisfies the following condition.

$$\int \bar{\phi}^s(x)\phi^{s'}(x)d\mu(x) = \delta_{ss'} \qquad \text{denoted as}$$

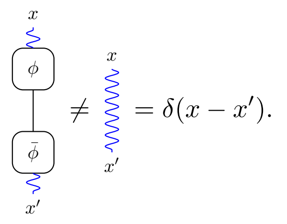


$$= \delta_{ss'} \tag{4}$$

A local feature map acting on a single continuous variable x creates a normalized discrete wavefunction if local feature map satisfies the condition

$$\sum_s |\phi^s(x)|^2 = 1 \tag{5}$$

Note that this does not imply and is different from the resolution of identity $\sum_s \phi^s(x)\bar{\phi}^s(x') = \delta(x-x')$. That is, in general,



(6)

This is because the resolution of identity only holds when the local feature map is a complete orthonormal basis set of functions. Such feature map would be infinite dimensional and is not considered in practice.

Here we give some examples of the known local feature maps:
Example 1: $\phi(x) = [\cos(\frac{\pi}{2}x), \sin(\frac{\pi}{2}x)],\ x \in [0,1]$ satisfies the condition in Eq. (5) but not the orthonormal condition in Eq. (4).

Example 2: $\phi(x) = [e^{i(3\pi/2)x}\cos(\frac{\pi}{2}x), e^{-i(3\pi/2)x}\sin(\frac{\pi}{2}x)], x \in [0,1]$ satisfies the condition in Eq. (5) and the orthonormal condition in Eq. (4).

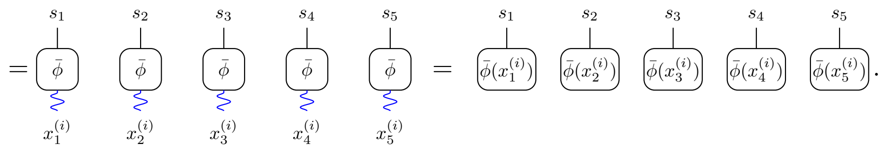
Example 3: $\phi(x) = [\mathrm{sgn}(x) - \mathrm{sgn}(x-0.5), \mathrm{sgn}(x-0.5) - \mathrm{sgn}(x-1)], x \in [0,1]$ satisfies the condition in Eq. (5) and the orthonormal condition in Eq. (4).

The (global) feature map over variables is often taken to be the tensor product of local feature maps.

$$\Phi^{\mathbf{s}}(\mathbf{x}) = \Phi^{s_1,s_2,\dots s_N}(\mathbf{x}) = \phi^{s_1}(x_1) \otimes \phi^{s_2}(x_2) \otimes \cdots \otimes \phi^{s_N}(x_N). \tag{7}$$

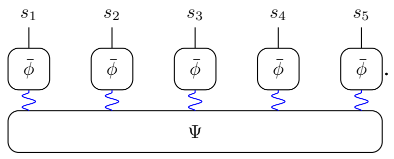
The feature map maps any single input data of continuous variables $\mathbf{x}^{(i)}$ to a normalized wavefunction $\Phi^{\mathbf{s}}(\mathbf{x}^{(i)})$ over the discrete variables $\mathbf{s}$ if the local feature map satisfies the condition in Eq. (5). We can re-express the description as

$$\bar{\Phi}^{\mathbf{s}}(\mathbf{x}^{(i)}) = \int \bar{\Phi}^{\mathbf{s}}(\mathbf{x})\delta(\mathbf{x}-\mathbf{x}^{(i)})d\mathbf{x} \tag{8}$$



That is we can think of the normalized wavefunction $\Phi^{\mathbf{s}}(\mathbf{x}^{(i)})$ on discrete variables $\mathbf{s}$ as the feature map $\bar{\Phi}^{\mathbf{s}}(\mathbf{x})$ acting on the input $\delta(\mathbf{x}-\mathbf{x}^{(i)})$. With the same formalism, one could also map any wavefunction $\Psi(\mathbf{x})$ over continuous variables $\mathbf{x}$ to a wavefunction $\Psi^{\mathbf{s}}$ over discrete variables $\mathbf{s}$ by

