Quantum Neural Network States*

Zhih-Ahn Jia,[1,2,†] Biao Yi,[3] Rui Zhai,[4] Yu-Chun Wu,[1,2,‡] Guang-Can Guo,[1,2] and Guo-Ping Guo[1,2]

[1] *Key Laboratory of Quantum Information, Chinese Academy of Sciences, School of Physics, University of Science and Technology of China, Hefei, Anhui, 230026, P.R. China*

[2] *CAS Center For Excellence in Quantum Information and Quantum Physics, University of Science and Technology of China, Hefei, Anhui, 230026, P.R. China*

[3] *Department of Mathematics, Capital Normal University, Beijing 100048, P.R. China*

[4] *Institute of Technical Physics, Department of Engineering Physics, Tsinghua University, Beijing 10084, P.R. China*

(Dated: June 7, 2019)

In this work, we investigate the possibility of using artificial neural network to build ansatz quantum many-body states. The progresses on representing quantum many body states by stochastic recurrent neural network, restricted or unrestricted Boltzmann machine, are reviewed. Besides, we discuss the possibility of representing quantum states using feed-forward neural network which is relatively less studied in literatures. At last, entanglement features of the quantum neural network states are discussed for comparison with the tensor network states.

I. INTRODUCTION

One of the most challenging problems in condensed matter physics is to find the ground state of a given Hamiltonian, the difficulty is mainly originated from the power scaling of the Hilbert space dimension[1,2]. To obtain a better understanding of quantum many-body physical system beyond mean field paradigm and to study the behavior of strongly correlated electrons require us to find some good ways to approach the problem. Although the dimension of the Hilbert space of the system is exponential at the number of particles in general, fortunately, physical states frequently lie in a small corner of the whole space[3–7]. Physical properties of the system usually restrict the form of the ground state, for example, area-law states[8], ground states of local gapped systems[9] and many-body localized systems can be efficiently represented by tensor network[3,10,11] which is a new tool developed recent years for attacking the difficulty of efficiently representing quantum many-body states. Tensor network approach makes some great triumphs for quantum many-body problems, it has became a standard tool and many classical algorithms based tensor networks have been developed, such as density-matrix renormalization group (DMRG)[12], projected entangled pair states (PEPS)[13], folding algorithm[14], entanglement renormalization[15], time-evolving block decimation (TEBD)[16], etc.

In 2016, a new approach based on machine learning techniques, neural networks, is introduced as a variational ansatz for representing quantum many-body ground states[17], this stimulates an explosion of results to apply machine learning methods to investigate condensed matter physics, see Refs. for example[18–25].

During the last few years, machine learning has became the most rapidly growing interdisciplinary filed, machine learning techniques have also been successfully applied into many different scientific areas[26–28]: computer vision, speech recognition, chemical synthesis, etc. Among which the combination of quantum physics and machine learning generates a new exciting research field, quantum machine learning (QML)[29], which has recently attracted many attentions[17–23,25]. The researches of QML can be loosely categorized into two branches, the first one is to develop new quantum algorithms which share some features of machine learning and behave faster and better than their classical counterparts[18–20], the second one, which is also the focus of this work, is to use the classical machine learning methods to assist the study of quantum systems: such as distinguishing phases[21], quantum control[30], error-correcting of topological codes[31], quantum tomography[32,33] and so on. Besides all these progresses, we stress here that it can also be used to attack the difficulty of quantum many-body states.

Carleo and Troyer initially introduced the Restricted Boltzmann machine (RBM) to solve the Ising model and study the time evolution of the system[17]. Later, the entanglement properties of the RBM states is investigated[22], and its representational power is also studied[25,34]. Many explicit RBM constructs for different systems are given , including Ising model[17], toric code[23], graph states[25], stabilizer code[24] and topologically ordered states[23,24,34]. On the other hand, the deep Boltzmann machine (DBM) states are also investigated from different directions[25,35,36]. Despite all these progresses for applying neural networks in quantum physics, many important topics still remain to be explored, the first one is the exact definition of a quantum neural network state, although RBM and DBM states are investigated from different aspects, there are many other neural networks, it is natural to ask if they can similarly be applied for representing quantum states. Secondly, one central problem in studying neural network is its representational power, we can ask what is its counterpart in quantum mechanics and how to make the neural network work efficiently

---

*The paper is prepared for the special issue on quantum machine learning of the journal *advanced quantum technology* for representing quantum states, what kind of states can be efficiently represented by a specific neural network. In this work, we investigate partially these problems and review the important progresses in the field.

The work is organized as follows. In Sec. II we introduce the definition of artificial neural network. We explain the feed-forward neural network, perceptron and logistic neural networks and stochastic recurrent neural network Boltzmann machine (BM) in Sec. II A. Then we explain the representation power for feed-forward neural network in representing given functions and Boltzmann machine in approximating given probability distributions in Sec. II B. In Sec. III we explain how the neural network can be used as a variational ansatz for quantum states, the method given in Sec. III A is model-independent, that is, the way to construct states can be applied to any neural network (with the ability to continuously output real or complex number). Some concrete examples of neural network states is given in Sec. III B. Sec. III C is devoted to the efficient representational power of neural network in representing quantum states. In Sec. IV we discuss the entanglement features of the neural network states. In last section, some concluding remarks are given.

