Deep learning topological invariants of band insulators

Ning Sun,[1] Jinmin Yi,[1,2] Pengfei Zhang,[1] Huitao Shen,[3] and Hui Zhai[1,4]
[1]*Institute for Advanced Study, Tsinghua University, Beijing, 100084, China*
[2]*Department of Physics, Peking University, Beijing, 100871, China*
[3]*Department of Physics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*
[4]*Collaborative Innovation Center of Quantum Matter, Beijing, 100084, China*

(Received 8 June 2018; published 2 August 2018)

In this work we design and train deep neural networks to predict topological invariants for one-dimensional four-band insulators in AIII class whose topological invariant is the winding number, and two-dimensional two-band insulators in A class whose topological invariant is the Chern number. Given Hamiltonians in the momentum space as the input, neural networks can predict topological invariants for both classes with accuracy close to or higher than 90%, even for Hamiltonians whose invariants are beyond the training data set. Despite the complexity of the neural network, we find that the output of certain intermediate hidden layers resembles either the winding angle for models in AIII class or the solid angle (Berry curvature) for models in A class, indicating that neural networks essentially capture the mathematical formula of topological invariants. Our work demonstrates the ability of neural networks to predict topological invariants for complicated models with local Hamiltonians as the only input, and offers an example that even a deep neural network is understandable.

DOI: 10.1103/PhysRevB.98.085402

I. INTRODUCTION

Machine learning has achieved huge success recently in industrial applications. In particular, deep learning prevails for its performance in several different fields including image recognition and speech transcription [1–8]. In terms of applications in assisting academic research, aside from analyzing experimental data in high-energy physics [9,10] and astrophysics [11–14], progresses have also been made on recognizing phases of matter [15–45], accelerating Monte Carlo simulations [46–52], and extracting relations between many-body wave functions, entanglement, and neural networks [53–58]. Among these progresses, one challenging and interesting problem is to extract global topological features from local inputs, for instance, by supervised training a neural network, and to understand how the neural network works.

In Ref. [15] a convolutional neural network is trained to predict the topological invariant for band insulators with high accuracy. The highlights of that work are twofold. First, only local Hamiltonians are used as the input and no human knowledge is used as a prior. Second, by analyzing the neural network after training, it is found the formula fitted by the neural network is precisely the same as the mathematical formula for the winding number. However, the limitations of Ref. [15] are also twofold. Only one-dimensional models in AIII class whose topological invariants are the winding numbers are considered. Moreover, only two-band models are considered.

In this work we extend the realm of the previous work to more sophisticated scenarios, including (i) one-dimensional models in AIII class with more than two bands and (ii) two-dimensional two-band models in A class. We find that in both cases, the neural network can predict topological invariants with high accuracy, even for testing Hamiltonians whose topological numbers are beyond those in the training set. Similar to Ref. [15], we use local Hamiltonians as the input and do not feature engineer the input data with any human knowledge. Also, the design of the neural network architecture follows general principles, without specifically making use of the prior understanding of topological invariants. The only knowledge we explicitly exploit about these models is the translational symmetry, as we choose convolutional layers as the building blocks of our neural networks. Convolutional layers respect the translational symmetry by construction and reduce the redundancy in the parametrization [59].

Learning topological invariants of these two models is significantly harder than that in Ref. [15], as the mathematical formula of topological invariants in these models are intrinsically more complicated [see Eqs. (2) and (7)] and the sizes of the input data are much larger. Consequently, to guarantee a good performance, neural networks used in this work are much deeper than the one used in Ref. [15]. As shown in Fig. 1, there are more than nine hidden layers in each neural network. Because the neural network becomes more complicated, it becomes more difficult to analyze how the neural network works. Nevertheless, we show that the intermediate output of a certain hidden layer is, for case (i) the local winding angle, and for case (ii) the local Berry curvature—both are the integrands in the mathematical formula of the corresponding topological invariant. In this way we demonstrate that the complicated function fitted by the neural network is essentially the same as the mathematical formula for the topological invariant.

The paper is organized as follows. In Sec. II we train a neural network to learn the winding number of one-dimensional four-band models in AIII class. After introducing the model Hamiltonian and the mathematical formula of the winding number, we present our neural network in detail and report its performance. We then analyze the mechanism of why the

