CHIN. PHYS. LETT. Vol. 38, No. 7 (2021) 070302

Supervised Machine Learning Topological States of One-Dimensional Non-Hermitian Systems

Zhuo Cheng(成卓)[1] and Zhenhua Yu(俞振华)[1,2*]

[1]*Guangdong Provincial Key Laboratory of Quantum Metrology and Sensing, School of Physics and Astronomy, Sun Yat-Sen University, Zhuhai 519082, China*

[2]*State Key Laboratory of Optoelectronic Meterials and Technologies, Sun Yat-Sen University, Guangzhou 510275, China*

(Received 19 March 2021; accepted 6 May 2021; published online 3 July 2021)

We apply supervised machine learning to study the topological states of one-dimensional non-Hermitian systems. Unlike Hermitian systems, the winding number of such non-Hermitian systems can take half integers. We focus on a non-Hermitian model, an extension of the Su–Schrieffer–Heeger model. The non-Hermitian model maintains the chiral symmetry. We find that trained neuron networks can reproduce the topological phase diagram of our model with high accuracy. This successful reproduction goes beyond the parameter space used in the training process. Through analyzing the intermediate output of the networks, we attribute the success of the networks to their mastery of computation of the winding number. Our work may motivate further investigation on the machine learning of non-Hermitian systems.

DOI: 10.1088/0256-307X/38/7/070302

In recent years, notions drawn from non-Hermitian systems have attracted wide attention.[1–6] Unlike Hermitian systems,[7,8] the Hamiltonians of non-Hermitian systems generally have complex eigenvalues. Such non-Hermitian systems have traditionally been used to describe dissipative processes. As topological invariants play a key role in studying topological states of physical systems, due to the existence of possible exceptional points and complex spectrum, conventional topological invariants such as the winding number and the Chern number have been generalized to non-hermitian systems.[3,9,10]

On the other hand, machine learning has recently shown wide applications in research of physics, such as identification of phase transitions[11–18] and stimulation of quantum states.[19] Even though the topology of a phase is intrinsically a global property, trained artificial neuron networks have achieved classifying topological phases in Hermitian systems.[11,20,21]

In this Letter, we employ the multilayer perceptron (MLP) and convolutional neural networks (CNN) to study topological states of non-Hermitian Hamiltonians. Among various topological models, the Su–Schrieffer–Heeger (SSH) model is a paradigm of symmetry protected topological models exhibiting band topology in condensed matter physics;[22] the chiral symmetry fulfilled by the SSH model guarantees the winding number of the model to be an integer.[23] In our study, we focus on a non-Hermitian model which is extended from the SSH one [see Eq. (1)]. We carry out the supervised machine learning of the networks on the non-Hermitian model. After training, the networks are able to produce the correct phase diagram of the model in the parameter space larger than the one used in training. Our analysis of the intermediate output of the networks suggests that the networks have acquired the topological invariant, i.e., the winding number, through the training of the non-Hermitian model. Our work may motivate further studies on the machine learning of non-Hermitian systems.

We consider a non-Hermitian extension of the SSH model[24–26] whose Hamiltonian reads

$$H = \sum_n \big[(t-\delta)\hat{a}_n^\dagger \hat{b}_n + (t+\delta)\hat{b}_n^\dagger \hat{a}_n + t'\hat{a}_{n+1}^\dagger \hat{b}_n + t'\hat{b}_n^\dagger \hat{a}_{n+1} + \Delta \hat{a}_{n+2}^\dagger \hat{b}_n + \Delta \hat{b}_n^\dagger \hat{a}_{n+2}\big]. \tag{1}$$

The parameters t, t', δ, Δ are all assumed to be real. The introduction of nonzero δ makes the model non-Hermitian, which gives rise to biased hopping amplitudes in the left and right directions. The presence of Δ expands the parameter space of our model, and results in five topological phases (see below).

Applying the periodic boundary condition, we transform the Hamiltonian into the form

$$H = \sum_k \hat{\psi}_k^\dagger h(k) \hat{\psi}_k, \tag{2}$$

Supported by the Key Area Research and Development Program of Guangdong Province (Grant No. 2019B030330001), the National Natural Science Foundation of China (Grant Nos. 11474179, 11722438, 91736103, and 12074440), and Guangdong Project (Grant No. 2017GC010613).