II. ARTIFICIAL NEURAL NETWORKS AND THEIR REPRESENTATIONAL POWER

Neural network is the mathematical model for abstracting the biological nerve system, which consists of adaptive units called neurons that are connected with each other extensively[37]. The basic elements comprise a neural network are artificial neurons, they are the mathematical objects which are abstractions of the biological neurons. In biological nerve system, each biological neuron is connected with many other neurons, when activated it will release of neurotransmitters to other neurons and makes the electric potentials of these neurons change. There is a threshold potential value for each neuron, while the electric potential exceeds the threshold, the neuron is activated.

There are several kinds of artificial neuron models, here, we introduced the most commonly used McCulloch-Pitts neuron model[38]. As shown in 1, there are n inputs $x_1, x_2, \cdots, x_n$ which are transmitted by n corresponding weighted connections $w_1, w_2, \cdots, w_n$. After the signals have reached the neuron, they firstly are added together with weights, the value then is compared with the bias b of the neuron to determine if the neuron is activated or deactivated. The process is characterized by the activation function f, the output of an neuron is then $y = f(\sum_{i=1}^{n} w_i x_i - b)$. Note that we can regard the bias as a weight w_0 for some fixed input $x_0 = -1$, the output then have a more compact form $y = f(\sum_{i=0}^{n} w_i x_i)$. When we put a large number of neurons together and make them connected with each other with some kind of connecting pattern, we get a neural network.

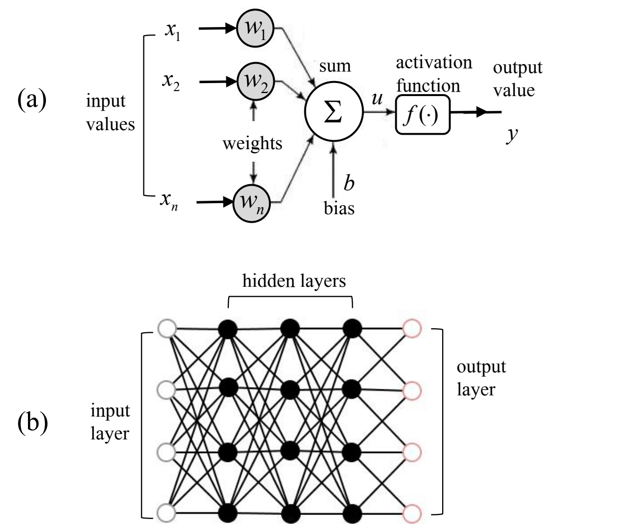

FIG. 1: (a) McCulloch-Pitts neuron model, where $x_1, \cdots, x_n$ are inputs of the neuron, $w_1, \cdots, w_n$ are weights corresponding to each input, Σ is the summation function, b is the bias, f is activation function and y is the output of the neuron; (b) a simple artificial neural network.

A. Some special neural networks

An artificial neural network is a set of neurons where some, or all, neurons are connected according to certain pattern. Note that we will put neurons at input and output ends. The input and output neurons are not neurons as we have introduced former, they are drawn just for convenience, there is no activation function on them.

1. *Rosenblatt's perceptron and logistic neuron network*

To explain the neural network, we first see an example of feed-forward neural network, perceptron, which was invented by Rosenblatt. In the history of artificial neural networks, multilayer perceptron plays a crucial role. In a perceptron the activation function of each neuron is chosen as Heaviside's step function

$$f(x) = \begin{cases} 0, & x \leq 0, \\ 1, & x > 0, \end{cases} \tag{1}$$

where zero value represents the activation status of the neuron and one value represents the deactivation status of the neuron. The output value of one neuron is the input value of the neurons connecting the given neuron.

The power of perceptron mainly comes from its hierarchical recursion structure. It has been shown that perceptron can be used for doing universal classical computation[38–40]:

Theorem 1. Rosenblatt's perceptron can be used for doing universal computation.

To see this, we note that NAND and FANOUT operations are universal for classical computation. In perceptron, we still assume the FANOUT operation works [1]. We only need to show that the perceptron can simulate the NAND operation. Suppose that x_1, x_2 are two inputs of the neuron and with weights both -2 and the bias of the neuron is chosen as -3. When inputs are $x_1 = x_2 = 1$, $f(x_1 w_1 + x_2 w_2) = 0$ otherwise its output is 1 which is exactly the output of NAND operation.

The perceptron is powerful in many applications but it still has some shortcomings. The most outstanding one is that the activation function is not continuous, small change of the weights may cause a big change of the output of the network, this make the learning process difficult. One way to remedy the shortcoming is to polish the activation function, usually the smooth activation function is chosen as logistic function:

$$f(x) = \frac{1}{1+e^{-x}}. \tag{2}$$

This kind of networks are usually named as logistic neural network or sigmoid neural network. Logistical neural network are used extensively in practice. Many problems can be solved by logistic neural network. We would like to say that the logistic function is chosen for convenience in updating process of learning and many other polished function of step function can be chosen as activation function and it is not necessary that in a neural network that all activation functions are the same.