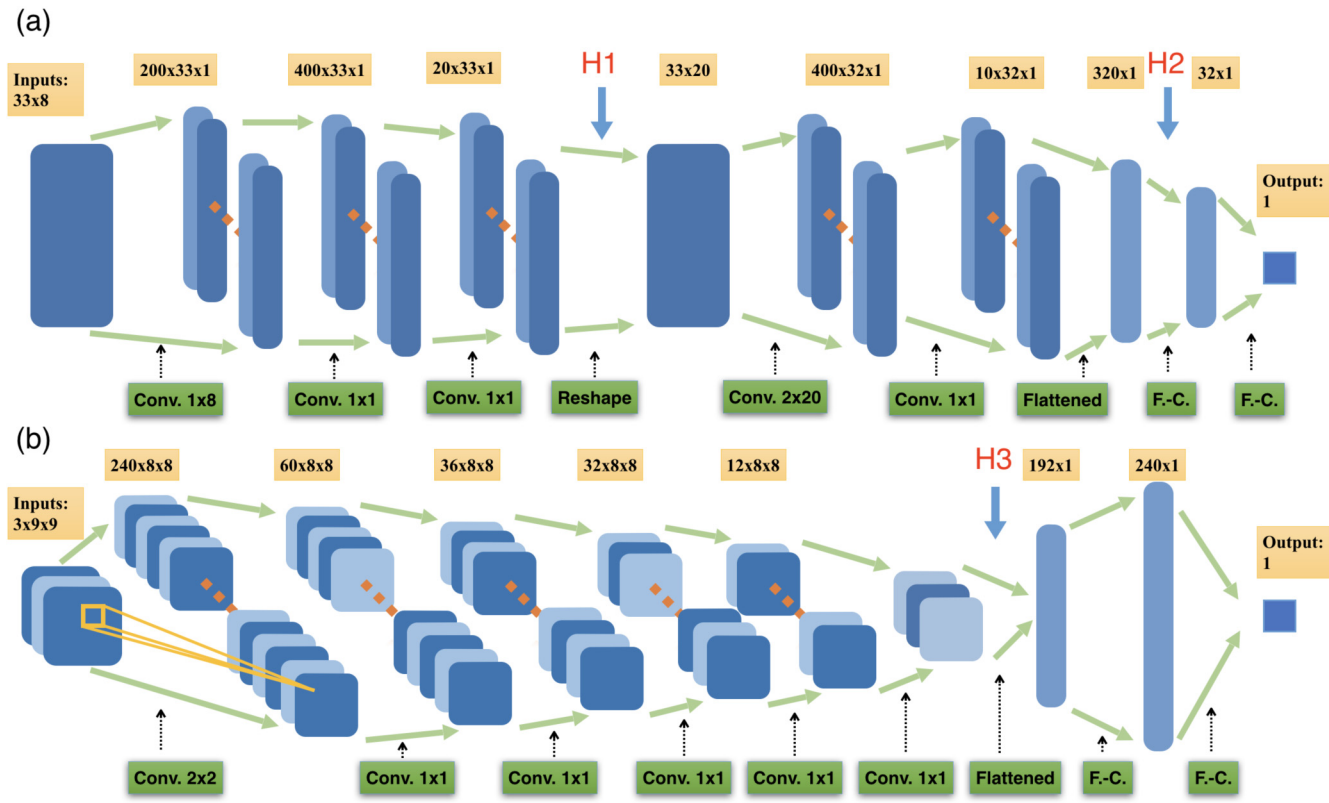

FIG. 1. The architecture of neural networks used for learning (a) the winding number of one-dimensional AIII class four-band Hamiltonians, and for (b) the Chern number of two-dimensional A class two-band Hamiltonians. In both figures, each linear transformation layer is followed by a subsequent nonlinear ReLU function. The Conv. and F.-C. in the figure denote the convolutional layer and the fully connected layer, respectively. The label $a \times b(\times c)$ specifies the dimension of the fully connected (convolutional) layer. H1, H2, and H3 label layers that we will analyze later.

neural network works. We follow this routine in Sec. III and show the result for two-dimensional two-band models in A class.

II. WINDING NUMBER WITH MULTIPLE BANDS

A. Model

Here we study the problem directly in momentum space as in [15]. Using input data in real space to train a neural network for distinguishing topological phases is studied in [32,33].

Consider a $2d$-band model in one dimension and introduce $\hat{\Psi}_k^\dagger = (\hat{c}_{1,k}^\dagger, \hat{c}_{2,k}^\dagger, \ldots, \hat{c}_{2d,k}^\dagger)$, where $\hat{c}_{ik}^\dagger$ is the creation operator for a fermion on i orbital with momentum k. A general one-dimensional four-band Hamiltonian in AIII class can be written as $\hat{H} = \sum_k \hat{\Psi}_k^\dagger H(k) \hat{\Psi}_k$, where

$$H(k) = \begin{pmatrix} 0 & D(k) \\ D^\dagger(k) & 0 \end{pmatrix}. \tag{1}$$

Without loss of generality, here $D(k) \in U(d)$ is a d-dimensional unitary matrix [60] and $k \in [-\pi, \pi]$. The topological classification of band Hamiltonians in AIII class is the group $\mathbb{Z}$ [61]. When the model is half-filled, the topological invariant is computed by

$$w = \frac{1}{2\pi} \int_{-\pi}^{\pi} dk \mathrm{Tr}[D^{-1}(k) i \partial_k D(k)]. \tag{2}$$

Since $D(k)$ is unitary, it can be diagonalized as $D(k) = V^\dagger(k) M(k) V(k)$, where $M(k)$ is a d-dimensional diagonal matrix with diagonal elements $\{e^{-i\theta_1(k)}, e^{-i\theta_2(k)}, \ldots, e^{-i\theta_d(k)}\}$. Formally, $D(k)$ can also be uniquely decomposed as $D(k) = e^{-i\alpha(k)} \tilde{D}(k)$, where $\tilde{D}(k) \in \mathrm{SU}(d)$ is a d-dimensional unitary matrix with determinant 1 and $\alpha(k) = \sum_i \theta_i(k)/d \in [-\pi/d, \pi/d)$ is the winding angle at momentum k.

To be concrete, we restrict our discussion to $d = 2$, which corresponds to four-band models. The winding number formula of Eq. (2) can then be reduced to

$$w = \frac{1}{\pi} \int_{-\pi}^{\pi} dk \partial_k \alpha(k), \tag{3}$$

where $\alpha(k) = [\theta_1(k) + \theta_2(k)]/2 \mod \pi$ so that $\alpha(k) \in [-\pi/2, \pi/2)$. The discretized version of the winding number formula is

$$\begin{aligned} w &= \frac{1}{\pi} \sum_{l=1}^{L} \Delta\alpha(k_l) \\ &= \frac{1}{\pi} \sum_{l=1}^{L} [\alpha(k_{l+1}) - \alpha(k_l)] \mod \pi, \end{aligned} \tag{4}$$

where k_i, $i = 1, \ldots, L$ are distributed uniformly in the Brillouin zone and $\Delta\alpha(k) \in [-\pi/2, \pi/2)$.