*Corresponding author. Email: yuzhh5@mail.sysu.edu.cn

© 2021 Chinese Physical Society and IOP Publishing Ltd

where the Bloch Hamiltonian is

$$h(k) = \begin{bmatrix} 0 & t-\delta+t'e^{-\mathrm{ik}}+\Delta e^{-2ik} \\ t+\delta+t'e^{ik}+\Delta e^{2ik} & 0 \end{bmatrix}, \tag{3}$$

and $\hat{\psi}_k = (\hat{a}_k, \hat{b}_k)^T$, and $\hat{a}_k, \hat{b}_k$ are the Fourier transforms of $\hat{a}_n, \hat{b}_n$, respectively. In terms of the Pauli matrices, we have $h(k) = B_x(k)\sigma_x + B_y(k)\sigma_y + B_z(k)\sigma_z$ with $B_x(k) = t + t'\cos(k) + \Delta\cos(2k)$, $B_y(k) = t'\sin(k) + \Delta\sin(2k) - i\delta$ and $B_z(k) = 0$. Note that our model maintains the chiral symmetry, i.e., $\sigma_z h(k)\sigma_z = -h(k)$, and the Zak phase of the system shall be quantized.[23]

The Zak phase can be explicitly calculated as

$$\gamma = \int_0^{2\pi} dk \frac{B_x(k)\partial_k B_y(k) - B_y(k)\partial_k B_x(k)}{B_x^2(k) + B_y^2(k)}. \tag{4}$$

Equivalently, we can define the complex angle $\phi(k)$ as $\tan\phi(k) = B_y(k)/B_x(k)$, and have

$$\gamma = \int_0^{2\pi} dk \partial_k \phi(k). \tag{5}$$

As ϕ_i is a continuous periodic function of k, γ calculated by Eq. (5) is real.[24] The winding number $\nu = \gamma/2\pi$ is usually used to label topological phases. Figure 1 shows the phase diagram of our model obtained by numerical calculation of Eq. (5). We take t' as the energy unit, i.e., $t' = 1$. In the parameter regime of $-4 < \delta < 4$ and $-4 < t < 4$ with $\Delta = 1$, we find five different topological phases represented in different colors corresponding to $\nu = 0, 1/2, 1, 3/2, 2$.

In the following, we are going to test the capability of neural networks in learning the classification of topological phases of our non-Hermitian model. Here we employ two types of networks: One is the multilayer perceptron (MLP) network,[27] and the other is the convolutional neural (CNN) network.[28] Both networks are composed of multi-layers. The schematic is given in Fig. 2. Our test procedure beginning with generating a large training dataset consisting of the non-Hermitian Hamiltonians $h(k)$ of varying parameters values for δ and t in the regime $-3 < \delta < 3$ and $-3 < t < 3$ with $t' = 1$ and $\Delta = 1$, and their corresponding values for the winding number ν. Next, we feed the training dataset to our two types of neural networks for the purpose of training. Finally, we use a test dataset generated independently in the same parameter regime with known classification to evaluate the performance of the networks.[29]

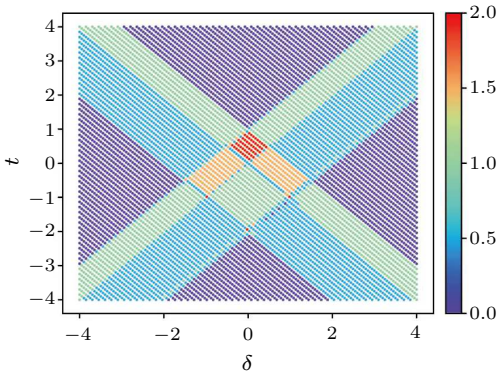

Fig. 1. The phase diagram of the non-Hermitian model, Eq. (1), with $t' = 1$ and $\Delta = 1$. The phase diagram is obtained by numerical calculation. Colors are used to distinguish topological phases of different values of the winding number.

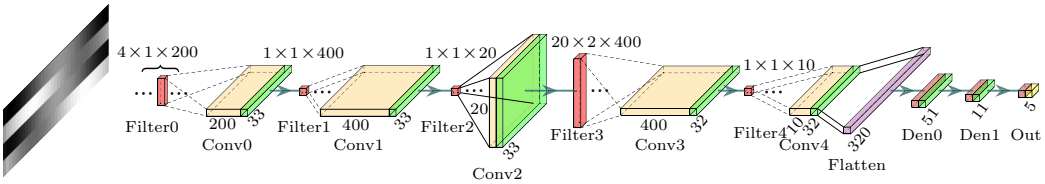

Fig. 2. Schematic of CNN. The network consists of convolution layers and dense layers, layers include 126833 trainable parameters; data flow in the direction of the arrows.