2. *Boltzmann machine*

Now we introduce another special type of artificial neural networks, Boltzmann machine (a.k.a. stochastic Hopfield network with hidden units), which is an energy-based neural network model[41,42]. It has recently been introduced in many different physical areas[17,22,24,25,31,34,43–48] and their quantum version, quantum BMs, are also investigated[49]. For Boltzmann machine is very simlar as classical Ising model, here we will explain the Boltzmann machine neural network by frequently referring to the terminology of Ising model. Notice that BM is very different from the perceptrons and logistic neural network, it does not treat each neuron individually, thus there is not activation function attached to each specific neuron. Instead, Boltzmann machine treat neurons as a whole.

Given a graph G with vertex set $V(G)$ and edge set $E(G)$, the neurons $s_1, \cdots, s_n$ (spins in Ising model) are put on vertices, $n = |V(G)|$. If two vertices i and j are connected, there will be a weight w_{ij} (coupling constant in Ising model) between corresponding neurons s_i and s_j. For each neuron s_i, there is also a corresponding local bias (local field in Ising model). Like what have been done for Ising model, for each series of input values $\mathbf{s} = (s_1, \cdots, s_n)$ (spin configuration in Ising model) we can define the energy

$$E(\mathbf{s}) = -\sum_{\langle ij\rangle \in E(G)} w_{ij} s_i s_j - \sum_i s_i b_i. \tag{3}$$

Up to now, every thing is just like Ising model, no new concepts or techniques are introduced. The main difference is that, Boltzmann machine construction introduces a coloring on vertex. It labels the vertex by labels *hidden* and *visible*, we assume the first k neurons are hidden neurons denoted by $h_1, \cdots, h_k$ and the left l neurons are visible neurons denoted by $v_1, \cdots, v_l$ and $k + l = n$. Therefore, the energy is now $E(\mathbf{h}, \mathbf{v})$. Boltzmann machine is a parametric model of joint probability distribution between variables $\mathbf{h}$ and $\mathbf{v}$, the probability is given by

$$p(\mathbf{h}, \mathbf{v}) = \frac{e^{-E(\mathbf{h},\mathbf{v})}}{Z}, \tag{4}$$

where $Z = \sum_{\mathbf{h},\mathbf{v}} e^{-E(\mathbf{h},\mathbf{v})}$ is the partition function.

The general BM is very difficult to train, thus some restricted architecture of BM is introduced. The restricted Boltzmann machine (RBM) was initially invented by Smolensky[50] in 1986. In RBM, it is assumed that the graph G is a bipartite graph, hidden neurons only connect with visible neurons and there is no intra-layer connections. This kind of the restricted structure makes the neural network more easy to train and thus has be extensively investigated and utilized[17,22,24,25,31,34,43–48]. It has been shown that RBM can approximate every discrete probability distribution[51,64].

It is worth mentioning that Boltzmann machine is stochastic recurrent neural network, perceptron and logistic neural network is feed-forward neural network. There are many other type of neural networks, for more comprehensive knowledge, see books like[52,53] and articles on *Neural Networks* journal.

B. Representational power of neural network

In 1900, Hilbert formulated his famous 23 problems, among which the thirteenth problem is devoted to the possibility of representing an n-variable function as a superposition of functions of a lesser number of variables. This problem is closely related to the representational power of neural networks. Kolmogorov[54,55] and Arnold[56] proved that for continuous n-variable functions, this is the case, the result is known as Kolmogorov-Arnold representation theorem (a.k.a. Kolmogorov superposition theorem):

---

[1] Here, we emphasize the importance of the FANOUT operation, which is usually omitted from the universal set of gates in classical computation theory. But the operation is forbidden in quantum computation by the famous no-cloning theorem.

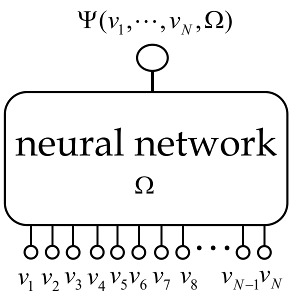

FIG. 2: Schematic diagram for neural network ansatz state.

Theorem 2. Any n-variable real continuous function $f : [0,1]^n \to \mathbb{R}$ can be written as sums and compositions of continuous univariate fuctions, more precisely, there exist real positive number $a, b, \lambda_p, \lambda_{p,q}$ and a real monotonic increasing function $\phi : [0,1] \to [0,1]$ such that

$$f(x_1, \cdots, x_n) = \sum_{q=1}^{2n+1} F(\sum_{p=1}^{n} \lambda_p \phi(x_p + aq) + bq), \quad (5)$$

or

$$f(x_1, \cdots, x_n) = \sum_{q=1}^{2n+1} F(\sum_{p=1}^{n} \lambda_{pq} \phi(x_p + aq)), \quad (6)$$

where F is a real and continuous function called outer function, and ϕ called inner function. Notice that a and F can be different in two representations.

