Unsupervised prototype learning in an associative-memory network

Huiling Zhen[1], Shang-Nan Wang[1,2], and Hai-Jun Zhou[1,2*]
[1] *Key Laboratory of Theoretical Physics, Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China*
[2] *College of Physical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China*

Unsupervised learning in a generalized Hopfield associative-memory network is investigated in this work. First, we prove that the (generalized) Hopfield model is equivalent to a semi-restricted Boltzmann machine with a layer of visible neurons and another layer of hidden binary neurons, so it could serve as the building block for a multilayered deep-learning system. We then demonstrate that the Hopfield network can learn to form a faithful internal representation of the observed samples, with the learned memory patterns being prototypes of the input data. Furthermore, we propose a spectral method to extract a small set of concepts (idealized prototypes) as the most concise summary or abstraction of the empirical data.

I. INTRODUCTION

Biological nervous systems store patterns in a distributed way by encoding memory in the synaptic weights of neuron-neuron connections [1]. Hebb in 1949 conceived a fundamental microscopic mechanism of associative learning by suggesting that simultaneous excitations of pre- and post-synaptic neurons will lead to an increase in the synaptic weight [2]. Inspired by this Hebbian rule, Amari and peers in the 1970s constructed the first neural network models for associative memory and concept formation [3, 4]. Later in 1982 Hopfield designed an explicit memory-encoding formula to investigate the storage capacity of associative-memory networks [5]. The energy function of the Hopfield model is

$$E(\mathbf{v}) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} v_i W_{ij} v_j \ , \tag{1}$$

where $\mathbf{v} \equiv (v_1, v_2, \ldots, v_N)$ denotes a generic spin configuration of the N neurons in the system, with $v_i = +1$ (active) or $v_i = -1$ (quiescent). The coupling W_{ij} between neurons i and j are determined by the K stored memory patterns $\xi^\mu \equiv (\xi_1^\mu, \xi_2^\mu, \ldots, \xi_N^\mu)$, with $\mu = 1, 2, \ldots, K$, through

$$W_{ij} \equiv \frac{1}{K}\sum_{\mu=1}^{K} \xi_i^\mu \xi_j^\mu \ . \tag{2}$$

In the case of the stored patterns being binary and completely random ($\xi_i^\mu = \pm 1$ for $i \in [1, N]$ and $\mu \in [1, K]$), reliable memory retrieval is feasible as long as the number K of stored patterns is below $0.14N$ [6].

The storage capacity of the Hopfield model and its extensions has been thoroughly explored in the statistical physics community following these pioneering work [3, 5, 6]. However, the dynamical processes of learning in the Hopfield network were much less investigated. Huang considered learning as an inverse Ising problem in several recent papers [7–9], but he focused on the neuron-neuron couplings W_{ij} instead of the memory patterns ξ^μ. The Hopfield model is a recurrent neural network in which every neuron is directly affected by all the other neurons. It has not yet been widely adopted in deep-learning applications [10]. Instead the restricted Boltzmann machine (RBM) proposed by Smolensky in 1986 [11] has been much more popular. An RBM is an extension of a perceptron [4]. It has a layer of N visible neurons and another layer of K hidden neurons, and its energy function is

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i=1}^{N} a_i v_i - \sum_{\mu=1}^{K} b_\mu h_\mu - \sum_{i=1}^{N}\sum_{\mu=1}^{K} v_i C_{i\mu} h_\mu \ , \tag{3}$$

where $\mathbf{h} \equiv (h_1, h_2, \ldots, h_K)$ is a spin configuration of the hidden neurons. The bias parameters a_i and b_μ and the coupling $C_{i\mu}$ between visible and hidden neurons are usually learned on a large set of sample configurations $\mathbf{v}$ of the visible neurons [10, 12–15]. From the learned coupling matrix one can define K feature vectors as $f_\mu \equiv (C_{1\mu}, C_{2\mu}, \ldots, C_{N\mu})$ for $\mu = 1, 2, \ldots, K$. The hidden (output) neurons of one RBM can serve as the visible (input) neurons of another RBM to form a multilayered deep neural network [10, 13].

The RBM (3) is a bipartite network with conditional independence within the visible neurons and within the hidden neurons. Configuration sampling is therefore much easier in the RBM than in the Hopfield model. On the other hand, the feature vectors f_μ inferred by the RBM often have only little apparent resemblance to the individual input visible configurations [10, 12, 14]. It is quite non-trivial to interpret the "physical" meanings of the feature vectors and to figure out what the RBM has really learned about the input data.

In the last few years there was a revival of research attention on the Hopfield neural network [16–19]. Of especial interest to us was the demonstration by Krotov and Hopfield that dense associative-memory networks can achieve very good performance in recognizing patterns through supervised learning [16, 17]. In the present work we extend these earlier efforts to study unsuper-

*Corresponding author. Email address: zhouhj@itp.ac.cn

vised leaning in the (generalized) Hopfield network. Different from Refs. [16, 17] which focused on the discrimination performance, here we aim at achieving better understanding on the learned memory parameters and on the learning capacity of the Hopfield network.

In the next section we describe in more detail the generalized version of the Hopfield model and then prove its equivalence to a semi-restricted Boltzmann machine (semi-RBM) with two layers of binary neurons. This equivalence means that the Hopfield network can be stacked to form a deep neural network. After describing the learning algorithm in Sec. III, we then work on random datasets obtained through a simple generative process (Sec. IV) and on the MNIST data set of handwritten digits (Sec. V). We demonstrate that the learned memory patterns of the Hopfield neural network can be interpreted as prototypes of the input microscopic configurations. It is also possible to extract a small set of "concepts" (idealized prototypes) from the much larger set of learned memory patterns. These extracted concepts are the most concise summary or abstraction of the input data.

Our work is an effort to explore the statistical physics of learning. For the sake of clarity we only consider the simplest network architecture in the present work. When several Hopfield networks cooperate with each other in the form of a multilayered deep network, the learning power will surely be further boosted [20]. A lot of theoretical and algorithmic efforts are needed along this important direction.

II. THE GENERALIZED HOPFIELD MODEL

We consider a generalized form of the Hopfield model, which contains N neurons and K stored patterns. This model is equivalent to a semi-RBM with N visible neurons and K binary hidden neurons.