Our training dataset consists of 10^6 Hamiltonians and the test dataset 10^3 Hamiltonians. Each Hamiltonian is represented discretely in the momentum space by the real and imaginary part of $B_x(k)$ and $B_y(k)$, i.e., $B_{xr}(k), B_{yr}(k), B_{xi}(k), B_{yi}(k)$, in the matrix form

$$\begin{bmatrix} B_{xr}(0) & B_{xr}[2\pi i/(N\times k)] & \ldots & B_{xr}(2\pi) \\ B_{xi}(0) & B_{xi}[2\pi i/(N\times k)] & \ldots & B_{xi}(2\pi) \\ B_{yr}(0) & B_{yr}[2\pi i/(N\times k)] & \ldots & B_{yr}(2\pi) \\ B_{yi}(0) & B_{yi}[2\pi i/(N\times k)] & \ldots & B_{yi}(2\pi) \end{bmatrix} \tag{6}$$

with $i = 0, 1, \ldots, 32$ and $N = 32$. The output layer of both MLP and CNN has 5 neurons indexed from 0 to 4, corresponding to the five topological phases of our model of $\nu = 0, 1/2, 1, 3/2, 2$. The output value of each neuron is from 0 to 1, reflecting the possibility of the input Hamiltonians belonging to the corresponding topological phase; the neuron giving the highest value among all the five neurons is taken to be the topological phase that the input Hamiltonian belongs to.

Before the training process, we initialize the parameters of the networks $\boldsymbol{w}$ randomly according to the normal distribution with the Xavier algorithm.[30] The training is a process of optimizing the loss averaged function $L(\boldsymbol{w})$ given as

$$L(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} L_i(x_i, y_i, \boldsymbol{w}). \tag{7}$$

The explicit form of the loss function that we use is the categorical cross entropy.[31] According to our training dataset, x represents a Hamiltonian, and y represents its winding number. The training dataset is equally divided into N parts, we randomly sample one Hamiltonian in the ith part and compute its loss function $L_i\left(x_i, y_i, \boldsymbol{w}\right)$.

To update the parameters $\boldsymbol{w}$, we choose the momentum algorithm as the optimizer,[32] which is based on the stochastic gradient descent (SGD) algorithm;[33,34] the updated values $\boldsymbol{w}_{t+1}$ change from the current values $\boldsymbol{w}_t$ by the amount $\Delta \boldsymbol{w}_t$ as

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \Delta \boldsymbol{w}_t. \tag{8}$$

The changes are given by

$$\Delta \boldsymbol{w}_t = -\epsilon \nabla_{\boldsymbol{w}} \Big[\frac{1}{N}\sum_{i=1}^{N} L_i(x_i, y_i, W)\Big] + p\Delta \boldsymbol{w}_{t-1}, \tag{9}$$

where ϵ and p are two hyper-parameters: ϵ is the learning rate, p is the momentum. For our training, we take $\epsilon = 0.01$, $p = 0.9$. The optimum values of $\boldsymbol{w}_t$ are found by minimizing the difference between the output of the networks with respect to the known winding number for the test dataset.

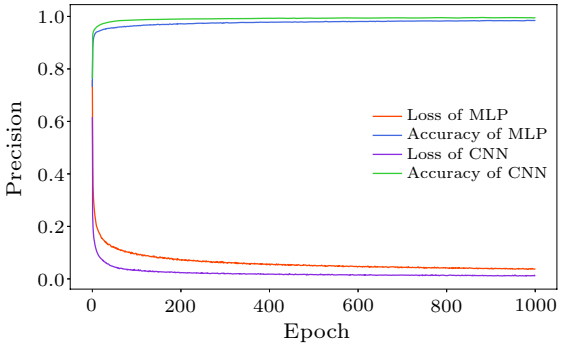

Fig. 3. The loss and accuracy of CNN and MLP networks through the training process.