Later, many follow-up works contributed to understand more deeply the representation power of neural networks from different aspects[57]. Mathematicians consider the problem in different support sets and different metric between functions. The discrete-function version of the problem is also extensively investigated. As we have indicated in last section, McCulloch and Pitts[38] showed that any Boolean function can be represented by perceptron and based on this Rosenblatt developed the learning algorithm[58]. Słupecki proved that all k-logic functions can be represented as a superposition of one-variable functions and any given significant function[59]. Cybenko[60], Funahashi[61] and Hornik et al.[62] proved that n-variable functions defined on compact subset of $\mathbb{R}^n$ can be approximated by a four-layer network with only logistic activation functions and linear activation function. Hecht[63] took a step further, he proved that any n-variable continuous function can be represented by a two-layer neural network involed logistic activation functions of the first layer and arbitrary activation functions on second layer. These results can summarized as the following:

Theorem 3. The feed-forward neural network can approximates any continuous n-variable functions and any n-variable discrete functions.

For the stochastic recurrent neural network BM, the power of approximating probability distributions is also studied extensively. An important result of Le Roux and Bengio[51] claims that:

Theorem 4. Any discrete probability distribution $p : \mathbb{B}^n \to \mathbb{R}_{\geq 0}$ can be approximated with an RBM with $k+1$ hidden neurons where $k = |\mathrm{supp}(p)|$ is the cardinality of the support of p (i.e., the number of vectors with nonzero probabilities) arbitrarily well in the metric of the Kullback–Leibler divergence.

The theorem means that any discrete probability distribution can be approximated by RBM. The bound of number of hidden neurons is later improved[64].

Here we must stress that these representation theorems only concern if the given function or probability distribution can be represented by the neural network. In practice, the number of parameters to be learned can not be too large at the number of input neurons when we build a neural networks. If a neural network can represent a function or distribution in polynomial time (the number of parameters depend polynomially on the number of input neurons), we say that the representation is efficient.

III. ARTIFICIAL NEURAL NETWORK ANSATZ FOR QUANTUM MANY-BODY SYSTEM

A. Neural network ansatz state

We now explain how neural network can be used as a variational ansatz for quantum many-body systems. For a given many-body pure state $|\Psi\rangle$ of an n-particle p-level physical system

$$|\Psi\rangle = \sum_{v_1, v_2 \cdots, v_N = 1}^{p} \Psi(v_1, v_2 \cdots, v_N) |v_1\rangle \otimes |v_2\rangle \otimes \cdots \otimes |v_N\rangle.$$

The coefficient $\Psi(v_1, v_2 \cdots, v_N)$ of the state can be regarded as a N-variable complex function. To characterize a state, we only need to give the corresponding value of Ψ function for each variable $\mathbf{v} = (v_1, v_2 \cdots, v_N)$. One of the difficulty in quantum many-body physics is that fully characterizing of an N-particle system requires $O(p^N)$ coefficients, which is exponentially large in system size N and thus is computational inefficient. Let us now see how neural network can be used to attack the difficulty.

To represent a state, we first build a specific architecture of neural network for which we denote the set of adjustable parameters as $\Omega = \{w_{ij}, b_i\}$. The number of input neurons is assumed to be the same as the number of physical particles N. For each series of input $\mathbf{v} = (v_1, \cdots, v_N)$, we can anticipate the neural network to output a complex number $\Psi(\mathbf{v}, \Omega)$ which both depends on input values and parameters of the net. In this way, a variational state

$$|\Psi(\Omega)\rangle = \sum_{\mathbf{v}\in\mathbb{Z}_p^N} \Psi(\mathbf{v},\Omega)|\mathbf{v}\rangle, \tag{7}$$

is obtained, where the sum is run over all basis labels. The state in Eq. (7) is a variational state, for a given Hamiltonian H, the corresponding energy functional is

$$E(\Omega) = \frac{\langle\Psi(\Omega)|H|\Psi(\Omega)\rangle}{\langle\Psi(\Omega)|\Psi(\Omega)\rangle}. \tag{8}$$

According variational method, the aim now is to minimize the energy functional and obtain the corresponding parameter values and then the (approximated) ground state is obtained. The process of adjusting parameters and find the minimum of energy functional is done using the neural network learning (See Fig. 2 for schematic illustration).

The notion of the efficiency of the neural network ansatz in representing a quantum many-body state is defined as the dependency relation of the number of nonzero parameters $|\Omega|$ involved in the representation and number of physical particles N: if the $|\Omega| = O(\mathsf{poly}(N))$, the representation is called efficient. The aim for solving a given eigenvalue equation is thus to build an neural network for which the ground state can be represented efficiently.

B. Some examples

Let us now see some concrete examples. The first one we are considering is the logistic neural network state, where weights and biases now must be chosen as complex numbers and the activation function $f(z) = 1/(1+e^{-z})$ complex function too. As shown in Fig. 3, we take two-qubit state to give an illustration. We assume the biases are $b_1,\cdots,b_4$ for hidden neurons $h_1,\cdots,h_4$ respectively, and the weights between neurons are denoted as w_{ij}. We construct the state coefficient neuron by neuron now.