A. Associative memory

An instantaneous state (configuration) of the N neurons is $\mathbf{v} \equiv (v_1, v_2, \ldots, v_N)$ with $v_i \in \pm 1$. This microscopic configuration evolves with time under the constraints of K memory patterns. The μ-th memory pattern is denoted as $\xi^\mu \equiv (\xi_1^\mu, \xi_2^\mu, \ldots, \xi_N^\mu)$ with ξ_i^μ being a real-valued synaptic weight. The configuration energy $E(\mathbf{v})$ has the following general form

$$E(\mathbf{v}) = \sum_{\mu=1}^{K} E_\mu(\mathbf{v}) \ , \tag{4}$$

where $E_\mu(\mathbf{v})$ is the pattern energy associated with ξ^μ. The μ-th memory pattern favors configurations $\mathbf{v}$ that are similar to ξ^μ. Let us define the overlap, q_μ, between

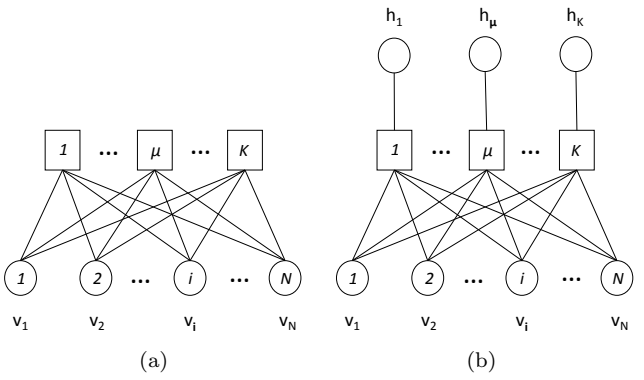


FIG. 1: The generalized Hopfield model and the corresponding Boltzmann machine. (a) The Hopfield model as a bipartite graph of variable nodes (circles) and factor nodes (squares). Each variable node denotes a neuron $i \in \{1, 2, \ldots, N\}$ and its state is $v_i \in \pm 1$. Each factor node $\mu \in \{1, 2, \ldots, K\}$ denotes the energy associated with a memory pattern $\xi^\mu \equiv (\xi_1^\mu, \xi_2^\mu, \ldots, \xi_N^\mu)$. The edge (i, μ) between variable node i and factor node μ has a synaptic weight ξ_i^μ. (b) The Hopfield model is equivalent to a semi-restricted Boltzmann machine containing N visible neurons and K hidden neurons. A hidden neuron is introduced for each memory pattern ξ^μ and it has a binary state $h_\mu \in \pm 1$.

configuration $\mathbf{v}$ and pattern ξ^μ as

$$q_\mu \equiv \frac{1}{N} \sum_{j=1}^{N} \xi_j^\mu v_j \ . \tag{5}$$

In this work we follow Ref. [16] and assume that the pattern energy E_μ is equal to the p-th power of q_μ, namely

$$E_\mu(\mathbf{v}) \equiv -\frac{N}{p!} (q_\mu)^p = -\frac{1}{p! N^{p-1}} \Big(\sum_{i=1}^{N} \xi_i^\mu v_i \Big)^p \ . \tag{6}$$

Notice that if the spin configuration $\mathbf{v}$ is very similar to ξ^μ and so q_μ is of order unity, the pattern energy $E_\mu(\mathbf{v}) \approx -\frac{N}{p!}$. On the other hand, if $\mathbf{v}$ is almost orthogonal to ξ^μ with $q_\mu \approx 0$, the pattern energy $E_\mu \approx 0$.

The exponent p in Eq. (6) is chosen to be $p = 2$ in the original Hopfield model [5], which leads to the two-body interaction model (1). It is well established that the storage capacity of associative memory is proportional to N^{p-1} [16, 21–23], so choosing $p \geq 3$ enhances the discrimination power of the neural network [17]. In the present work we consider both the case of $p = 2$ and the more general cases of $p \geq 3$. For simplicity we assume that the exponent p is the same for all the pattern energies, but in practical applications it might be helpful to allow p to be pattern-dependent. For example, if we assign an energy function $E_\nu(\mathbf{v}) = -\sum_{i=1}^{N} \xi_i^\nu v_i$ to a particular pattern ξ^ν, this is equivalent to introducing an external field ξ_i^ν on each neuron i.

The Hopfield model (4) can be described by a bipartite graph of variable nodes (circles) and factor nodes

(squares), with each variable node representing a neuron and each factor node representing the energy function of a pattern (Fig. 1(a)). Each pattern ξ^μ is stored distributively as the synaptic weights ξ_i^μ of the edges between factor nodes and variable nodes. In this paper we study unsupervised learning in the Hopfield model, so the synaptic weights will change during the learning process.

B. Equivalence to a semi-RBM

The Hopfield model as defined by Eq. (4) has only one layer of neurons. In the special case of $p = 2$ this model has been mapped to a RBM with another layer of hidden neurons, but the state of each hidden neuron has to be a continuous variable instead of being binary [18]. We find that it is actually easy to introduce a layer of hidden *binary* neurons to the general system (even for $p \neq 2$) to convert it to a semi-RBM. As shown in Fig. 1(b), we can attach a hidden neuron μ to the factor node of memory pattern ξ^μ and assign it a binary state $h_\mu \in \pm 1$. The partition function Z for this semi-RMB is defined as

$$Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \prod_{\mu=1}^{K} Z_\mu(h_\mu, \mathbf{v}) \;, \tag{7}$$

where $\mathbf{h} \equiv (h_1, h_2, \ldots, h_M)$ is a generic configuration of the K hidden neurons. At a given inverse temperature β, the Boltzmann factor $Z_\mu(h_\mu, \mathbf{v})$ associated with the factor node μ is defined in the following way:

1. If $\sum_i \xi_i^\mu v_i \geq 0$ (i.e., overlap $q_\mu \geq 0$) , then

$$Z_\mu(h_\mu, \mathbf{v}) = \begin{cases} e^{-\beta E_\mu(\mathbf{v})} - \frac{1}{2} & (h_\mu = +1) \;, \\ \\ \frac{1}{2} & (h_\mu = -1) \;. \end{cases} \tag{8}$$