The performance of the trained networks can be quantified by their accuracy of predicting correctly the topological classes of the Hamiltonians in the test dataset, the test dataset consists of 1000 Hamiltonians; the accuracy is calculated as the ratio between the number of correct predictions to the total number of predictions for the Hamiltonians in the test dataset. Figure 3 shows that the accuracy of MLP reaches 98.50% and CNN 99.40% respectively after training 1000 epochs.

In addition to the accuracy, confusion matrices[35] are widely used to reveal more information regarding the performance of trained networks in classification problems.[36] Figure 4 shows the confusion matrices for the trained MLP and CNN networks. In the figure, each row of the confusion matrix represents the actual classes that the test Hamiltonians should be in, while each column represents the classes that the trained networks predict the test Hamiltonians to be in. The elements of the confusion matrix are the number of the test Hamiltonians allocated.

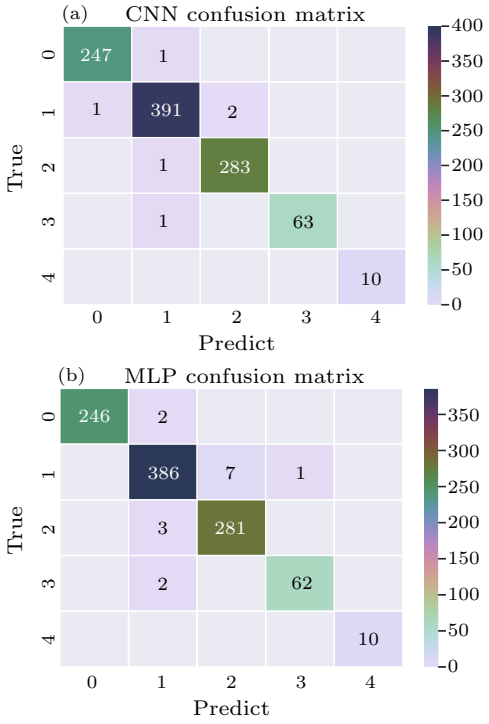

Fig. 4. (a) Confusion matrix of CNN. (b) Confusion matrix of MLP. The matrix elements are the number of the test Hamiltonians allocated.

Based on the confusion matrix, we can define the precision in each class, which is the ratio between the number of the Hamiltonians correctly assigned in the class denoted as N_{true} and the total number of the Hamiltonians assigned as N_{total}. Thus the precision of a class is given by

$$P = N_{\text{true}}/N_{\text{total}}. \tag{10}$$

Table 1 shows the precision of each class for CNN and MLP respectively; CNN exhibits slightly better precision than MLP in all classes.

Table 1. Precision of the CNN and MLP networks.

Winding number	0	1/2	1	3/2	2
CNN precision	99.59%	99.23%	99.64%	98.43%	100%
MLP precision	99.19%	97.96%	98.94%	96.87%	100%

To further demonstrate the capability of the trained networks in classifying the non-Hermitian Hamiltonians, we go beyond the previous test database and generate a new test database of 10^6 Hamiltonians in the parameter regime $-4 < \delta < 4$ and $-4 < t < 4$ with $\Delta = 1$, $t' = 1$. We use the CNN network trained previously to classify the Hamiltonians in this new test database. Figure 5 shows the phase diagram generated by CNN. Compared with Fig. 1 which we calculated numerically, CNN yields rather satis-

factory classification of the Hamiltonians outside the parameter regime of the original training database. Misclassified Hamiltonians concentrate in the vicinity of the topological phase transition boundaries, where the magnitude of the denominator in Eq. (5) becomes rather small at certain momentum k; large errors occur to the numerical calculation of the winding number by the networks.

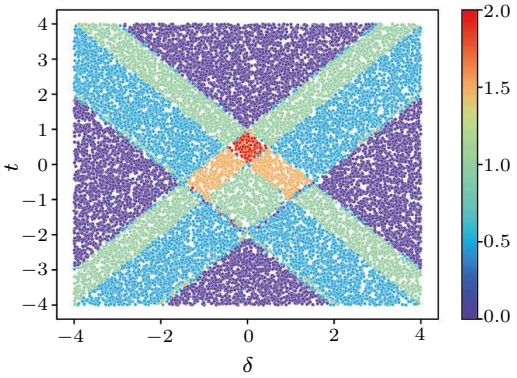

Fig. 5. The phase diagram of the non-Hermitian model, Eq. (1), generated by the trained CNN networks. Colors are used to represent different values of the winding number $\nu = 0, 1/2, 1, 3/2, 2$.