In Fig. 3, the output for $h_i, i=1,2,3$ is $y_i = f(v_1w_{1i}+v_2w_{2i}-b_i)$ respectively, these outputs are transmitted to h_4, after acting by h_4 we get the state coefficient as

$$\Psi_{log}(v_1,v_2,\Omega) = f(w_{14}y_1 + w_{24}y_2 + w_{34}y_3 - b_4), \tag{9}$$

where $\Omega = \{w_{ij}, b_i\}$. Summing over all possible input values, we obtain the quantum state $|\Psi_{log}(\Omega)\rangle = \sum_{v_1,v_2}\Psi_{log}(v_1,v_2,\Omega)|v_1,v_2\rangle$ up to a normalization factor. We see that logistic neural network state have a hierarchical iteration control structure which is responsible for the representation power of the network in representing states.

However, when we want to give the neural network parameters of a given state $|\Psi\rangle$ explicitly, we find that $f(z) = 1/(1+e^{-z})$ can not take zero and one values exactly. Zero and one values are its asymptotic values. This shortcoming can be remedy by polishing step function in another way. Here we give a real function solution, the complex case can be done similarly. The idea is very simple, we cut the function into pieces and then gluing them together in some polished way. Suppose that we want to construct a smooth activation function $F(x)$ such that

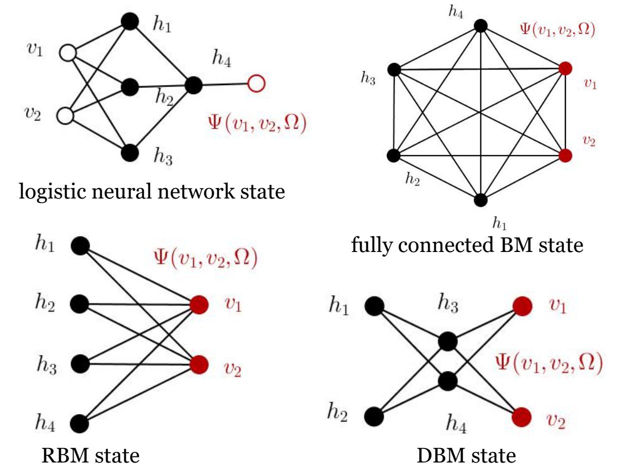

FIG. 3: Examples of two-qubit neural network ansatz states.

$$F(x)\begin{cases} =0, & x\leq -\frac{a}{2}, \\ \in(0,1) & -\frac{a}{2} < x < \frac{a}{2}, \\ =1, & x \geq \frac{a}{2}, \end{cases} \tag{10}$$

we can choose a kernel function

$$K(x) = \begin{cases} \frac{4x}{a^2}+\frac{2}{a}, & -\frac{a}{2}\leq x\leq 0, \\ \frac{2}{a}-\frac{4x}{a^2}, & 0\leq x < \frac{a}{2}, \end{cases} \tag{11}$$

the require function can then be constructed as

$$F(x) = \int_{x-\frac{a}{2}}^{x+\frac{a}{2}} K(x-t)s(t)dt, \tag{12}$$

where $s(t)$ is step function. It's easy to check that constructed function $F(x)$ is differentiable and satisfies Eq. (10). In this way the explicit neural network parameters can be given for a given state.

For BM states, we notice that classical BM networks can approximate discrete probability distribution, the quantum state coefficient $\Psi(\mathbf{v})$ is the square root of the probability distribution thus should also be able to represented by BM. This is one reason that BM states are introduced as a representation for quantum states. Here we only treat the case of fully connected BM states in Fig. 3, the cases for RBM and DBM are similar. As in logistic states, the weights and biases of BM are now complex numbers. The energy function is

$$\begin{aligned} E(\mathbf{h},\mathbf{v}) = &-(\sum_i v_i a_i + \sum_j h_j b_j + \sum_{\langle ij\rangle} w_{ij}v_ih_j \\ &+ \sum_{\langle jj'\rangle} w_{jj'}h_jh_{j'} + \sum_{\langle ii'\rangle} w_{ii'}v_iv_{i'}), \end{aligned} \tag{13}$$

where a_i, b_j are biases of visible neurons and hidden neurons respectively, w_{ij}, $w_{jj'}$, $w_{i,i'}$ are connection weights. The state coefficients are now

$$\Psi_{BM}(\mathbf{v},\Omega)=\sum_{\mathbf{h}}\frac{e^{-E(\mathbf{h},\mathbf{v})}}{Z} \tag{14}$$

with $Z=\sum_{\mathbf{v},\mathbf{h}}e^{-E(\mathbf{h},\mathbf{v})}$ the partition function. The quantum state is $|\Psi_{BM}(\Omega)\rangle=\frac{\sum_{\mathbf{v}}\Psi_{BM}(\mathbf{v},\Omega)|\mathbf{v}\rangle}{\mathcal{N}}$ where $\mathcal{N}$ is the normalizing factor.

C. Representational power for neural network states

Since neural network states were introduced in many-body physics to efficiently represent the ground state of gapped many-body quantum system[17], many people studied their representation power. We now know that RBMs are capable of representing many different classes of states[23–25,46]. Since RBM allows efficient sampling, unlike its unrestricted counterparts, they are also the most studied cases. The DBM states are also explored in some works[25,35,36]. In this section we briefly review the progresses in this direction.