2. If $\sum_i \xi_i^\mu v_i < 0$, then for exponent p being even

$$Z_\mu(h_\mu, \mathbf{v}) = \begin{cases} \frac{1}{2} & (h_\mu = +1) \;, \\ \\ \frac{1}{2} e^{-\beta E_\mu(\mathbf{v})} - \frac{1}{2} & (h_\mu = -1) \;; \end{cases} \tag{9}$$

while for p being odd

$$Z_\mu(h_\mu, \mathbf{v}) = \begin{cases} \frac{1}{2} e^{-2\beta E_\mu(\mathbf{v})} & (h_\mu = +1) \;, \\ \\ e^{-\beta E_\mu(\mathbf{v})} - \frac{1}{2} e^{-2\beta E_\mu(\mathbf{v})} & (h_\mu = -1) \;. \end{cases} \tag{10}$$

It is easy to check that the Boltzmann factor Z_μ is guaranteed to be positive. Notice that if the configurations $\mathbf{h}$ of the hidden neurons are summarized out in Eq. (7) the partition function is simplified to

$$Z = \sum_{\mathbf{v}} \exp\Big(-\beta \sum_{\mu=1}^{M} E_\mu(\mathbf{v})\Big) \;, \tag{11}$$

which is just the partition function for the Hopfield model (4). The Hopfield model (4) is therefore equivalent to a Boltzmann machine with a layer of visible neurons and a layer of hidden neurons. Each memory pattern ξ^μ is associated with a many-body interaction in the Boltzmann machine which involves N visible neurons and one hidden neuron, and its energy $\mathcal{E}_\mu$ is

$$\mathcal{E}_\mu(h_\mu, \mathbf{v}) \equiv -\frac{1}{\beta} \ln Z_\mu(h_\mu, \mathbf{v}) \;. \tag{12}$$

Notice that, given a configuration $\mathbf{h}$ of the K hidden neurons, the states of the N visible neurons are still strongly depend on each other. This non-symmetry between the hidden and visible layers might actually be a beneficial property in practical learning tasks [24].

Given a configuration $\mathbf{v}$ of the N visible neurons, the states of the K hidden neurons are conditionally independent and they are easy to sample. If the overlap between $\mathbf{v}$ and ξ^μ is positive ($q_\mu > 0$), the mean value of h_μ is

$$\big\langle h_\mu \big\rangle = \frac{e^{-\beta E_\mu(\mathbf{v})} - 1}{e^{-\beta E_\mu(\mathbf{v})}} = 1 - e^{\beta E_\mu(\mathbf{v})} \;, \tag{13}$$

which is also positive. On the other hand, if $q_\mu < 0$, then the mean value of h_μ is negative and is expressed as

$$\big\langle h_\mu \big\rangle = \begin{cases} e^{\beta E_\mu(\mathbf{v})} - 1 & (p \;\; \text{even}) \;, \\ \\ e^{-\beta E_\mu(\mathbf{v})} - 1 & (p \;\; \text{odd}) \;. \end{cases} \tag{14}$$

The state h_μ of the hidden neuron can be understood as an output signal: if $h_\mu = 1$, the output neuron μ judges that the input configuration $\mathbf{v}$ is similar to the memory pattern ξ^μ; while if $h_\mu = -1$, the input configuration $\mathbf{v}$ is judged to be distinct from the memory pattern. We can regard a configuration $\mathbf{h}$ of the K hidden neurons as a representation of the visible configuration $\mathbf{v}$. If needed, this representation $\mathbf{h}$ can serve as the input to another Hopfield neural network for further processing. Therefore the semi-RBM (7) can be easily extended into a deep (multilayered) neural network.

III. TRAINING ALGORITHM

Given a collection of input spin configurations $\mathbf{v}$ (the observed data) as the learning task, there are many unsupervised learning methods [10] to train the Hopfield model (4). Here we adopt the Contrastive Divergence (CD) algorithm of Hinton [12, 14] for its simplicity. The numerical results of the next two sections are essentially unchanged with other more sophisticated learning methods such as pseudo-likelihood maximization [25] and mean-field inference algorithms [26, 27].

The associative-memory network (4) is a typical energy-based model which assigns an energy to each microscopic configuration. Learning corresponds to modifying the energy function so that the observed (input)

spin configurations are the most typical configurations of the model. At inverse temperature β and with the set of memory parameters $\{\xi^\mu\}$, the probability of observing a particular configuration $\mathbf{v}$ is

$$P(\mathbf{v}) \equiv \frac{1}{Z} \exp\Big[-\beta \sum_{\mu=1}^{M} E_\mu(\mathbf{v})\Big] , \tag{15}$$

where Z is the partition function defined in Eq. (11). Our goal is to maximize the probabilities $P(\mathbf{v})$ for the observed sample configurations $\mathbf{v}$. Since the memory parameters ξ_i^μ are adjusted during the learning process, without loss of generality we set $\beta \equiv p! N^{p-1}$ to simplify the notation. Under this convention, the loss function (also called the objective function) $\mathcal{L}(\mathbf{v}) \equiv \ln P(\mathbf{v})$ is expressed as

$$\mathcal{L}(\mathbf{v}) = \sum_{\mu=1}^{K} \Big(\sum_{i=1}^{N} \xi_i^\mu v_i \Big)^p - \ln \Big[\sum_{\mathbf{v}'} \exp\Big[\sum_{\mu=1}^{K} \Big(\sum_{i=1}^{N} \xi_i^\mu v_i' \Big)^p \Big] \Big] . \tag{16}$$

We perform stochastic gradient ascent on the loss function to guide the evolution of the memory parameters ξ_i^μ. Given an observed configuration $(v_1, v_2, \ldots, v_N)$ and trying to increase its log-likelihood of being observed, the updating rule for ξ_i^μ is then

$$\xi_i^\mu \;\leftarrow\; \xi_i^\mu + \epsilon\, \Delta\xi_i^\mu - c\, \xi_i^\mu . \tag{17}$$

Here ϵ is the learning rate and is chosen to be a small positive value; c is a regularization coefficient to avoid overfitting; and the increment $\Delta\xi_i^\mu$ is proportional to $\frac{\partial \mathcal{L}}{\partial \xi_i^\mu}$:

$$\Delta\xi_i^\mu \equiv v_i \Big(\sum_{i=1}^{N} \xi_i^\mu v_i \Big)^{p-1} - \sum_{\mathbf{v}'} P(\mathbf{v}') v_i' \Big(\sum_{i=1}^{N} \xi_i^\mu v_i' \Big)^{p-1} . \tag{18}$$

The first term of Eq. (18) is straightforward to compute. The second term, however, is much harder as it involves sampling configurations $\mathbf{v}'$ from the probability distribution (15). In the CD algorithm, sampling from $P(\mathbf{v}')$ is achieved by Monte Carlo simulations [14]. At the beginning of each learning epoch, we first divide the training samples uniformly at random into equal-sized minibatches, each of which containing a small number of samples. To learn from a minibatch B, we first obtain the average of the first term of Eq. (18) on the samples in B. Then, starting from each spin configuration $\mathbf{v}$ in B, we generate a Markovian chain of spin configurations and record the final configuration $\mathbf{v}'$ at the R-th iteration step. Each iteration step consists of N elementary updating trials; in each of such trials a neuron i is chosen uniformly at random from the N neurons and its spin state is updated by the conventional Metropolis importance sampling rule [28] (all the other spins are kept unchanged). After a set of model configurations $\mathbf{v}'$ is sampled through this process, the second term of Eq. (18) is then approximated by averaging over all these model configurations (without the reweighting factor $P(\mathbf{v}')$, of course).