TABLE I. The accuracy of the neural network prediction on test Hamiltonians with winding numbers $w = 0, \pm1, \pm2, \pm3, \pm4$, respectively.

w	0	±1	±2	±3	±4
Accuracy	97%	96%	96%	95%	93%

B. Neural network performance

Since the neural network can only take discrete input, we first discretize the entire Brillouin zone uniformly into L points $\{k_l \in [-\pi, \pi) | l = 1, \ldots, L+1\}$ by choosing $k_l = 2\pi(l-1)/L$. At each point, since the Hamiltonian is determined by the 2×2 matrix $D(k)$, we denote its four elements as $D_{11}, D_{12}, D_{21}, D_{22}$. The input data is therefore a $8 \times (L+1)$-dimensional matrix of the following form:

$$\begin{pmatrix} \mathrm{Re}[D_{11}(0)] & \mathrm{Re}[D_{11}(2\pi/L)] & \cdots & \mathrm{Re}[D_{11}(2\pi)] \\ \mathrm{Im}[D_{11}(0)] & \mathrm{Im}[D_{11}(2\pi/L)] & \cdots & \mathrm{Im}[D_{11}(2\pi)] \\ \mathrm{Re}[D_{12}(0)] & \mathrm{Re}[D_{12}(2\pi/L)] & \cdots & \mathrm{Re}[D_{12}(2\pi)] \\ \mathrm{Im}[D_{12}(0)] & \mathrm{Im}[D_{12}(2\pi/L)] & \cdots & \mathrm{Im}[D_{12}(2\pi)] \\ \mathrm{Re}[D_{21}(0)] & \mathrm{Re}[D_{21}(2\pi/L)] & \cdots & \mathrm{Re}[D_{21}(2\pi)] \\ \mathrm{Im}[D_{21}(0)] & \mathrm{Im}[D_{21}(2\pi/L)] & \cdots & \mathrm{Im}[D_{21}(2\pi)] \\ \mathrm{Re}[D_{22}(0)] & \mathrm{Re}[D_{22}(2\pi/L)] & \cdots & \mathrm{Re}[D_{22}(2\pi)] \\ \mathrm{Im}[D_{22}(0)] & \mathrm{Im}[D_{22}(2\pi/L)] & \cdots & \mathrm{Im}[D_{22}(2\pi)] \end{pmatrix}. \tag{5}$$

In the following we set $L = 32$.

The structure of the deep neural network is shown in Fig. 1(a). It first contains several convolutional layers with kernel sizes marked in the figure, which are followed by two fully connected layers leading to the final output. In each layer, a linear mapping is followed by a nonlinear ReLU function. We feed the neural network with a set of 3×10^4 discretized training Hamiltonians with winding number $\{0, \pm1, \pm2, \pm3\}$ for supervised training.

To compute accuracy, the final winding number is taken as the closest integer of the numerical value predicted by the network. It is considered as a correct prediction if the rounded integer matches the value computed by Eq. (4). The accuracy of this neural network is shown in Table I. After training, the neural network achieves a prediction accuracy of 96% on Hamiltonians with winding numbers $\{0, \pm1, \pm2, \pm3\}$ in a separate test data set, and an accuracy of more than 90% on Hamiltonians with winding number of $\{\pm4\}$ that are beyond the training set. The numerical values of the winding number predicted for each Hamiltonian in the test set are shown in Fig. 2.

C. Neural network analysis

To see why the neural network excels predicting the topological winding number, it is illuminating to check whether the complicated function fitted by the neural network is consistent with the mathematical formula Eq. (4) introduced above. We open up the neural network at H1 and H2 marked in Fig. 1 by feeding test Hamiltonians into the neural network and plotting intermediate outputs at H1 and H2 separately. Notice that the output of H1 is of dimension $(L+1) \times 20$, while the dimension of H2 is $L \times 10$. Each row of H1 can be interpreted as a vector $r \in \mathbb{R}^{L+1}$, and each row of H2 can be interpreted as vector $v \in \mathbb{R}^L$. They respectively have the same dimension as the discretized $\alpha(k)$ and $\Delta\alpha(k)$ defined in Sec. II A. On the other hand, the exact value of $\alpha(k)$ and $\Delta\alpha(k)$ of the corresponding Hamiltonian can also be obtained directly according to the definition in Sec. II A. In Fig. 3(a) we plot $\{(\alpha(k_i), r_i) | i = 1, \ldots, L+1\}$, where r_i is the ith component of a selected row of H1, for various k_i and input Hamiltonians. The plot for H2 in Fig. 3(b) is similar where $\{(\Delta\alpha(k_i), v_i) | i = 1, \ldots, L\}$ are plotted.