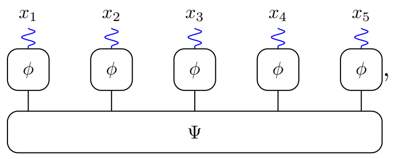
$$\int \bar{\Phi}^{\mathbf{s}}(\mathbf{x})\Psi(\mathbf{x})d\mu(\mathbf{x}) = \Psi^{\mathbf{s}} = \tag{9}$$



Note, however, the resulting wavefunction $\Psi^{\mathbf{s}}$ is not normalized since the local feature map is in general not the resolution of identity.

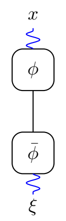
The feature map could map any normalized wavefunction $\Psi^{\mathbf{s}}$ to a normalized wavefunction $\Psi(\mathbf{x})$ by

$$\sum_{\mathbf{s}} \Phi^{\mathbf{s}}(\mathbf{x})\Psi^{\mathbf{s}} = \Psi(\mathbf{x}) = \tag{10}$$



if the orthonormal condition Eq. (4) holds. Examples of such orthonormal mappings are given above.

An important and elementary expression that will appear repeatedly is



$$= \sum_s \phi^s(x)\bar{\phi}^s(\xi) = \sum_s \int \phi^s(x)\bar{\phi}^s(x')\delta(x'-\xi)dx', \tag{11}$$

which can be interpreted as the projection of the Dirac-delta function at ξ on a finite basis set $\phi^s(x)$. Therefore, this expression represents an approximation or smoothing of Dirac-delta functions peaked at $x = \xi$ (see Fig. 2 in Appendix B) . We will see from this perspective that the summation of quantum states encoding the data points is related to the summation of smoothed Dirac-delta functions over the given data points.

2.2 Generative Models: Born Machine based on MPS and feature maps

Born machines: Quantum wavefunctions could be utilized as machine learning tools describing the probability distribution given by Born's rule, i.e., $P = |\psi|^2$. The Born machines [11] are parameterized quantum wavefunctions, e.g. tensor network states [8, 12] or parameterized quantum circuits [13], that are applied as generative models over discrete variables for learning the probability distributions of images. Born machines based on MPSs have the advantage that they give a tractable likelihood, i.e., the probability amplitude can be evaluated efficiently, and allows ancestral sampling [14].

Born machine based on MPS over continuous variables: In this work, we consider Born machines based on MPSs for both discrete and continuous variables. MPSs could also parameterize wavefunctions over continuous variables with the help of orthonormal feature maps [7]. We see this from Eq. (10) by replacing the $\Psi^{\mathbf{s}}$ by an MPS. Algorithms for training generative models of discrete variables also work for models with continuous variables [8]. Moreover, the combination of an MPS with a feature map not only defines a model over continuous variables $\mathbf{x}$, but it also gives an additional useful interpretation as a "quantum" latent variable model, where the latent variables are the discrete variables $\mathbf{s}$. Such an interpretation leads to a sampling algorithm for continuous variables.

Latent variable models: Latent variable models are defined with some latent variables $\mathbf{z}$ and the parametrized distributions $P_\theta(\mathbf{x}|\mathbf{z})$ and $P_\theta(\mathbf{z})$ such that the probability distribution $P(\mathbf{x})$ over the variables $\mathbf{x}$ can be expressed as $P(\mathbf{x}) = \sum_{\mathbf{z}} P_\theta(\mathbf{x}|\mathbf{z})P_\theta(\mathbf{z})$. Latent variable models can be efficiently sampled by first sampling the latent variables $\mathbf{z} \sim P_\theta(\mathbf{z})$ and then sampling the variables $\mathbf{x} \sim P_\theta(\mathbf{x}|\mathbf{z})$ given the latent variables $\mathbf{z}$.