After all the minibatches of sample configurations are examined once by the above-mentioned updating process, one epoch of the learning process is finished. Then another learning epoch is carried on with newly assembled minibatches, until all the memory patterns ξ^μ are stabilized and the summed value of $\mathcal{L}(\mathbf{v})$ over all the input configurations $\mathbf{v}$ no longer increases with further learning.

In all the numerical simulations of the present work, the learning rate is fixed to $\epsilon = 0.05$, the regularization coefficient is fixed to $c = 0.05$, and the minibatch size is fixed to be 200. The total number of learning epochs is set to be $T = 20$ for $p = 2$ and to be $T = 50$ for $p = 5$ unless otherwise specified. The iteration step R of each Monte Carlo trajectory is set to be $R = T$ unless otherwise specified. Our choice of parameter values may not necessarily be optimal in terms of learning efficiency, but we have checked that the numerical results of the next two sections are robust to small variations in these learning parameters.

IV. APPLICATION ON RANDOM DATASETS

We first apply the Hopfield learning system to randomly generated datasets. A random dataset is generated by two steps:

1. Generate S_0 random seed configurations $\mathbf{v}^{(m)} \equiv (v_1^{(m)}, v_2^{(m)}, \ldots, v_N^{(m)})$, with $m = 1, 2, \ldots, S_0$. The seed configurations are drawn independently and completely at random from the 2^N possible candidate spin configurations. Each seed spin value $v_i^{(m)}$ is independent of all the other seed spin values, and $v_i^{(m)} = \pm 1$ with equal probability.

2. Generate S_1 perturbed samples $\mathbf{v}$ for each seed configuration $\mathbf{v}^{(m)}$. The spin of neuron i in each sample is $v_i = v_i^{(m)}$ with probability $1 - f$ and is $v_i = -v_i^{(m)}$ with the remaining probability f. The adjustable parameter f is the noise level.

By construction these samples form S_0 clusters. The total number of sample configurations is then $S = S_0 \times S_1$ and these samples serve as the input data for the unsupervised learning task.

In our numerical simulations the number of neurons is fixed to $N = 784$ and the number of perturbed copies for each seed configuration is set to $S_1 = 1000$. We change the degree of learning difficulty by changing the seed number S_0.

A. Number of planted clusters being $S_0 = 20$

At a given noise level f we generate $S = 20,000$ perturbed configuration samples $\mathbf{v}$ from $S_0 = 20$ random seed configurations. Unsupervised learning (with

$K = 500$ memory patterns and energy exponent $p = 2$) is then performed on this dataset.

To measure the similarity of two memory patterns ξ^μ and ξ^ν, we define the distance $d_{\mu\nu}$ and the overlap $q_{\mu\nu}$ between them as

$$d_{\mu\nu} = \frac{1}{N}\sum_{i=1}^{N}\left|\xi_i^\mu - \xi_i^\nu\right| , \tag{19a}$$

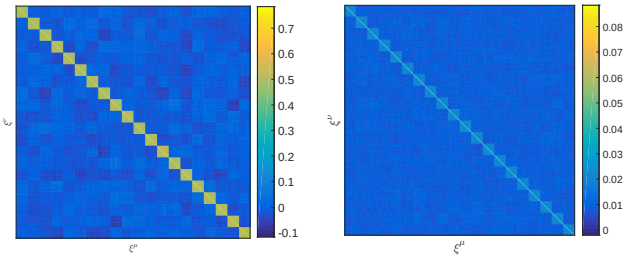
$$q_{\mu\nu} = \frac{1}{N}\sum_{i=1}^{N}\xi_i^\mu\xi_i^\nu . \tag{19b}$$

With these $K(K-1)/2$ distance values $d_{\mu\nu}$, we can perform a hierarchical minimum-variance clustering analysis on the K learned memory patterns [29–31].

The results of this clustering analysis are shown in Fig. 2(a) and Fig. 2(b) for noise level $f = 0.10$ and $f = 0.45$, respectively. We see that the learned memory patterns faithfully manifest the hidden planted structure of the sample configurations, even at very high noise level $f = 0.45$. The 500 memory patterns form 20 different groups, with each group containing ≈ 25 elements. We have also verified that the memory patterns of each group have relatively high similarity with one of the seed configurations $\mathbf{v}^{(m)}$ but are distinct from the other seed configurations, see Fig. 3(a). The K learned memory patterns can therefore be interpreted as prototypes of the S input sample configurations.

An interesting question is: can we reconstruct the S_0 seed configurations solely from the K learned memory patterns? Let us denote by Ξ_m the set of memory patterns belonging to one of the S_0 clusters (blocks) in the overlap matrix. A direct way to infer the corresponding seed configuration is to define an averaged memory $\overline{\xi^{(m)}} \equiv (\overline{\xi_1^{(m)}}, \overline{\xi_2^{(m)}}, \ldots, \overline{\xi_N^{(m)}})$, with

$$\overline{\xi_i^{(m)}} \equiv \frac{1}{|\Xi_m|}\sum_{\xi\in\Xi_m}\xi_i , \tag{20}$$



(a) (b)

FIG. 2: The overlap matrix ($q_{\mu\nu}$) between the memory patterns (ξ^μ and ξ^ν) that are learned through the CD algorithm. The Hopfield model contains $K = 500$ memory patterns and the energy exponent is set to be $p = 2$. The training dataset has 20,000 configuration samples $\mathbf{v}$ of dimensionality $N = 784$, which are generated from $S_0 = 20$ random seed configurations under noise level $f = 0.10$ (a) or $f = 0.45$ (b).

where $|\Xi_m|$ denotes the cardinality of set Ξ_m. The value $\overline{\xi_i^{(m)}}$ is the averaged value of ξ_i^μ for all the elements of set Ξ_m. If we take the sign of $\overline{\xi_i^{(m)}}$ as the prediction on the i-th spin of the seed configuration, we find that the reconstruction error is zero for $f \leq 0.20$ but goes beyond 10% for $f \geq 0.35$, see Fig. 3(a).