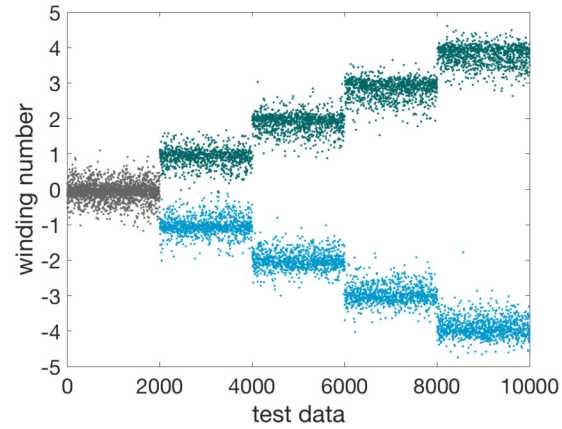

FIG. 2. The test data set contains 10^4 Hamiltonians which are labeled from 1 to 10 000. Hamiltonians labeled from $2000i$ to $2000(i+1)$ have winding number $\pm i$, with different colors distinguishing $+i$ from $-i$. The vertical axis shows the winding number (direct output) predicted by the neural network.

As can be seen in Fig. 3(a), the intermediate output at H1 is approximately piecewise linear with α, implying that this row of neuron successfully extracts the winding angle α within some range. Other rows of neurons extract winding angles at different ranges. In Fig. 3(b) the intermediate output at H2 is approximately linear with $\Delta\alpha$ within some range, and each row of neurons functions as a $\Delta\alpha$ extractor for different ranges of

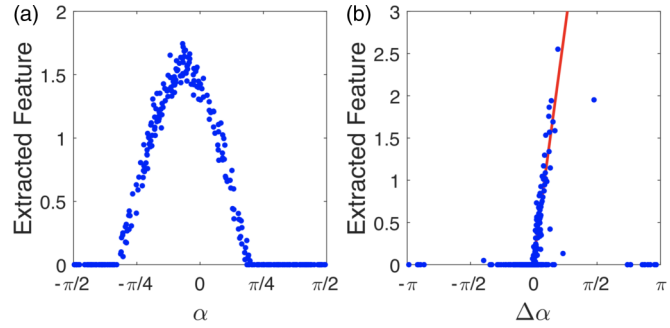

FIG. 3. Extracted features of the hidden layers. (a) The intermediate output r_i which is a typical row of the layer marked by H1 in Fig. 1 vs the corresponding exact value of $\alpha(k_i)$ for the input Hamiltonian. Other rows exhibit similar behavior which is not shown. (b) The intermediate output v_i which is a typical row of the layer marked by H2 vs the corresponding exact value of $\Delta\alpha(k_i)$. Other rows exhibit similar behavior which is not shown. In both figures, the results of five different test Hamiltonians are plotted so that there are $5(L+1)$ and $5L$ points in total, respectively.

$\Delta\alpha$. Although their ranges may overlap with each other or have different slopes in their linear relations with the exact $\Delta\alpha$, a linear combination of these extractors with correct coefficients in the following fully connected layer can easily lead to a function proportional to $\Delta\alpha$ at all ranges. In this way, the winding number is calculated essentially the same way as that using the mathematical formula Eq. (4).

As emphasized in Sec. II A, it is important to notice the input Hamiltonian can be written as the product of a phase factor and a SU(d) matrix. The SU(d) matrix does not play any role in determining the winding number and only the phase factor matters. It is quite impressive that the neural network successfully distills the phase factor from the irrelevant SU(d) part.

III. CHERN NUMBER IN TWO DIMENSIONS

A. Model

Consider a two-band model in two dimensions and introduce $\hat{\Psi}_k^\dagger = (\hat{c}_{1,\mathbf{k}}^\dagger, \hat{c}_{2,\mathbf{k}}^\dagger)$, where $\hat{c}_{i,\mathbf{k}}^\dagger$ is the creation operator for a fermion on i orbital with momentum $\mathbf{k} = (k_x, k_y)$. A general two-dimensional two-band Hamiltonian in A class can be written as $\hat{H} = \sum_{\mathbf{k}} \hat{\Psi}_{\mathbf{k}}^\dagger H(\mathbf{k}) \hat{\Psi}_{\mathbf{k}}$, where

$$H(\mathbf{k}) = \mathbf{h}(\mathbf{k}) \cdot \boldsymbol{\sigma} = h_x(\mathbf{k})\sigma_x + h_y(\mathbf{k})\sigma_y + h_z(\mathbf{k})\sigma_z. \quad (6)$$

Here $\boldsymbol{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$ is a vector of Pauli matrices. Without loss of generality, we can take $|\mathbf{h}(\mathbf{k})| = 1$ as the normalization [62]. In two dimensions, the Chern number can be computed as

$$C = \frac{1}{2\pi} \int_{T^2} d^2\mathbf{k} F_{xy}(\mathbf{k}), \quad (7)$$

where T^2 is the torus of the Brillouin zone and

$$A_\mu(\mathbf{k}) = i\langle u(\mathbf{k})|\partial_\mu u(\mathbf{k})\rangle, \quad F_{\mu\nu}(\mathbf{k}) = \partial_\mu A_\nu - \partial_\nu A_\mu. \quad (8)$$

Here we assume the model is half-filled so that $|u(\mathbf{k})\rangle$ is the energy eigenstate with the lower energy $H(\mathbf{k})|u(\mathbf{k})\rangle = -|u(\mathbf{k})\rangle$. The integrand in Eq. (7) is then the Berry curvature of the lower band. For discretized lattices, the Berry curvature and the Chern number can be defined through the Wilson-loop approach, as is elaborated in the Appendix.