Latent variable wavefunctions: Similarly, we can interpret the MPS with local feature maps as a latent variable wavefunction or a quantum latent variable model. The probability amplitude over the continuous variables $\mathbf{x}$ is given by the parameterized amplitudes $\Psi_\theta(\mathbf{x}|\mathbf{s})$ and $\Psi_\theta(\mathbf{s})$, where $\mathbf{s}$ are the discrete latent variables, e.g., the spins. The conditional probability amplitudes need to satisfy the normalization condition, i.e., $\int |\Psi_\theta(\mathbf{x}|\mathbf{s})|^2 d\mathbf{x} = 1,\ \forall\, \mathbf{s}$, which is automatically satisfied by the feature map $\Phi^{\mathbf{s}}(\mathbf{x})$ if the local feature map is orthonormal. In our setup, we consider a fixed conditional probability amplitude $\Psi(\mathbf{x}|\mathbf{s}) = \Phi^{\mathbf{s}}(\mathbf{x})$. As a result, we have $\Psi(\mathbf{x}) = \sum_{\mathbf{s}} \Phi(\mathbf{x}|\mathbf{s})\Psi_\theta(\mathbf{s})$.

Because the probability of a latent variable wavefunction defines a latent variable model, we can similarly sample the continuous variables $\mathbf{x}$ according to the probability distribution $|\Psi(\mathbf{x})|^2$: one can first sample the discrete variables $\mathbf{s}$ according to $\mathbf{s} \sim |\Psi_\theta(\mathbf{s})|^2$ and then sample the continuous variables $\mathbf{x}$ according to $\mathbf{x} \sim |\Psi_\theta(\mathbf{x}|\mathbf{s})|^2$.

2.3 Algorithm

Here, we propose a pre-training algorithm for MPS-based Born machines which provides good initialization for bond dimension χ MPSs that could be applied to generative modeling or supervised learning. We review the general setup of generative modeling in Appendix A. The motivation of the algorithm comes from the observation that a special wavefunction, dubbed the digit wavefunction or the sum state $|\Sigma_l\rangle$ [10], captures the data distribution and performs well in the classification task.

The algorithm is based on MPS compression algorithms over this wavefunction. The digit wavefunction $|\Sigma_l\rangle$ of digit l is defined as the sum over all the training data of digit l in product state representation:

$$|\Sigma_l\rangle = \Big(\sum_{i;y^{(i)}=l} |\Phi^{\mathbf{s}}(\mathbf{x}^{(i)})\rangle \Big)/C_{\text{Norm}}. \tag{12}$$

The C_{Norm} is a normalization constant. It is tempting to consider this state directly as the quantum state defining the probability of the corresponding digit. As pointed out by [10], these states have high entanglement and can not be approximated accurately using MPSs with low bond dimensions. Nevertheless, we consider the pre-training algorithm for MPS-based Born machine with bond dimension χ as finding the approximated compression of these states. We denote the uncompressed wavefunction as $|\Sigma_l\rangle$ and the compressed state with bond dimension χ MPS as $|\Sigma_l^{\chi}\rangle$. We find that although the compression to low bond dimension MPSs have only small overlaps with the original states, such compressed states are still useful and give reasonable results in both generative modeling and supervised learning tasks.

The state $|\Sigma_l\rangle$ can be constructed by summing up the product states utilizing MPS arithmetic [5], which leads to a block-diagonal sparse MPS. Naively one could first sum up all the product states, then variationally find the optimal truncated MPS of bond dimension χ. We call this method the direct compression.

$$|\Sigma_l\rangle = \frac{\sum_i \Phi^{\mathbf{s}}(\mathbf{x}^{(i)})}{C_{\text{Norm}}} \xrightarrow[\text{compression}]{\text{MPS}} |\Sigma_l^{\chi}\rangle \tag{13}$$

A more efficient, though approximated, method is the parallel compression. It is a parallel algorithm with the summation and compression together as a reduction operation. We first divide the data into N_{data}/χ batches, where each batch can be represented as an MPS of bond dimension χ exactly. Then we perform MPS summation and compression between batch 1 and batch 2, batch 3 and batch 4, etc, and obtain again in the end $N_{\text{data}}/(2 * \chi)$ batches of MPS with bond dimension χ. Repeating this process, we can sum up the states in tree-like (fan-in) fashion, which has $\log(N_{\text{data}})$ complexity in time provided enough computation resource.