The capability of the trained networks demonstrated above suggests that the networks have acquired a way to numerically calculate the winding numbers of the non-Hermitian Hamiltonians, i.e., Eq. (6). To confirm this suggestion, we investigate the decision-making process of the trained networks by analyzing the intermediate outputs; in particular, we look into the outputs of the layer Conv4. There are 320 neurons in the layer Conv4, arranged in a 10×32 array. To be specific, we focus on one column of the layer Conv4, which consists of 32 neurons. Figure 6(a) plots the values of the 32 neurons for 20 test Hamiltonians.

Equation (5) is the continuous way to calculate the Zak phase. In the discrete version, the Zak phase can be likely calculated as

$$\gamma = \sum_{i=1}^{N} \Delta\phi(k_i) \tag{11}$$

with the winding angle difference $\Delta\phi(k_i) = \phi(k_i) - \phi(k_i - \Delta k)$, $k_i = i\Delta k$, $N = 2\pi/\Delta k$ and Δk being the difference to discretize $k \in [0, 2\pi)$. Since the input Hamiltonians take the matrix form of Eq. (6), it is natural to expect $N = 32$. For the 20 test Hamiltonian, we numerically calculated the winding angle differences $\{\Delta\phi(k_i)\}$ and compared them with the values of the 32 neurons one by one in order. Figure 6(b) shows a good linear correlation between them. Similar correlations persist to the other columns in the layer Conv4, though not shown here. The good linear correlation suggests that the convolution operation led to the successful calculation of $\{\Delta\phi(k_i)\}$, via which the networks compute the Zak phase of the non-Hermitian Hamiltonians in a discrete way in regard to Eq. (6). Previous studies on Hermitian Hamiltonians reach the same conclusion.[20,21]

The networks that we used here have redundancy. For example, we tried a smaller CNN network with 20407 training parameters and repeated the training. The accuracy of this smaller network reaches 99.27%, and the precision of $\nu = 0, 1/2, 1, 3/2, 2$ is $99.59\%, 99.74\%, 99.29\%, 96.87\%, 100\%$. However, when we tried an even smaller CNN network with 4483 training parameters, we found that the value of loss function and accuracy became unstable during the training process. We leave detailed investigation on how to optimize the network size to future study.

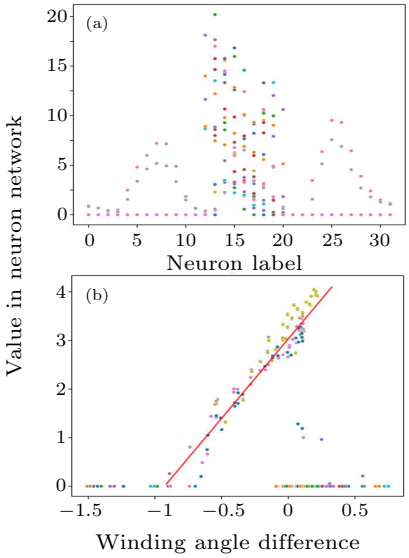

Fig. 6. (a) Values of the 32 neurons in a certain column in the layer Conv4 for 20 test Hamiltonians. (b) Neuron values versus the numerical values of the winding angle differences for 20 test Hamiltonians. The red line is a guide.

In summary, we have demonstrated that the deep learning technology is capable of learning the topological invariant for non-Hermitian Hamiltonians in one dimension. The trained CNN and MLP networks can predict the winding numbers of test Hamiltonians with high precision. Our study shows that there is a linear correlation between the intermediate data produced in the networks and the numerically calculated values for the winding angle differences. This correlation provides strong evidence that the trained networks have acquired the mathematical definition of the winding number though in a discrete form.

It is worth emphasizing that the topological invariant that we study here is the winding number in the Brillouin zone, while it is well known now that the bulk-boundary correspondence of non-Hermitian systems involves rather the winding number in the gen-

eralized Brillouin zone.[26,37] To calculate the winding number in the generalized Brillouin zone, one needs to replace the Bloch Hamiltonian $H(k)$ by $H(k+i\kappa)$ in the generalized Brillouin zone. This approach has recently been carried out in the work of machine learning "non-Bloch winding number" in Ref. [38].