B. Neural network performance

The input data are Hamiltonians in the discretized Brillouin zone, i.e., $3 \times (L+1) \times (L+1)$ tensors $(\mathcal{H}_x, \mathcal{H}_y, \mathcal{H}_z)$ with

$$\mathcal{H}_\mu = \begin{pmatrix} h_\mu(0,0) & h_\mu\left(0, \frac{2\pi}{L}\right) & \cdots & h_\mu(0, 2\pi) \\ h_\mu\left(\frac{2\pi}{L}, 0\right) & h_\mu\left(\frac{2\pi}{L}, \frac{2\pi}{L}\right) & \cdots & h_\mu\left(\frac{2\pi}{L}, 2\pi\right) \\ \vdots & \vdots & \ddots & \vdots \\ h_\mu(2\pi, 0) & h_\mu\left(2\pi, \frac{2\pi}{L}\right) & \cdots & h_\mu(2\pi, 2\pi) \end{pmatrix}. \quad (9)$$

The corresponding Chern numbers are calculated using the method presented in the Appendix. In the following we take $L = 8$.

The structure of the neural network is shown in Fig. 1(b) which is similar to that used for the winding number. We feed the neural network with 10^4 randomly generated Hamiltonians with Chern numbers limited to $\{0, \pm1, \pm2\}$. The accuracy here is computed similarly to before by rounding the final output of the network to the closet integer. After training, the neural network can achieve an accuracy of 92% on Hamiltonians with Chern numbers $C \in \{0, \pm1, \pm2\}$, an accuracy of 84% on Hamiltonians with Chern numbers ±3, and an accuracy of 85% on Hamiltonians with Chern numbers ±4. These results are shown in Fig. 4 and are summarized in Table II.

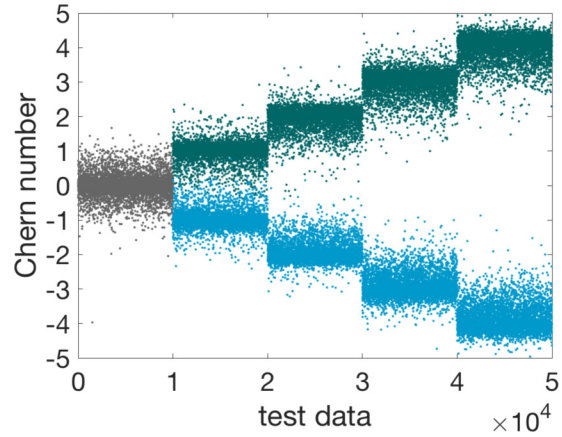

FIG. 4. The test data set contains 5×10^4 Hamiltonians which are labeled from 1 to 5×10^4. The data labeled from $i+1$ to $(i+1)10^4$ has Chern number $\pm i$, with different colors distinguishing $+i$ from $-i$. The vertical axis shows the Chern number (direct output) predicted by the neural network.

C. Neural network analysis

We feed the neural network with a Hamiltonian in the test data set and plot the intermediate output of the last convolutional layer [marked by H3 in Fig. 1(b)] in Figs. 5(b)–5(d). The output consists of three layers of $L \times L$ matrices, which are respectively shown in Figs. 5(b), 5(c) and 5(d). They should be compared with the exact Berry curvature for the corresponding Hamiltonian shown in Fig. 5(a). Since the intermediate output is positive due to nature of the ReLU function while the Berry curvature are generally positive somewhere and negative elsewhere, the intermediate output reproduces the positive part of the Berry curvature in one layer [Fig. 5(b)] and the negative part in another layer [Fig. 5(c)]. The remaining third layer is almost irresponsive [Fig. 5(d)]. This result shows the neural network compute the topological invariant by first computing local Berry curvatures in the momentum space and then adding them together, which is essentially the same as Eq. (7).

IV. SUMMARY

In summary, we have trained deep neural networks to predict the winding number of one-dimensional four-band models in

TABLE II. The accuracy of the neural network prediction on test Hamiltonians with Chern numbers $C = 0, \pm1, \pm2, \pm3, \pm4$, respectively.

C	0	±1	±2	±3	±4
Accuracy	93%	92%	90%	86%	85%

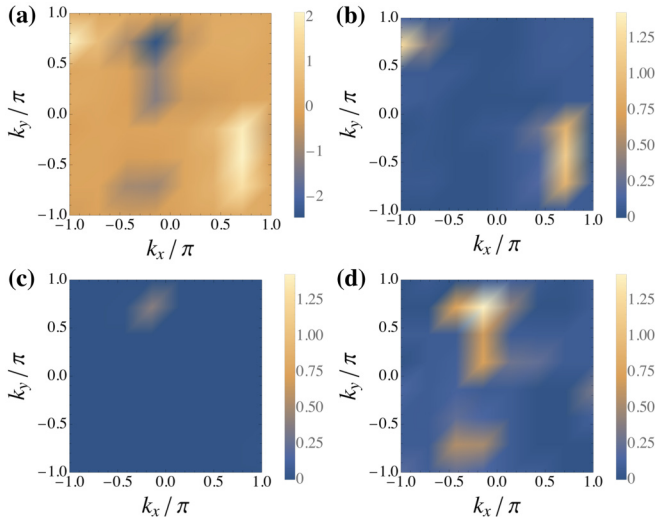

FIG. 5. (a) The calculated Berry curvature for a test Hamiltonian in the first Brillouin zone. (b)–(d) The corresponding intermediate outputs at the layer marked by H3 in Fig. 1(b) before the fully connected layers. Notice that the output is a three tensor, (b), (c), and (d) corresponds to three different components of the three tensor.