There are several advantages of using MPSs [5]: (i) The addition of two MPSs with bond dimensions χ_1, χ_2 can be expressed exactly as an MPS with bond dimension $\chi = \chi_1 + \chi_2$. (ii) Given an MPS with higher bond dimension, there are efficient algorithms to find the optimal MPS with lower bond dimension to approximate the given MPS. Combining these two properties, the proposed learning algorithm could in principle be parallelized. This is similar to any standard parallel reduction algorithm where MPSs addition and compression are the reduction operation. This is drastically different to the training using stochastic gradient descend methods which are intrinsically serial and are hard to parallelize. The algorithm ends when it goes through the dataset exactly once and is suitable for distributive pre-training of generative models.

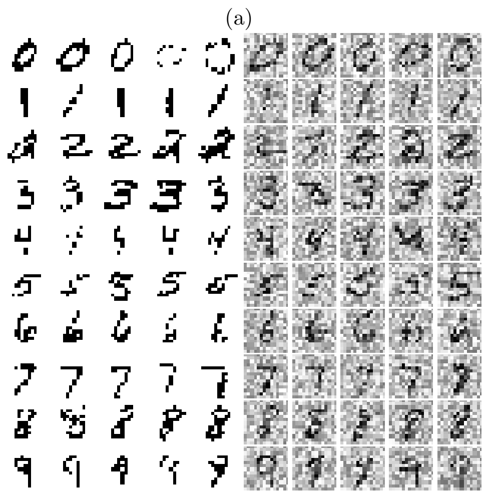
In the following, we provide an interpretation of the learning algorithm. Despite the simplicity of the learning process, we show that such algorithm indeed works in Sec. 3.

Interpretation: We first recall the kernel density estimation (KDE) method, which is a class of non-parametric approaches for density estimation. Given the empirical distribution $\hat{P}(\mathbf{x}) = (\sum_i \delta(\mathbf{x} - \mathbf{x}^{(i)}))/N_{\text{data}}$ from the data points, the model is constructed directly from the data points convolving with the kernel, which gives a smoother probability distribution for the data.

$$P_{\text{KDE}}(\mathbf{x}') = \int K(\mathbf{x}' - \mathbf{x})\hat{P}(\mathbf{x})d\mathbf{x} = \int K(\mathbf{x}' - \mathbf{x})\frac{\sum_i \delta(\mathbf{x} - \mathbf{x}^{(i)})}{N_{\text{data}}}d\mathbf{x} = \sum_i \frac{K(\mathbf{x}' - \mathbf{x}^{(i)})}{N_{\text{data}}}\ , \tag{14}$$

where $K(\mathbf{x}' - \mathbf{x})$ is a kernel function that is non-negative, for example a Gaussian function.

Given the state $|\Sigma_l\rangle$, one can perform generative modeling or classification based on the evaluation of the overlap $\langle\Phi^{\mathbf{s}}(\mathbf{x})|\Sigma_l\rangle$. As pointed out in Sec. 2.1, the projection with local feature maps made by a finite basis set will broaden the Dirac-delta function representing the training data. Therefore, we can interpret the overlap $\langle\Phi^{\mathbf{s}}(\mathbf{x})|\Sigma_l\rangle$ in a different way. The combination of local feature maps and the summation of the training data are equivalent to performing a summation