We first list some known classes of states that can be efficiently represented by RBM:

- $\mathbb{Z}_2$-toric code states[23];
- Graph states[25];
- Stabilizer states with generators of pure type, $\mathbf{S}_X, \mathbf{S}_Y, \mathbf{S}_Z$ and their arbitrary union[24];
- Perfect surface code states, surface code states with boundary, defect and twist[24];
- Kitaev's $D(\mathbb{Z}_d)$ quantum double ground states[24];

Although many important classes of states can be represented by RMB, there is a crucial result about the limitation of RBM[25]: there exist states[65] which can be expressed as PEPS but can not be efficiently represented by RBM and class of RBM states is not closed under unitary transformations. One way to remedy the defect is by adding one more hidden layer, thus by using DBM.

The DBM can efficiently represent physical states including:

- Any state which can be efficiently represented by RBMs [2];
- Any n-qubit quantum states generated by a quantum circuit of depth T, the number of hidden neurons is $O(nT)$[25];

[2] This can be done by setting the parameters involved in deep hidden layer all zeros and only parameters of shallow hidden layer are left nonzero.

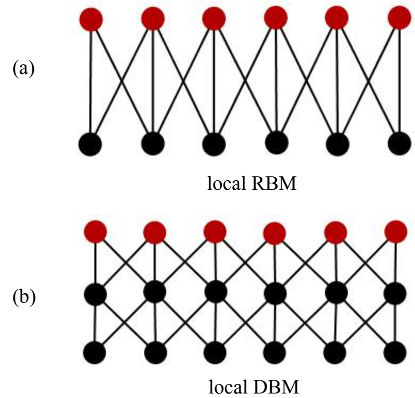

FIG. 4: (a) Example of local RBM state; (b) Example of local DBM state.

- Tensor network states consist of n-local tensors with bound dimension D and maximum coordination number d, the number of hidden neurons is $O(nD^{2d})$[25];
- The ground states of Hamiltonian with gap Δ, the number of hidden neurons is $O(\frac{m^2}{\Delta}(n-\log\epsilon))$ where ϵ is the representational error[25];

Note that there are many known results about the BM states, but the case for other neural networks is barely explored.

IV. ENTANGLEMENT PROPERTIES OF ARTIFICIAL NEURAL NETWORK STATES

The notion of entanglement is ubiquitous in physics now, to understand the entanglement properties of the many-body state is one of the central issues in both condensed matter physics and quantum information theory. Tensor network representations of quantum states have an important advantage that entanglement can be read out more easily. Here we discuss the entanglement properties of the neural network states for comparison with tensor networks.

For a given N-particle quantum system in state $|\Psi\rangle$, we can divide the N particles into two groups $\mathcal{A}$ and $\mathcal{A}^c$. In this bipartition, we can calculate the Rényi entanglement entropy $S_R^\alpha(\mathcal{A}):=\frac{1}{1-\alpha}\log\mathrm{Tr}\rho_{\mathcal{A}}^\alpha$ which characterizes the entanglement between $\mathcal{A}$ and $\mathcal{A}^c$, where $\rho_{\mathcal{A}}=\mathrm{Tr}_{\mathcal{A}^c}(|\Psi\rangle\langle\Psi|)$ is the reduced density matrix. If Rényi entanglement entropy is nonzero, then $\mathcal{A}$ and $\mathcal{A}^c$ are entangled.

The entanglement property is encoded in the geometry of contraction patterns of the local tensors for tensor network states. For neural network states, it was shown that then entanglement is encoded in the connecting patterns of the neural networks[22,36,46–48]. For RBM states, Deng et al.[22] showed that locally connected RBM states obey the entanglement area law, see Fig. 4 (a) for an illustration of local RBM states. Nonlocal connections result in the volume-law entanglement of the states[22]. The result is extended for any BM by us, we shown that by cutting intra-layer connection and adding hidden neurons, any BM states can be reduced into a DBM with several hidden layers. Then using the folding trick, folding the odd layers and even layers separately, every BM is reduced into a DBM with only two hidden layers[25,36]. Then we shown that locally connected DBM states obey the entanglement area law, and DBM with nonlocal connections will possess volume-law entanglement[36], see Fig. 4 (b) for an illustration of local DBM states.

The relation of BM states and tensor network states are investigated in Refs.[25,46,47] and some algorithmic way of transforming a RBM into matrix product states is given in Ref.[47]. The ability to represent tensor network states using BM is investigated in[25,46] from complexity theoretic aspect.

One more thing worth mentioning is to realize the holographic geometry-entanglement correspondence using BM states[36,48]. When proving the entanglement area law and volume law of the BM states, the conception of locality must be introduced, this means that we must introduce a geometry between neurons. This geometry results in the entanglement features of the state. When we try to understand the holographic entanglement entropy, we first tile the neurons in a given geometry and then make it learn from data. After the learning process is done, we can see the connecting pattern of the neural network and analyze the corresponding entanglement properties which have the direct relation with the given geometry, like the signs of curvature of the space.

Although many progresses on entanglement properties of the neural network states have been made, we still know very little about it. The entanglement features of neural networks other than BM are not investigated at all. These are all left to be explored in the future.