AIII class and the Chern number of two-dimensional two-band models in A class. In addition to the high prediction accuracies after the training, it is understood that deep neural networks essentially fit the mathematical formula for both topological invariants. In the first case, the network successfully distills the $U(d)$ phase factors of Hamiltonians between two successive momenta and discards the SU(d) degrees of freedom that is redundant in determining the topology. In the second case, the network successfully extracts the Berry curvature in momentum space. Our work provides an explicit example that even a complicated deep neural network can be understood. Our work can be further combined with *ab initio* calculations, and paves the way to the direct prediction of topological properties of real materials using machine learning.

N.S. and J.Y. contributed equally to this work.

APPENDIX: CHERN NUMBER IN DISCRETE SPACES

The continuous version of Chern number and Berry curvature is defined in Eq. (8) in the main text. To introduce the discrete version of Chern number, it is convenient to first define the Berry curvature in discrete spaces [63]. The Chern number is then the summation of Berry curvatures in the space.

The definition of the Berry curvature and the Chern number in discrete spaces, and the procedure for computing them are outlined as follows.

1. Discretize a two-dimensional parameter space as $L \times L$ sites. With periodic boundary condition by identifying sites at the boundary, there are $L \times L$ plaquettes in total. In our setting, sites are labeled as $\mathbf{k} = (k_x, k_y)$. For uniform discretization, the area of each plaquette is $s(\mathbf{k}) = \Delta k_x \Delta k_y$, where Δk_x and Δk_y is the distance of neighboring sites along k_x and k_y respectively.

2. At each site $\mathbf{k} = (k_x, k_y)$ in the discretized two-dimensional parameter space, diagonalize the Hamiltonian $H(\mathbf{k}) = V(\mathbf{k})D(\mathbf{k})V^\dagger(\mathbf{k})$ to obtain the eigenstates of the nth band $|u^{(n)}(\mathbf{k})\rangle$. $D(\mathbf{k})$ is a diagonal matrix with its diagonal elements the eigenenergy of each band.

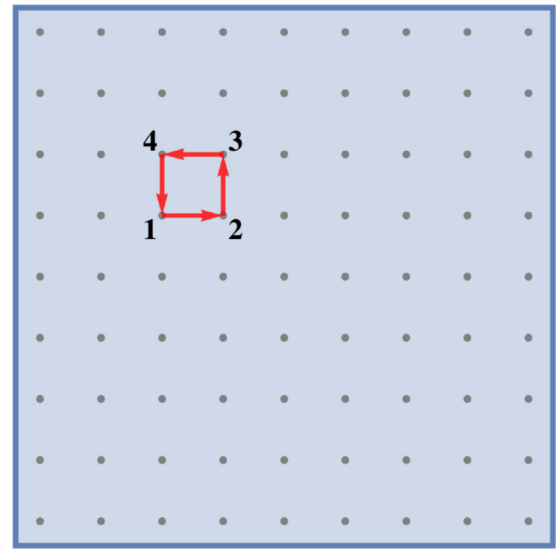

FIG. 6. Schematic of discretized two-dimensional parameter space and the Wilson loop. Numbers label the ordering of the loop.

3. All four vertices in each plaquette construct an ordered loop, called the Wilson loop (Fig. 6).

(a) Compute the ordered inner product of the eigenstates along the ordered loop in each plaquette. Specifically, define

$$U_{12} = V^\dagger(\mathbf{k}_2)V(\mathbf{k}_1), \quad U_{23} = V^\dagger(\mathbf{k}_3)V(\mathbf{k}_2),$$
$$U_{34} = V^\dagger(\mathbf{k}_4)V(\mathbf{k}_3), \quad U_{41} = V^\dagger(\mathbf{k}_1)V(\mathbf{k}_4),$$

(b) Define $\mathcal{U}_{ij} = \mathrm{diag}(U_{ij})$, where $\mathrm{diag}(\cdots)$ means to extract the diagonal elements and construct a diagonal matrix. That is, $(\mathcal{U}_{ij})_{mn} = \delta_{mn}(U_{ij})_{nn}$.

(c) Define $T_{\mathrm{loop}}(\mathbf{k}_1) = \mathcal{U}_{41}\mathcal{U}_{34}\mathcal{U}_{23}\mathcal{U}_{12}$. $-i \log T(\mathbf{k}_1)$ is the (non-Abelian) Berry curvature at the plaquette labeled $\mathbf{k}_1$. Define $\theta_n(\mathbf{k}) = -i \log[T_{\mathrm{loop}}(k_i, k_j)]_{nn}$ and the Berry curvature of the nth band $\mathcal{F}^{(n)}_{xy}$,

$$\mathcal{F}^{(n)}_{xy}(\mathbf{k}) = \theta_n(\mathbf{k})/s(\mathbf{k}). \tag{A1}$$

4. The Chern number is the summation of the Berry curvature of all plaquettes. Define c_n as the Chern number of the nth band:

$$c_n = \frac{1}{2\pi}\sum_{i=1}^{L\times L}\theta(\mathbf{k}_i)$$
$$= \frac{1}{2\pi}\sum_{i=1}^{L}\sum_{j=1}^{L} -i \log T^{(nn)}_{\mathrm{loop}}(k_i, k_j). \tag{A2}$$

It can be verified that the Chern number defined above is quantized and gauge invariant. For a model defined in the continuous space but whose Chern number is computed only on discretized points in the continuous space, Eq. (A2) gives the same result as Eq. (7) if the discretization is dense enough. Hence Eqs. (A1) and (A2) can be seen as the generalization of the Berry curvature and the Chern number to discrete spaces.

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

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, in *Advances in Neural Information Processing Systems 25*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., 2012), pp. 1097–1105.