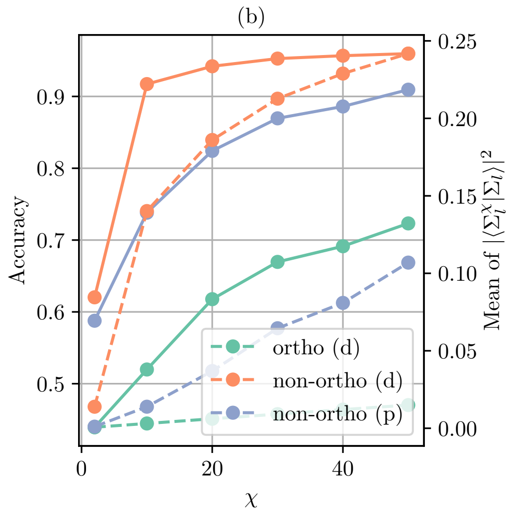


Figure 1: (a) Images sampled from pre-trained MPSs of bond dimension $\chi = 50$ by iterative compression scheme. We sample 5 binary images and 5 grey-scaled images from each MPS $|\Sigma_l^\chi\rangle$. (b) Test accuracy for MNIST classification by compressed MPS $|\Sigma_l^\chi\rangle$. We consider the direct compression with orthonormal (green) and the non-orthonormal feature map (orange) and the parallel compression with non-orthonormal feature map (blue). The solid lines indicate the test accuracy and the dashed lines indicate the mean of the square of overlaps to the exact state $|\Sigma_l\rangle$.

over smoothed data points to estimate the *probability amplitude* over variables $\mathbf{x}$. To be more precise, we apply the feature map $\Phi^{\mathbf{s}}(\mathbf{x})$ on the state $|\Sigma_l\rangle$ and obtain the probability amplitude $\Psi(\mathbf{x}) = \langle\Phi^{\mathbf{s}}(\mathbf{x})|\Sigma_l\rangle = \sum_{(i),s} \langle\mathbf{x}|\Phi^{\mathbf{s}}\rangle \langle\Phi^{\mathbf{s}}|\mathbf{x}^{(i)}\rangle / C_{\text{Norm}}$ over continuous variables $\mathbf{x}$. Notice the similarity to Eq. (14). We reinterpret the overlap $\langle\Phi^{\mathbf{s}}(\mathbf{x})|\Sigma_l\rangle$ as a quantum KDE model estimating the probability amplitude of the input data. While the quantum KDE requires storing information for all the data points in $|\Sigma_l\rangle$, the compression step in the algorithm acts as finding a compact representation for the models by minimizing the $\mathcal{L}_2$ distance, which resembles the standard approach of minimizing KL-divergence between the model distribution and the empirical distribution.

3 Results

To demonstrate the proposed algorithm, we test the direct compression and iterative compression for obtaining the compressed MPS $|\Sigma_l^\chi\rangle$ for the MNIST dataset. We illustrate the result in Fig. 1.

To show that the MPS learning procedure could be a good initialization for generative modeling, we perform an iterative compression of product states encoded by the non-orthonormal local feature map $[\cos(\frac{\pi x}{2}), \sin(\frac{\pi x}{2})]$ to obtain the MPS $|\Sigma_l^\chi\rangle$. Then we perform ancestral sampling from the MPS wavefunction to get binary images. In addition, we also perform two-stage sampling by first sampling the MPS to get $\mathbf{s}$ and then sampling the orthonormal local feature map $\phi(x) = [e^{i(3\pi/2)x}\cos(\frac{\pi}{2}x), e^{-i(3\pi/2)x}\sin(\frac{\pi}{2}x)]$ to get the grey-scale images $\mathbf{x}$. The result is shown in Fig. 1a.

We use the learned digit wavefunctions to predict the label on test data. The digit wavefunction, i.e. MPS, with the largest likelihood gives the prediction. We plot the test accuracy in Fig. 1b for direct compression and parallel compression over orthonormal and non-orthonormal local feature maps.

4 Discussion

We propose a pre-training algorithm for MPS models by iterative compression of digit wavefunctions (sum states) $|\Sigma_l\rangle$, which could be parallelized and have potentially exponential speedup. We test the algorithm on the MNIST dataset and observe reasonable results for tasks including sampling and classification. We provide a new interpretation of the overlap $\langle\Phi^{\mathbf{s}}(\mathbf{x})|\Psi_\theta^{\mathbf{s}}\rangle$ as the probability amplitude $\Psi_\theta(\mathbf{x})$. The overlap between the new data point $|\Phi^{\mathrm{s}}(\mathbf{x})\rangle$ and the state $|\Sigma_l\rangle$ can then be interpreted as a quantum version of a kernel density estimate for the probability amplitude.