Given the set Ξ_m of memory patterns belonging to the same cluster, we can define a $N\times N$ memory matrix $\mathcal{J}^{(m)}$ through the following expression for all its elements:

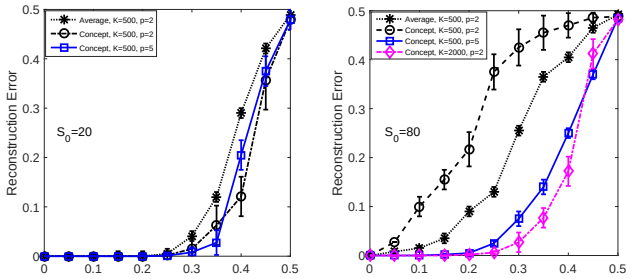
$$\mathcal{J}_{ij}^{(m)} \equiv \frac{1}{|\Xi_m|}\sum_{\xi\in\Xi_m}\xi_i\xi_j , \qquad i,j\in[1,N] . \tag{21}$$

Notice the summation is restricted to memory patterns in Ξ_m. The matrix $\mathcal{J}^{(m)}$ is real-valued and is symmetric. We find that the largest eigenvalue ($\lambda_1^{(m)}$) of the matrix $\mathcal{J}^{(m)}$ is positive and is much larger than all the other eigenvalues. The leading (first) eigenvector, $\hat{\xi}^{(m)} \equiv (\hat{\xi}_1^{(m)}, \hat{\xi}_2^{(m)}, \ldots, \hat{\xi}_N^{(m)})$, of $\mathcal{J}^{(m)}$ is obtained by solving the following eigenproblem

$$\mathcal{J}^{(m)}\hat{\xi}^{(m)} = \lambda_1^{(m)}\hat{\xi}^{(m)} . \tag{22}$$

Without loss of generality we may require $\hat{\xi}^{(m)}$ to have lenth $\sqrt{N}$, namely $\sum_{i=1}^{N}\left(\hat{\xi}_i^{(m)}\right)^2 = N$.

In this work we refer to $\hat{\xi}^{(m)}$ as a "concept" (or an idealized prototype) extracted from the learned memory patterns. Since the memory patterns form S_0 clusters (Fig. 2), we get S_0 different concepts.



(a) (b)

FIG. 3: Inferring the random seed configurations from the K learned memory patterns. The number of seed configurations is $S_0 = 20$ in (a) and $S_0 = 80$ in (b), while the total number of sample configurations is $S = 20,000$ in (a) and $S = 80,000$ in (b). The noise level of sample configurations is f. Reconstruction error is defined as the fraction of incorrectly predicted spin values; the error bars mark the standard deviations of reconstruction error on the S_0 seed configurations. Direct inference based on the averaged memory patterns (Average, Eq. (20))are compared with spectral inference based on the extracted concepts (Concept, Eq. (22)).The number of neurons is $N = 784$; the energy exponent is $p = 2$ (except for the square points, for which $p = 5$); the number of memory patterns is $K = 500$ (except for the diamond points, for which $K = 2000$).

We find that, when the noise level $f \leq 0.25$, the sign vectors of the S_0 extracted concepts are identical to the S_0 seed configurations (Fig. 3(a)). At high noise levels ($f \geq 0.30$) we fail to perfectly recover all the seed configurations from the concepts, but still the extracted concepts contain more information about the seed configurations than the averaged memory patterns $\overline{\xi^{(m)}}$. The performance of reconstruction can be improved by increasing the number K of memory patterns (see next subsection) or by choosing a larger energy exponent p (e.g., taking $p = 5$, see square points of Fig. 3(a)).

B. Number of planted clusters being $S_0 = 80$

We then increase the number of random seed configurations to $S_0 = 80$ to see whether unsupervised learning is still feasible. At a given noise level f we generate $S = 80,000$ perturbed sample configurations from the S_0 seed configurations.

For the Hopfield model with energy exponent $p = 2$ and number of memory patterns $K = 500$, we observe that the learned memory patterns do not form 80 perfect clusters but instead form a few number of merged larger clusters (Fig. 4(a)), indicating that the system fails to fully capture the hidden structure of the input data. However, the learning performance improves if we add more memory patterns to the network. At $K = 2000$, for example, the learned memory patterns form 80 perfect clusters (Fig. 4(b)). Error-free reconstruction of the 80 seed configurations is achieved from the extracted 80 concepts when the noise level $f \leq 0.20$, see diamond points of Fig. 3(b).

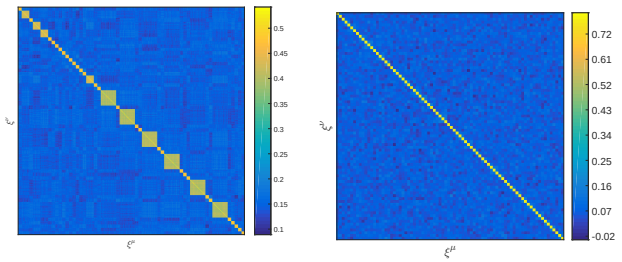
We can also improve the performance of the Hopfield model by choosing a larger value of the energy exponent p. If we set $p = 5$, for example, the learned memory patterns will again form 80 clusters of approximately equal size even if the number of memory patterns is kept to the smaller value of $K = 500$ (each cluster then contains 6.25 memory patterns on average). Error-free reconstruction of all the random seed configurations is again achieved for noise levels $f \leq 0.20$, see square points of Fig. 3(b).

C. Number of planted clusters being $S_0 = 120$

We now further increase the learning difficulty and set the number of random seed configurations to be $S_0 = 120$. At a given noise level f we generate $S = 120,000$ perturbed sample configurations from these seed configurations. A Hopfield network containing K memory patterns then learns from these samples. The energy exponent is set to be $p = 2$ for simplicity.