[3] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, Learning hierarchical features for scene labeling, IEEE Trans. Pattern Anal. Machine Intell. **35**, 1925 (2013).

[4] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation, in *Advances in Neural Information Processing Systems 27*, edited by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Curran Associates, Inc., 2014), pp. 1799–1807.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, Going Deeper with Convolutions, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 1–9.

[6] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Cernocky, Strategies for training large scale neural network language models, *2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)* (IEEE, Washington, DC, 2011).

[7] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, IEEE Signal Process. Mag. **29**, 82 (2012).

[8] T. N. Sainath, A.-R. Mohamed, B. Kingsbury, and B. Ramabhadran, Deep convolutional neural networks for LVCSR, *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, Washington, DC, 2013).

[9] P. Baldi, P. Sadowski, and D. Whiteson, Searching for exotic particles in high-energy physics with deep learning, Nat. Commun. **5**, 4308 (2014).

[10] S. Whiteson and D. Whiteson, Machine learning for event selection in high energy physics, Eng. Appl. Artificial Intell. **22**, 1203 (2009).

[11] S. Ravanbakhsh, F. Lanusse, R. Mandelbaum, J. G. Schneider, and B. Poczos, *Enabling Dark Energy Science with Deep Generative Models of Galaxy Images* (AAAI, Palo Alto, CA, 2017).

[12] N. M. Ball and R. J. Brunner, Data mining and machine learning in astronomy, Int. J. Mod. Phys. D **19**, 1049 (2010).

[13] O. Y. Al-Jarrah, P. D. Yoo, S. Muhaidat, G. K. Karagiannidis, and K. Taha, Efficient machine learning for big data: A review, Big Data Res. **2**, 87 (2015).

[14] S. Dieleman, K. W. Willett, and J. Dambre, Rotation-invariant convolutional neural networks for galaxy morphology prediction, Mon. Not. R. Astron. Soc. **450**, 1441 (2015).

[15] P. Zhang, H. Shen, and H. Zhai, Machine Learning Topological Invariants with Neural Networks, Phys. Rev. Lett. **120**, 066401 (2018).

[16] A. Morningstar and R. G. Melko, Deep learning the Ising model near criticality, arXiv:1708.04622.

[17] T. Mano and T. Ohtsuki, Phase diagrams of three-dimensional Anderson and quantum percolation models using deep three-dimensional convolutional neural network, J. Phys. Soc. Jpn. **86**, 113704 (2017).

[18] W.-J. Rao, Z. Li, Q. Zhu, M. Luo, and X. Wan, Identifying product order with restricted Boltzmann machines, Phys. Rev. B **97**, 094207 (2018).

[19] Y. Nomura, A. S. Darmawan, Y. Yamaji, and M. Imada, Restricted Boltzmann machine learning for solving strongly correlated quantum systems, Phys. Rev. B **96**, 205152 (2017).

[20] N. Yoshioka, Y. Akagi, and H. Katsura, Machine learning disordered topological phases by statistical recovery of symmetry, Bull. Am. Phys. Soc. (2018), https://meetings.aps.org/Meeting/MAR18/Session/P34.5.

[21] J. Venderley, V. Khemani, and E.-A. Kim, Machine Learning Out-of-Equilibrium Phases of Matter, Phys. Rev. Lett. **120**, 257204 (2018).

[22] P. Suchsland and S. Wessel, Parameter diagnostics of phases and phase transition learning by neural networks, Phys. Rev. B **97**, 174435 (2018).

[23] S. Arai, M. Ohzeki, and K. Tanaka, Deep neural network detects quantum phase transition, J. Phys. Soc. Jpn. **87**, 033001 (2018).

[24] C.-D. Li, D.-R. Tan, and F.-J. Jiang, Applications of neural networks to the studies of phase transitions of two-dimensional Potts models, Annals Phys. **391**, 312 (2018).

[25] I. A. Iakovlev, O. M. Sotnikov, and V. V. Mazurenko, Supervised learning magnetic skyrmion phases, arXiv:1803.06682.

[26] P. Broecker, F. F. Assaad, and S. Trebst, Quantum phase recognition via unsupervised machine learning, arXiv:1707.00663.

[27] J. Carrasquilla and R. G. Melko, Machine learning phases of matter, Nat. Phys. **13**, 431 (2017).

[28] P. Broecker, J. Carrasquilla, R. G. Melko, and S. Trebst, Machine learning quantum phases of matter beyond the fermion sign problem, Sci. Rep. **7**, 8823 (2017).

[29] K. Ch'ng, J. Carrasquilla, R. Melko, and E. Khatami, Machine Learning Phases of Strongly Correlated Fermions, Phys. Rev. X **7**, 031038 (2017).

[30] Y. Zhang, and E.-A. Kim, Quantum Loop Topography for Machine Learning, Phys. Rev. Lett. **118**, 216401 (2017).

[31] Y. Zhang, R. G. Melko, and E.-A. Kim, Machine learning $\mathbb{Z}$ 2 quantum spin liquids with quasiparticle statistics, Phys. Rev. B **96**, 245119 (2017).

[32] T. Ohtsuki and T. Ohtsuki, Deep learning the quantum phase transitions in random two-dimensional electron systems, J. Phys. Soc. Jpn. **85**, 123706 (2016).

[33] T. Ohtsuki and T. Ohtsuki, Deep learning the quantum phase transitions in random electron systems: Applications to three dimensions, J. Phys. Soc. Jpn. **86**, 044708 (2017).