In a recent work [10], the authors studied the entanglement properties of the uncompressed digit wavefunctions $|\Sigma_l\rangle$ and found a flat Schmidt spectrum. The authors conclude that the MPS resulting from supervised learning [7] must be some different and less entangled states. Our work is consistent with their observation as we also observe small overlaps with the exact state $|\Sigma_l\rangle$ when using a small bond dimension MPS for compression. However, surprisingly, these states already lead to good results in supervised learning, e.g., $> 95\%$ test accuracy with $\chi = 50$ by direct compression. Moreover, these states capture the correct long-range correlation in the images, as seen from the sampling. This suggests that the compressed digit wavefunctions could serve as good initial states for both supervised and unsupervised learning, although they have very small overlap with the exact digit wavefunctions.

It may require further studies to understand whether the method would work for larger datasets. It would be related to the entanglement and the mutual information scaling of the dataset [15], and we envision it would be necessary to use projected entangled pair states (PEPS) [16, 17, 18, 19] instead of MPS to scale up to larger images. While in this work, we encode each digit wavefunction $|\Sigma_l\rangle$ as a different MPS, one could also encode all the data in one wavefunction by increasing one index for the labels. A potential future direction is to incorporate parameterized basis functions for the local feature maps as in [20, 21, 22].

References

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[3] M. B. Hastings, "An area law for one-dimensional quantum systems," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2007, no. 08, p. P08024, 2007.

[4] J. Eisert, M. Cramer, and M. B. Plenio, "Colloquium: Area laws for the entanglement entropy," *Reviews of Modern Physics*, vol. 82, no. 1, p. 277, 2010.

[5] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states," *Annals of Physics*, vol. 326, no. 1, pp. 96–192, 2011.

[6] J. C. Bridgeman and C. T. Chubb, "Hand-waving and interpretive dance: an introductory course on tensor networks," *Journal of Physics A: Mathematical and Theoretical*, vol. 50, no. 22, p. 223001, 2017.

[7] E. M. Stoudenmire and D. J. Schwab, "Supervised learning with quantum-inspired tensor networks," *arXiv preprint arXiv:1605.05775*, 2016.

[8] Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, "Unsupervised generative modeling using matrix product states," *Phys. Rev. X*, vol. 8, p. 031012, Jul 2018.

[9] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.

[10] J. Martyn, G. Vidal, C. Roberts, and S. Leichenauer, "Entanglement and tensor networks for supervised image classification," *arXiv preprint arXiv:2007.06082*, 2020.

[11] S. Cheng, J. Chen, and L. Wang, "Information perspective to probabilistic modeling: Boltzmann machines versus born machines," *Entropy*, vol. 20, no. 8, p. 583, 2018.

[12] S. Cheng, L. Wang, T. Xiang, and P. Zhang, "Tree tensor networks for generative modeling," *Physical Review B*, vol. 99, no. 15, p. 155131, 2019.

[13] J.-G. Liu and L. Wang, "Differentiable learning of quantum circuit born machines," *Physical Review A*, vol. 98, no. 6, p. 062324, 2018.

[14] A. J. Ferris and G. Vidal, "Perfect sampling with unitary tensor networks," *Physical Review B*, vol. 85, no. 16, p. 165146, 2012.

[15] S. Lu, M. Kanász-Nagy, I. Kukuljan, and J. I. Cirac, "Tensor networks and efficient descriptions of classical data," *arXiv preprint arXiv:2103.06872*, 2021.

[16] F. Verstraete and J. I. Cirac, "Renormalization algorithms for quantum-many body systems in two and higher dimensions," *arXiv preprint cond-mat/0407066*, 2004.

[17] H. Niggemann, A. Klümper, and J. Zittartz, "Quantum phase transition in spin-3/2 systems on the hexagonal lattice—optimum ground state approach," *Zeitschrift für Physik B Condensed Matter*, vol. 104, no. 1, pp. 103–110, 1997.

[18] T. Nishino and K. Okunishi, "A density matrix algorithm for 3d classical models," *Journal of the Physical Society of Japan*, vol. 67, no. 9, pp. 3066–3072, 1998.

[19] G. Sierra, "The density matrix renormalization group, quantum groups and conformal field theory," in *Proceedings of the Workshop on the Exact Renormalization Group*, World Scientific, 1998.

[20] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," *arXiv preprint arXiv:1701.05517*, 2017.