V. CONCLUSION AND DISCUSSIONS

In this work we have discussed aspects of the quantum neural network states. Two important kinds of neural networks, feed-forward and stochastic recurrent ones, are chosen as examples to illustrate how neural network can be used as a variational ansatz of quantum many-body systems. We review the progresses of research of BM states, which are the most explored ones. The representation power of neural network states is discussed and the entanglement features of the RBM and DBM states are reviewed.

Besides all above, here we want to give some remarks on the main open problems on this direction.

- Although the BM states have been studied from aspects, many other neural networks are less explored for representing quantum states both numerically and theoretically, this raises the question: if networks like convolutional neural network can also efficiently represent quantum states and what is the differences between these different representations?

- Developing the representation theorem for complex function is also a very important topic in quantum neural network states.

- Is there any easy way to read out entanglement properties from neural networks like what tensor networks do?

We hope that our review of the quantum neural network states will inspire more works and the above crucial topics can be explored in the future.

Acknowledgments

J.-A. Jia thanks Liang Kong, Tian Lan for discussions during his stay in Tsinghua university, and he also benefits from the discussion with Giuseppe Carleo during the first international conference on "Machine Learning and Physics" at IAS, Tsinghua university. This work was supported by the Anhui Initiative in Quantum Information Technologies (Grants No. AHY080000)

---

† Electronic address: giannjia@foxmail.com
‡ Electronic address: wuyuchun@ustc.edu.cn

[1] T. J. Osborne, "Hamiltonian complexity," Reports on Progress in Physics **75**, 022001 (2012).
[2] F. Verstraete, "Quantum hamiltonian complexity: Worth the wait," Nature Physics **11**, 524 (2015).
[3] R. Orús, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states," Annals of Physics **349**, 117 (2014).
[4] Z. Landau, U. Vazirani, and T. Vidick, "A polynomial time algorithm for the ground state of one-dimensional gapped local hamiltonians," Nature Physics **11**, 566 (2015).
[5] I. Arad, Z. Landau, U. Vazirani, and T. Vidick, "Rigorous rg algorithms and area laws for low energy eigenstates in 1d," Communications in Mathematical Physics **356**, 65 (2017).
[6] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, "Computational complexity of projected entangled pair states," Phys. Rev. Lett. **98**, 140506 (2007).
[7] A. Anshu, I. Arad, and A. Jain, "How local is the information in tensor networks of matrix product states or

projected entangled pairs states," Phys. Rev. B **94**, 195143 (2016).

[8] J. Eisert, M. Cramer, and M. B. Plenio, "Colloquium," Rev. Mod. Phys. **82**, 277 (2010).

[9] L. Amico, R. Fazio, A. Osterloh, and V. Vedral, "Entanglement in many-body systems," Rev. Mod. Phys. **80**, 517 (2008).

[10] M. Friesdorf, A. H. Werner, W. Brown, V. B. Scholz, and J. Eisert, "Many-body localization implies that eigenvectors are matrix-product states," Phys. Rev. Lett. **114**, 170505 (2015).

[11] F. Verstraete, V. Murg, and J. Cirac, "Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems," Advances in Physics **57**, 143 (2008), https://doi.org/10.1080/14789940801912366 .

[12] S. R. White, "Density matrix formulation for quantum renormalization groups," Phys. Rev. Lett. **69**, 2863 (1992).

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

[14] M. C. Bañuls, M. B. Hastings, F. Verstraete, and J. I. Cirac, "Matrix product states for dynamical simulation of infinite chains," Phys. Rev. Lett. **102**, 240603 (2009).

[15] G. Vidal, "Entanglement renormalization," Phys. Rev. Lett. **99**, 220405 (2007).

[16] G. Vidal, "Efficient classical simulation of slightly entangled quantum computations," Phys. Rev. Lett. **91**, 147902 (2003).

[17] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks," Science **355**, 602 (2017).

[18] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," Phys. Rev. Lett. **113**, 130503 (2014).

[19] V. Dunjko, J. M. Taylor, and H. J. Briegel, "Quantum-enhanced machine learning," Phys. Rev. Lett. **117**, 130501 (2016).

[20] A. Monràs, G. Sentís, and P. Wittek, "Inductive supervised quantum learning," Phys. Rev. Lett. **118**, 190503 (2017).

[21] J. Carrasquilla and R. G. Melko, "Machine learning phases of matter," Nature Physics **13**, 431 (2017).

[22] D.-L. Deng, X. Li, and S. Das Sarma, "Quantum entanglement in neural network states," Phys. Rev. X **7**, 021021 (2017).

[23] D.-L. Deng, X. Li, and S. Das Sarma, "Machine learning topological states," Phys. Rev. B **96**, 195145 (2017).

[24] Z.-A. Jia, Y.-H. Zhang, Y.-C. Wu, G.-C. Guo, and G.-P. Guo, "Efficient machine learning representations of surface code with boundaries, defects, domain walls and twists," arXiv preprint arXiv:1802.03738 (2018).

[25] X. Gao and L.-M. Duan, "Efficient representation of quantum many-body states with deep neural networks," Nature Communications **8**, 662 (2017).

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**, 436 (2015).

[27] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science **313**, 504 (2006).

[28] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, Vol. 1 (MIT press Cambridge, 1998).

[29] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," Nature **549**, 195 (2017).