Note added. Since this work was largely completed, we became aware of Ref. [39], in which the authors applied machine learning to study topological phases of non-Hermitian models different from ours.

References

[1] Gong Z, Ashida Y, Kawabata K, Takasan K, Higashikawa S and Ueda M 2018 *Phys. Rev. X* **8** 031079
[2] Song F, Yao S Y and Wang Z 2019 *Phys. Rev. Lett.* **123** 246801
[3] Esaki K, Sato M, Hasebe K and Kohmoto M 2011 *Phys. Rev. B* **84** 205128
[4] Lieu S 2018 *Phys. Rev. B* **97** 045106
[5] Moiseyev N 2011 *Non-Hermitian Quantum Mechanics* (Cambridge: Cambridge University Press)
[6] Xie D, Gou W, Xiao T *et al.* 2019 *npj. Quantum Inf.* **5** 55
[7] Dirac 1953 *Physica* **19** 1
[8] Berry M V 1984 *Proc. R. Soc. London.* **392** 45
[9] Heiss W D 2004 *J. Phys. A* **37** 2455
[10] Xu Y, Wang S T and Duan L M 2017 *Phys. Rev. Lett.* **118** 045701
[11] Deng D L, Li X P and Das S 2017 *Phys. Rev. B* **96** 195145
[12] Narayan B and Narayan A 2021 *Phys. Rev. B* **103** 035413
[13] Juan C and Roger G M 2017 *Nat. Phys.* **13** 431
[14] Ohtsuki T and Tomi O 2016 *J. Phys. Soc. Jpn.* **85** 123706
[15] Arai S, Ohzeki M and Tanaka K 2018 *J. Phys. Soc. Jpn.* **87** 033001
[16] Broecker P, Carrasquilla J, Melko R G *et al.* 2017 *Sci. Rep.* **7** 8823
[17] Zhang Y, Paul G and Kim E A 2020 *Phys. Rev. Res.* **2** 023283
[18] Wang C and Zhai H 2017 *Phys. Rev. B* **96** 144432
[19] Gao J and Qiao L F 2018 *Phys. Rev. Lett.* **120** 240501
[20] Zhang P F, Shen H T and Zhai H 2018 *Phys. Rev. Lett.* **120** 066401
[21] Sun N, Yi J M, Zhang P F, Shen H T and Zhai H 2018 *Phys. Rev. B* **98** 085402
[22] Ghatak A and Das T 2019 *J. Phys.: Condens. Matter* **31** 263001
[23] Chiu C K and Ryu S 2016 *Rev. Mod. Phys.* **88** 035005
[24] Yin C H, Jiang H, Li L H, Rong L and Chen S 2018 *Phys. Rev. A* **97** 052115
[25] Flore K K, Elisabet E, Jan C B and Emil J B 2018 *Phys. Rev. Lett.* **121** 026808
[26] Yao S Y and Wang Z 2018 *Phys. Rev. Lett.* **121** 086803
[27] Noriega L 2005 *School of Computing, Staffordshire University* (Multilayer Perceptron Tutorial)
[28] LeCun Y and Boser B 1989 *Neural Comput.* **1** 541
[29] Ian G, Yoshua B and Aaron C 2016 *Deep Learning* (Cambridge, USA: The MIT Press)
[30] Xavier G and Yoshua B 2010 *JMLR Workshop and Conference Proceedings* **9** 249
[31] Ketkar N 2017 *Deep Learning with Python* (Apress, Berkeley, USA: Manning Publications) p 97
[32] Polyak B T 1964 *USSR Comput. Math. Math. Phys.* **4** 1
[33] Niu Q 1999 *Neural Netw.* **12** 145
[34] Bottou L 2012 *Stochastic Gradient Descent Tricks* (Berlin: Springer) p 421
[35] Stehman S V 1997 *Remote Sens. Environ.* **62** 77
[36] Deng X Y, Liu Q, Deng Y and Mahadevand S 2016 *Inf. Sci.* **340** 250
[37] Kazuki Y and Shuichi M 2019 *Phys. Rev. Lett.* **123** 066404
[38] Yu L W and Deng D L 2021 *Phys. Rev. Lett.* **126** 240402
[39] Zhang L F, Tang L Z, Huang Z H, Zhang G Q, Huang W and Zhang D W 2021 *Phys. Rev. A* **103** 012419