[21] B. Hashemi and L. N. Trefethen, "Chebfun in three dimensions," *SIAM Journal on Scientific Computing*, vol. 39, no. 5, pp. C341–C363, 2017.

[22] A. Gorodetsky, S. Karaman, and Y. Marzouk, "A continuous analogue of the tensor-train decomposition," *Computer methods in applied mechanics and engineering*, vol. 347, pp. 59–84, 2019.

A Review for generative modeling

The problem setup for generative modeling is as follows: We are given a dataset $\{\mathbf{x}^{(i)}, y^{(i)}\}$, which is a collection of independent and identical distributed samples from the unknown distribution $P(\mathbf{x}, y)$ and i is the index of the samples. For example, the MNIST dataset consists of images $\{\mathbf{x}^{(i)}\}$ and labels $\{y^{(i)}\}$. We represent each image as a vector $\mathbf{x} = (x_1, x_2, \ldots, x_{n_{\text{pixel}}})$, which is either gray-scaled $\mathbf{x} \in \mathbb{R}^{n_{\text{pixel}}}$ or binary $\{0,1\}^{n_{\text{pixel}}}$. Every image has a label $y \in \mathbb{Z}^+$. The goal of the learning is to obtain an approximation to the true unknown distribution $P(\mathbf{x}, y)$ with some parameterized model $P_\theta(\mathbf{x}, y)$ from the observed data $\{\mathbf{x}^{(i)}, y^{(i)}\}$.

The generative models can be roughly separated into two categories by whether the models have tractable likelihood, i.e., efficient evaluation of normalized $P(\mathbf{x})$. The MPS models have tractable likelihood and permits an efficient direct sampling algorithm [14].

When the parameterized model has a tractable likelihood, one common approach for optimizing generative models is to minimize the forward KL-divergence between the empirical distribution $\hat{P}(\mathbf{x}) = \frac{\sum_i \delta(\mathbf{x}-\mathbf{x}^{(i)})}{C_{\text{Norm}}}$, where C_{Norm} is a normalization constant, and the parameterized model $P_\theta(\mathbf{x})$.

$$\begin{aligned}
\theta_{\text{opt}} &= \text{argmin}_\theta \int -\log P_\theta(\mathbf{x}) d\hat{P}(\mathbf{x}) &(15)\\
&= \text{argmin}_\theta \int -\log P_\theta(\mathbf{x}) \sum_i \delta(\mathbf{x}-\mathbf{x}^{(i)}) d\mu(\mathbf{x}) &(16)\\
&= \text{argmin}_\theta - \sum_i \log P_\theta(\mathbf{x}^{(\mathbf{i})}) &(17)
\end{aligned}$$

B Additional Data

Given a finite orthonormal basis set, the expansion

$$\Psi(x) = \sum_s \int \phi^s(x)\bar{\phi^s}(x')\delta(x'-\xi)dx' = \sum_s \langle x|\phi^s\rangle \langle \phi^s|\xi\rangle$$

of the Dirac-delta function at ξ would create a broadening of the function. We show the example in Fig. 2 considering the different orthonormal local feature maps: (i) $\phi(x) = [e^{i(3\pi/2)x}\cos(\pi x/2), e^{-i(3\pi/2)x}\sin(\pi x/2)]$, (ii) $\phi(x) = [\sin(\pi x), \sin(2\pi x), \sin(3\pi x), \sin(4\pi x)]$,