[30] M. August and X. Ni, "Using recurrent neural networks to optimize dynamical decoupling for quantum memory," Phys. Rev. A **95**, 012335 (2017).

[31] G. Torlai and R. G. Melko, "Neural decoder for topological codes," Phys. Rev. Lett. **119**, 030501 (2017).

[32] Y. Zhang and E.-A. Kim, "Quantum loop topography for machine learning," Phys. Rev. Lett. **118**, 216401 (2017).

[33] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, "Neural-network quantum state tomography," Nature Physics **14**, 447 (2018).

[34] Y. Huang and J. E. Moore, "Neural network representation of tensor network and chiral states," arXiv preprint arXiv:1701.06246 (2017).

[35] W.-C. Gan and F.-W. Shu, "Holography as deep learning," International Journal of Modern Physics D **26**, 1743020 (2017).

[36] Z.-A. Jia, R. Zhai, Y.-C. Wu, and G.-C. Guo, "Holographic entanglement-geometry duality in deep neural network states," to be published (2018).

[37] T. Kohonen, "An introduction to neural computing," Neural Networks **1**, 3 (1988).

[38] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," The bulletin of mathematical biophysics **5**, 115 (1943).

[39] M. Minsky and S. A. Papert, *Perceptrons: An introduction to computational geometry* (MIT press, 2017).

[40] M. A. Nielsen, *Neural Networks and Deep Learning* (Determination Press, 2015).

[41] G. E. Hinton and T. J. Sejnowski, "Optimal perceptual inference," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (IEEE New York, 1983) pp. 448–453.

[42] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for boltzmann machines," Cognitive science **9**, 147 (1985).

[43] G. Torlai and R. G. Melko, "Learning thermodynamics with boltzmann machines," Phys. Rev. B **94**, 165134 (2016).

[44] K.-I. Aoki and T. Kobayashi, "Restricted boltzmann machines for the long range ising models," Modern Physics Letters B **30**, 1650401 (2016).

[45] S. Weinstein, "Learning the einstein-podolsky-rosen correlations on a restricted boltzmann machine," arXiv preprint arXiv:1707.03114 (2017).

[46] L. Huang and L. Wang, "Accelerated monte carlo simulations with restricted boltzmann machines," Phys. Rev. B **95**, 035105 (2017).

[47] J. Chen, S. Cheng, H. Xie, L. Wang, and T. Xiang, "Equivalence of restricted boltzmann machines and tensor network states," Phys. Rev. B **97**, 085104 (2018).

[48] Y.-Z. You, Z. Yang, and X.-L. Qi, "Machine learning spatial geometry from entanglement features," Phys. Rev. B **97**, 045153 (2018).

[49] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, "Quantum boltzmann machine," Phys. Rev. X **8**, 021050 (2018).

[50] P. Smolensky, *Information processing in dynamical systems: Foundations of harmony theory*, Tech. Rep. (COLORADO UNIV AT BOULDER DEPT OF COMPUTER SCIENCE, 1986).

[51] N. L. Roux and Y. Bengio, "Representational power of restricted boltzmann machines and deep belief networks," Neural Computation **20**, 1631 (2008).

[52] C. Bishop, C. M. Bishop, *et al.*, *Neural networks for pattern recognition* (Oxford university press, 1995).
[53] L. V. Fausett *et al.*, *Fundamentals of neural networks: architectures, algorithms, and applications*, Vol. 3 (Prentice-Hall Englewood Cliffs, 1994).
[54] A. Kolmogorov, "The representation of continuous functions of several variables by superpositions of continuous functions of a smaller number of variables," .
[55] A. N. Kolmogorov, "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition," in *Doklady Akademii Nauk*, Vol. 114 (Russian Academy of Sciences, 1957) pp. 953–956.
[56] V. I. Arnold, *Vladimir I. Arnold-Collected Works: Representations of Functions, Celestial Mechanics, and KAM Theory 1957-1965*, Vol. 1 (Springer Science & Business Media, 2009).
[57] D. Alexeev, "Neural-network approximation of functions of several variables," Journal of Mathematical Sciences **168**, 5 (2010).
[58] F. Rosenblatt, *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*, Tech. Rep. (CORNELL AERONAUTICAL LAB INC BUFFALO NY, 1961).
[59] J. Słupecki, "A criterion of fullness of many-valued systems of propositional logic," Studia Logica **30**, 153 (1972).
[60] G. Cybenko, "Approximations by superpositions of a sigmoidal function," Mathematics of Control, Signals and Systems **2**, 183 (1989).
[61] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," Neural networks **2**, 183 (1989).
[62] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural networks **2**, 359 (1989).
[63] R. Hecht-Nielsen, "Kolmogorov's mapping neural network existence theorem," in *Proceedings of the IEEE International Conference on Neural Networks III* (IEEE Press, 1987) pp. 11–13.
[64] G. Montufar and N. Ay, "Refinements of universal approximation results for deep belief networks and restricted boltzmann machines," Neural Comput. **23**, 1306 (2011).
[65] X. Gao, S.-T. Wang, and L.-M. Duan, "Quantum supremacy for simulating a translation-invariant ising spin model," Phys. Rev. Lett. **118**, 040502 (2017).