[34] F. Schindler, N. Regnault, and T. Neupert, Probing many-body localization with neural networks, Phys. Rev. B **95**, 245134 (2017).

[35] P. Ponte and R. G. Melko, Kernel methods for interpretable machine learning of order parameters, Phys. Rev. B **96**, 205146 (2017).

[36] L. Wang, Discovering phase transitions with unsupervised learning, Phys. Rev. B **94**, 195105 (2016).

[37] A. Tanaka and A. Tomiya, Detection of phase transition via convolutional neural networks, J. Phys. Soc. Jpn. **86**, 063001 (2017).

[38] E. V. Nieuwenburg, Y.-H. Liu, and S. D. Huber, Learning phase transitions by confusion, Nat. Phys. **13**, 435 (2017).

[39] Y.-H. Liu and E. P. L. van Nieuwenburg, Discriminative Cooperative Networks for Detecting Phase Transitions, Phys. Rev. Lett. **120**, 176401 (2018).

[40] R. Li, J. Wang, Y. Shu, and Z. Xu, The discrepancy between Einstein mass and dynamical mass for SIS and power-law mass models, Astrophys. J. **855**, 64 (2018).

[41] S. J. Wetzel, Unsupervised learning of phase transitions: From principal component analysis to variational autoencoders, Phys. Rev. E **96**, 022140 (2017).

[42] W. Hu, R. R. P. Singh, and R. T. Scalettar, Discovering phases, phase transitions, and crossovers through unsupervised machine learning: A critical examination, Phys. Rev. E **95**, 062122 (2017).

[43] N. C. Costa, W. Hu, Z. J. Bai, R. T. Scalettar, and R. R. P. Singh, Principal component analysis for fermionic critical points, Phys. Rev. B **96**, 195138 (2017).

[44] C. Wang and H. Zhai, Machine learning of frustrated classical spin models. I. Principal component analysis, Phys. Rev. B **96**, 144432 (2017).

[45] K. Ch'ng, N. Vazquez, and E. Khatami, Unsupervised machine learning account of magnetic transitions in the Hubbard model, Phys. Rev. E **97**, 013306 (2018).

[46] L. Huang and L. Wang, Accelerated Monte Carlo simulations with restricted Boltzmann machines, Phys. Rev. B **95**, 035105 (2017).

[47] L. Huang, Y.-F. Yang, and L. Wang, Recommender engine for continuous-time quantum Monte Carlo methods, Phys. Rev. E **95**, 031301(R) (2017).

[48] J. Liu, Y. Qi, Z. Y. Meng, and L. Fu, Self-learning Monte Carlo method, Phys. Rev. B **95**, 041101(R) (2017).

[49] J. Liu, H. Shen, Y. Qi, Z. Y. Meng, and L. Fu, Self-learning Monte Carlo method and cumulative update in fermion systems, Phys. Rev. B **95**, 241104(R) (2017).

[50] Y. Nagai, H. Shen, Y. Qi, J. Liu, and L. Fu, Self-learning Monte Carlo method: Continuous-time algorithm, Phys. Rev. B **96**, 161102(R) (2017).

[51] X. Y. Xu, Y. Qi, J. Liu, L. Fu, and Z. Y. Meng, Self-learning quantum Monte Carlo method in interacting fermion systems, Phys. Rev. B **96**, 041119(R) (2017).

[52] H. Shen, J. Liu, and L. Fu, Self-learning Monte Carlo with deep neural networks, Phys. Rev. B **97**, 205140 (2018).

[53] L.-F. Arsenault, A. Lopez-Bezanilla, O. A. von Lilienfeld, and A. J. Millis, Machine learning for many-body physics: The case of the Anderson impurity model, Phys. Rev. B **90**, 155136 (2014).

[54] L.-F. Arsenault, O. A. von Lilienfeld, and A. J. Millis, Machine learning for many-body physics: Efficient solution of dynamical mean-field theory, arXiv:1506.08858.

[55] D.-L. Deng, X. Li, and S. Das Sarma, Quantum Entanglement in Neural Network States, Phys. Rev. X **7**, 021021 (2017).

[56] S. Lu, S. Huang, K. Li, J. Li, J. Chen, D. Lu, Z. Ji, Y. Shen, D. Zhou, and B. Zeng, A separability-entanglement classifier via machine learning, Phys. Rev. A **98**, 012315 2018.

[57] Y. Levine, D. Yakira, N. Cohen, and A. Shashua, Deep learning and quantum entanglement: Fundamental connections with implications to network design, arXiv:1704.01552.

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

[59] This is to be compared with a fully connected neural network without this symmetry constraint, which may contain more parameters but is more difficult to be trained to the optimal solution that reproduces the translational symmetry [15].

[60] Here we assume $D(k)$ is a square matrix for simplicity [61]. Although in general $D(k)$ might not be unitary, we can always make the flat-band approximation for a gapped system such that the topological property remains invariant. The $D(k)$ matrix is then unitary and takes the form of Eq. (1).

[61] C.-K. Chiu, J. C. Y. Teo, A. P. Schnyder, and S. Ryu, Classification of topological quantum matter with symmetries, Rev. Mod. Phys. **88**, 035005 (2016).

[62] This is similar to that $D(k)$ is taken as the unitary matrix in the previous case of the winding number, because we can always take flat-band approximation for an insulator without changing its band topology.

[63] T. Fukui, Y. Hatsugai, and H. Suzuki, Chern numbers in discretized Brillouin zone: Efficient method of computing (spin) Hall conductances, J. Phys. Soc. Jpn. **74**, 1674 (2005).