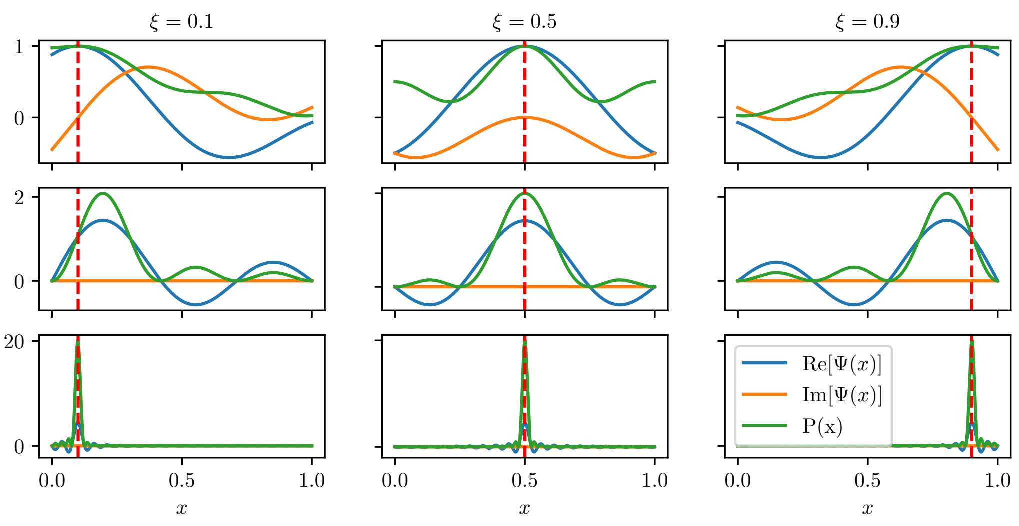


Figure 2: Data in the first row shows the $\Psi(x)$ resulting from the orthonormal local feature map $\phi(x) = [e^{i(3\pi/2)x}\cos(\frac{\pi}{2}x), e^{-i(3\pi/2)x}\sin(\frac{\pi}{2}x)]$. Data in the second row shows the $\Psi(x)$ resulting from the orthonormal local feature map $\phi(x) = [\sin(\pi x), \sin(2\pi x), \sin(3\pi x), \sin(4\pi x)]$. Data in the third row shows the $\Psi(x)$ resulting from the orthonormal local feature map $\phi(x) = [\sin(n\pi x)\ \ n \in \{1, \ldots, 40\}]$. The Dirac-delta functions are denoted by the red dashed lines.

(iii) $\phi(x) = [\sin(n\pi x),\ n \in \{1, \ldots, 40\}]$. We observe that the smoothing behaviour depends on the basis sets and gets closer to the Dirac-delta function with increasing number of basis states.

In Fig. 3, we show the result of taking partial training data to form $|\Sigma_l^\chi\rangle$ without performing compression. The bond dimension χ is exactly the number of training data taken. While in this case, we could still afford to exactly construct the states $|\Sigma_l^\chi\rangle$ of individual digit separately, in general the compression step is necessary for larger datasets and other tensor network states, e.g., PEPS.

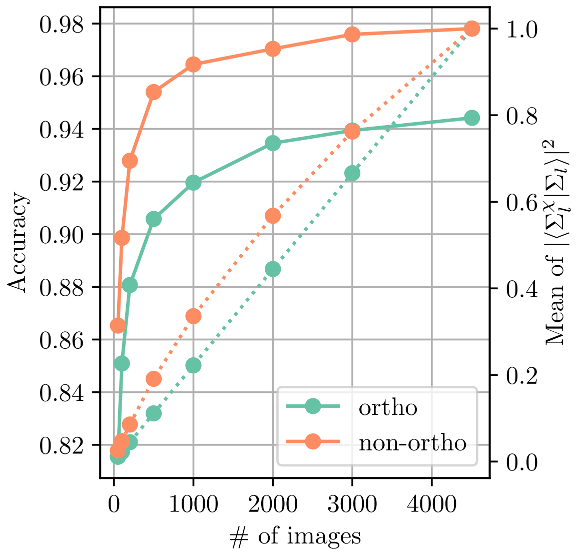


Figure 3: The results of $|\Sigma_l^\chi\rangle$ formed by subset of the training data without performing compression. The result for the orthonormal feature map $\phi(x) = [e^{i(3\pi/2)x}\cos(\frac{\pi}{2}x), e^{-i(3\pi/2)x}\sin(\frac{\pi}{2}x)]$ is colored in green. The result for the non-orthonormal feature map $\phi(x) = [\cos(\frac{\pi x}{2}), \sin(\frac{\pi x}{2})]$ is colored in orange. The solid lines indicate the test accuracy and the dashed lines indicate the mean of the square of overlaps to the exact state $|\Sigma_l\rangle$.