When $K \leq 4000$ we find the Hopfield network does not completely succeed in forming 120 clusters of memory patterns. Instead some of the seed configurations have no memory pattern to represent them, while some other seed configurations are represented by too many memory patterns. The overlap matrix of the memory patterns has similar structure to that of Fig. 4(a).

However, if a larger K value is chosen, the Hopfield network again achieves a perfect representation of the input configurations. As demonstrated in Fig. 5 for $K = 4800$, the learned memory patterns form 120 clusters of ap-



(a) (b)

FIG. 4: The overlap matrix ($q_{\mu\nu}$) between the memory patterns (ξ^μ and ξ^ν) that are learned through the CD algorithm. The training dataset has 80,000 configuration samples of dimensionality $N = 784$, which are generated based on $S_0 = 80$ random seed configurations at noise level $f = 0.10$. The energy exponent of the Hopfield model is $p = 2$, while the number of memory patterns is $K = 500$ in (a) and $K = 2000$ in (b). The learned memory patterns form 80 clusters in (b), with each cluster containing ≈ 25 patterns.

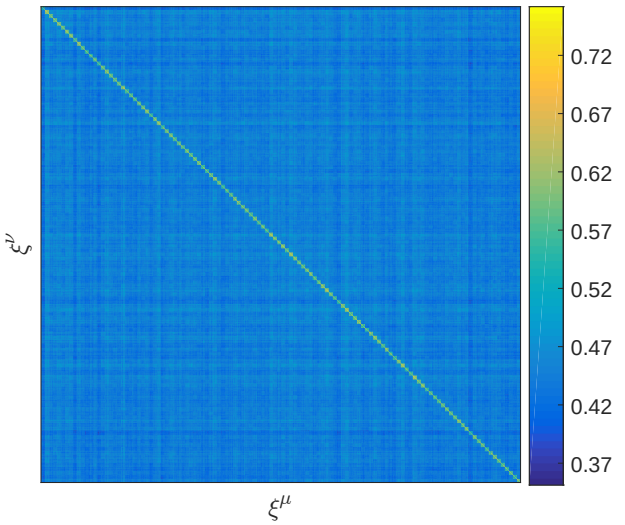


FIG. 5: The overlap matrix ($q_{\mu\nu}$) between the memory patterns (ξ^μ and ξ^ν) that are learned through the CD algorithm. The training dataset has 120,000 configuration samples of dimensionality $N = 784$, which are generated based on $S_0 = 120$ random seed configurations at noise level $f = 0.10$. The energy exponent of the Hopfield model is $p = 2$, while the number of memory patterns is $K = 4800$. The learned memory patterns form 120 clusters, each of which containing ≈ 40 patterns.

proximately equal sizes, with each cluster corresponding to one seed configuration.

Although the number S_0 of random seed configurations is exceeding the storage capacity $0.14N$ $(= 110)$ predicted in Ref. [6], it is not a real contradiction, since the synaptic weights ξ_i^μ take real values here instead of being binary and the network is not required to memorize any particular binary configurations [32]. But we indeed observe that the Hopfield network needs more learning epochs T to finish this difficult task (here we take $T = 50$). Consistent with the experience in [33], we find that the learning performance can be further improved if the minibatch size of the CD algorithm is set to smaller values.

V. APPLICATION ON HANDWRITTEN DIGITS

As a real-world application we consider the MNIST dataset of handwritten digits [34]. This dataset contains $60,000$ training images. Each image is square-shaped and is comprised of $N = 28 \times 28 = 784$ pixels with integer grey level ranging from 0 to 255. In the present work we consider the black-white image version by mapping all the positive pixel values to $+1$ and the zero pixel value to -1. Therefore each training image is a spin configuration of dimension N. Each input image corresponds to one of the ten digits $0, 1, \ldots, 9$ (but this label is not used in unsupervised learning). We use these sample configurations to train a Hopfield network. The energy exponent of the network is fixed to $p = 2$ and the number of memory patterns is set to be $K = 500$.

Similar to the observations in the preceding section, we find that the learned memory patterns form 10 well-distinguished clusters (Fig. 6). The memory patterns in each cluster are all prototypic handwritten forms of the same digit (see Fig. 7 for some examples). Due to the fact that the ten digits appear in the training dataset with almost equal frequency, the sizes of the 10 clusters of memory patterns are approximately equal.

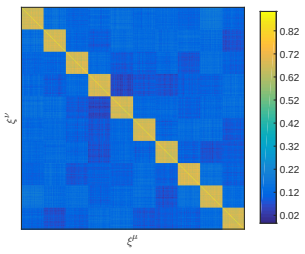


FIG. 6: The overlap matrix $(q_{\mu\nu})$ between the memory patterns (ξ^μ and ξ^ν) that are learned through the CD algorithm on the MNIST dataset of handwritten digits, which contains $60,000$ samples of dimensionality $N = 784$. The energy exponent of the Hopfield model is $p = 2$, and the number of memory patterns is $K = 500$.

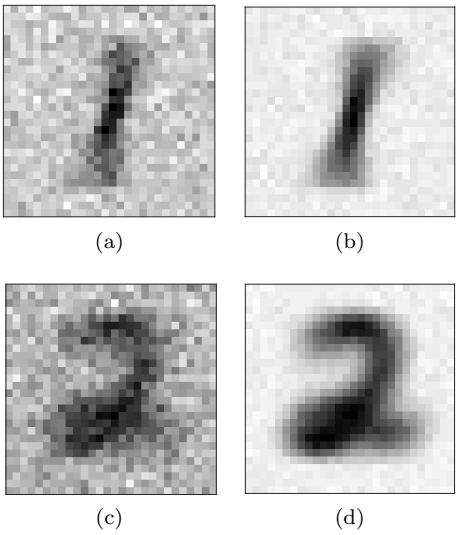


FIG. 7: Examples of learned memory patterns by the Hopfield network (with $p = 2$ and $K = 500$) on the MNIST dataset. The total number of learning epochs is $T = 20$. The iteration step R of the Monte Carlo routine in the CD algorithm is $R = 1$ in (a) and (c) and $R = 20$ in (b) and (d). Better learning is achieved at $R = 20$.

By applying Eq. (21) and Eq. (22) on these clusters we can obtain ten concepts (idealized prototypes) as the most succinct summary of the $60,000$ input images, see Fig. 8. Indeed these are the abstractions formed by the Hopfield neural network after learning from the input images. It may be tempting for us to take these ten concepts as the 'ideal' (and most beautiful) handwritten digits [35].

It was observed by Krotov and Hopfield [16] that, when the Hopfield model with energy exponent $p = 2$ was used to classify the images of the MNIST dataset into ten groups by supervised learning, the learned memory patterns do not look like digits, they are features of these digits but not prototypes. These authors estimated that a feature-to-prototype transition in the learned memory patterns occurs at a rather large value of $p \approx 20$ [16] in their supervised learning simulations. In our simulations of unsupervised learning, however, we find that the learned memory patterns are always prototypes for any $p \geq 2$, even when the iteration step of the Monte Carlo routine in the CD algorithm is set to be the minimum value of $R = 1$ (see left panel of Fig. 7). Such a big difference might be due to the fact that supervised learning and unsupervised learning have very different optimization objectives.

VI. CONCLUSION AND DISCUSSIONS

Inspired by the very recent work of Krotov and Hopfield [16, 17], in the present paper we studied unsuper-

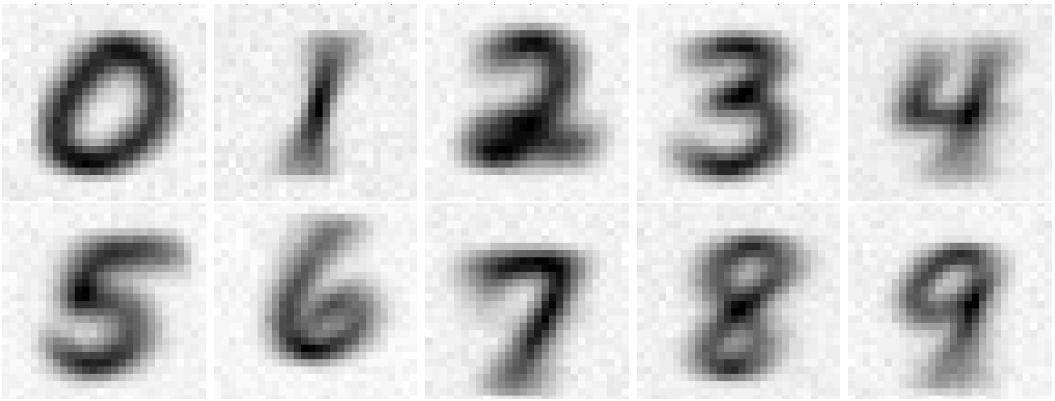


FIG. 8: The ten concepts of handwritten digits learned by the Hopfield model (energy exponent $p = 2$ and number of memory patterns $K = 500$) on the MNIST dataset. Each concept is the first eigenvector of one memory matrix $\mathcal{J}^{(m)}$ [Eq. (21)].

vised learning on the (generalized) Hopfield neural network and discussed the meaning of the learned memory parameters and their potential usefulness in the era of big data and deep learning. We demonstrated that the Hopfield network stores a set of prototypes of the input sample configurations in its edge weights, and these prototypes form different clusters if the input configurations have an intrinsic community structure. We further suggested that a small set of concepts (or idealized prototypes) can be extracted from the learned memory patterns to reveal the most essential properties of the input configurations. These results may facilitate application of the Hopfield neural network to high-dimensional data clustering, pattern recognition and reconstruction.

We also pointed out that the Hopfield model is equivalent to a semi-restricted Boltzmann machine (Fig. 1(b)). Given an input binary configuration $\mathbf{v}$, some binary configurations $\mathbf{h}$ of the hidden neurons can be generated by the learned semi-RBM in a straightforward way. These output configurations can be regarded as representations of the input configurations $\mathbf{v}$ and, if necessary, they can serve as inputs to another Hopfield network for further processing.

The unsupervised-learning performance of the Hopfield network improves when a larger exponent p of the pattern energy function is chosen. But a quite encouraging observation was that, even if we simply set $p = 2$ the network can still achieve unsupervised prototype learning. A value of $p = 2$ might be favorable than a large value of p (say $p \geq 30$ as in [16]) because it takes much fewer learning epochs to train the network. Indeed an energy function (6) with $p = 30$ is certainly not biologically natural.

When the number S_0 of clusters in the input configurations increases, we experienced that the number K of memory patterns in the Hopfield network needs to be increased accordingly. This is intuitively reasonable, since the statistical property of a cluster of configurations can only be well represented by a sufficient large number of prototypes in the memory. When the dimension N of the input configurations is large, we guess that K should scale at least linearly with S_0 (e.g., $K \geq 30S_0$). Rigorous theoretical studies on the quantitative relationship between the cluster number S_0 and the memory size K need to be carried out in the near future.

For simplicity we assumed the the state v_i of each visible neuron i is binary. This assumption can be relaxed in practical applications. For the most general situation of v_i being a continuous variable, we may define the pattern energy E_μ in Eq. (4) to be an increasing function of the Euclidean distance between $\mathbf{v}$ and the memory pattern ξ^μ. In the simplest situation of binary configurations, it might also be helpful to introduce a threshold parameter q_μ^0 and modify the pattern energy expression (6) into $E_\mu(\mathbf{v}) \propto \left(q_\mu - q_\mu^0\right)^p$. The optimal threshold values q_μ^0 can also be learned from the input configurations.

In the present work, the memory parameters ξ_i^μ in Eq. (5) take real values. It may also be very interesting to consider discrete synaptic weights (e.g., $\xi_i^\mu \in \{-1, 0, +1\}$) [36–40]. Discrete synaptic weights are quite desirable for theoretical treatments of unsupervised learning, but they may pose additional challenges for learning algorithms. Mean-field inference methods inspired by spin glass physics might be very helpful in this respect [26, 41, 42].

The datasets studied in this paper are relatively simple, they do not reflect the true complexity of real-world high-dimensional data, which usually have very rich hierarchical structures [18, 43, 44]. A single-layered Hopfield neural network is of course not sufficient for most practical applications. A natural next step is to investigate unsupervised learning in multilayered Hopfield neural networks.

Acknowledgement

We thank Pan Zhang for collaboration. The numerical simulations were carried out at the Tianhe-2 platform of the National Supercomputer Center in Guangzhou and also at the HPC cluster of ITP-CAS. We acknowledge technical support from Paratera (Beijing). This research was partially supported by the National Basic Research Program of China (grant number 2013CB932804) and the National Natural Science Foundation of China (grant numbers 11121403 and 11647601).

[1] M. F. Bear, B. W. Connors, and M. A. Paradiso. *Neuroscience: Exploring the Brain*. Lippincot Williams and Wilkins Inc., Philadelphia, USA, 2nd edition, 2001.

[2] D. O. Hebb. *The Organization of Behavior*. Wiley, New York, 1949.

[3] S.-I. Amari. Neural theory of association and concept-formation. *Biol. Cybernetics*, 26:175–185, 1977.

[4] B. Müller and J. Reinhardt. *Neural Networks: An Introduction*. Springer-Verlag, Berlin, 1991.

[5] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA*, 79:2554–2558, 1982.

[6] D. J. Amit, H. Gutfreund, and H. Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Phys. Rev. Lett.*, 55:1530–1533, 1985.

[7] H. Huang. Reconstructing the Hopfield network as an inverse Ising problem. *Phys. Rev. E*, 81:036104, 2010.

[8] H. Huang. State sampling dependence of Hopfield network inference. *Commun. Theor. Phys.*, 57:169–172, 2011.

[9] H. Huang. Sparse Hopfield network reconstruction with ℓ_1 regularization. *Eur. Phys. J. B*, 86:484, 2013.

[10] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, Cambridge, MA, 2016.

[11] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 1, pages 194–281, Cambridge, MA, 1986. MIT Press.

[12] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Naural Computation*, 14:1711–1800, 2002.

[13] G. E. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313:504–507, 2006.

[14] G. E. Hinton. A practical guide to training restricted Boltzmann machines. *Lect. Notes Comput. Sci.*, 7700:599–619, 2012.

[15] K. Nagatani and M. Hagiwara. Restricted Boltzmann machine associative memory. In *Proceedings of 2014 International Joint Conference on Neural Networks*, pages 3745–3750, Beijing, 2014. IEEE.

[16] D. Krotov and J. J. Hopfield. Dense associative memory for pattern recognition. *Adv. Neural Inf. Process. Sys.*, 29:1172–1180, 2016.

[17] D. Krotov and J. J. Hopfield. Dense associative memory is robust to adversarial inputs. *eprint*, arXiv:1701.00939, 2017.

[18] M. Mézard. Mean-field message-passing equations in the Hopfield model and its generalizations. *Phys. Rev. E*, 95:022117, 2017.

[19] R. Kruse, C. Borgelt, C. Braune, S. Mostaghim, and M. Steinbrecher. *Computational Intelligence*. Springer, London, 2016.

[20] Y. LeCun, Y. Bengio, and G. E. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[21] H. H. Chen, Y. C. Lee, G. Z. Sun, H. Y. Lee, T. Maxwell, and C. L. Giles. High order correlation model for associative memory. *AIP Conf. Proc.*, 151:86–99, 1986.

[22] D. Psaltis and C. H. Park. Nonlinear discriminant functions and associative memories. *AIP Conf. Proc.*, 151:370–375, 1986.

[23] P. Baldi and S. S. Venkatesh. Number of stable points for spin-glasses and neural networks of higher orders. *Phys. Rev. Lett.*, 58:913–916, 1987.

[24] S. Osindero and G. E. Hinton. Modeling image patches with a directed hierarchy of Markov random fields. In *Adv. Neural Inf. Proc. Syst.*, pages 1121–1128, 2008.

[25] E. Aurell and M. Ekeberg. Inverse Ising inference using all the data. *Phys. Rev. Lett.*, 108:090201, 2012.

[26] M. Opper and D. Saad, editors. *Advanced Mean Field Methods: Theory and Practice*. MIT Press, Cambridge, MA, 2001.

[27] T. Tanaka. Mean-field theory of Boltzmann machine learning. *Phys. Rev. E*, 58:2302–2310, 1998.

[28] M. E. J. Newman and G. T. Barkema. *Monte Carlo Methods in Statistical Physics*. Oxford University Press, New York, 1999.

[29] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1988.

[30] W. Barthel and A. K. Hartmann. Clustering analysis of the ground-state structure of the vertex-cover problem. *Phys. Rev. E*, 70:066120, 2004.

[31] H. J. Zhou. Solution space heterogeneity of the random K-satisfiability problem: Theory and simulations. *J. Phys.: Conf. Series*, 233:012011, 2010.

[32] Z. Zhou and H. Zhao. Improvement of the Hopfield neural network by MC-adaption rule. *Chinese Phys. Lett.*, 23:1402–1405, 2006.

[33] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *eprint*, arXiv:1609.04836, 2016.

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86:2278–2324, 1998.

[35] A. J. Rubenstein, J. H. Langlois, and L. A. Roggman. What makes a face attractive and why: The role of averageness in defining facial beauty. In G. Rhodes and L. A. Zebrowitz, editors, *Facial attractiveness: Evolutionary, Cognitive, and Social Perspectives*, pages 1–33. Ablex Publishing, Westport, CT, USA, 2002.

[36] A. Braunstein and R. Zecchina. Learning by message passing in networks of discrete synapses. *Phys. Rev. Lett.*, 96:030201, 2006.

[37] P. Sollich, D. Tantari, A. Annibale, and A. Barra. Ex-

tensive parallel processing on scale-free networks. *Phys. Rev. Lett.*, 113:238106, 2014.

[38] P. Zhang. Nonbacktracking operator for the Ising model and its applications in systems with multiple states. *Phys. Rev. E*, 91:042120, 2015.

[39] A. Barra, G. Genovese, P. Sollich, and D. Tantari. Phase transitions in restricted Boltzmann machines with generic priors. *eprint*, arXiv:1612.03132, 2016.

[40] H. Huang. The role of zero synapses in unsupervised feature learning. *eprint*, arXiv:1703.07943, 2017.

[41] L. Zdeborová and F. Krzakala. Statistical physics of inference: thresholds and algorithms. *Adv. Phys.*, 65:453–552, 2016.

[42] H. C. Nguyen, R. Zecchina, and J. Berg. Inverse statistical problems: from the inverse Ising problem to data science. *eprint*, arXiv:1702.01522, 2017.

[43] P. Mehta and D. J. Schwab. An exact mapping between the variational renormalization group and deep learning. *eprint*, arXiv:1410.3831, 2014.

[44] H. W. Lin and M. Tegmark. Critical behavior from deep dynamics: A hidden dimension in natural language. *eprint*, arXiv:1606.06737, 2016.