Supplementary Material for "Deep learning Local Reduced Density Matrices for Many-body Hamiltonian Estimation"

Xinran Ma[1], Z. C. Tu[1], and Shi-Ju Ran[2,*]

[1]Department of Physics, Beijing Normal University, Beijing 100875, China
[2]Department of Physics, Capital Normal University, Beijing 100048, China

[*]Corresponding author e-mail: sjran@cnu.edu.cn

October 5, 2021

In the supplementary material, more details about the Qubism map and CNN are provided, and the images obtained by the Qubism map from various models in 1D and 2D are shown. The detailed descriptions and some supplementary data on applying QubismNet to 2D quantum systems on the square and breathing kagome lattices are given.

1 Qubism Map

Consider a quantum system with L spins, denoted by $s = \{X_1Y_1X_2Y_2\dots X_{L/2}Y_{L/2}\}$ with $X_i, Y_i \in \{0, 1\}$. The resolution of the image obtained by the Qubism map is $(2^{L/2}, 2^{L/2})$. Each spin configuration corresponds to the pixel in the x-th row and y-th column of the image satisfying

$$\begin{aligned} x &= \sum_{i=1}^{L/2} X_i 2^{(L/2-i)} + 1, \\ y &= \sum_{i=1}^{L/2} Y_i 2^{(L/2-i)} + 1. \end{aligned} \tag{S1}$$

The gray-scale value of each pixel is taken as the coefficient of the quantum state in the corresponding spin configuration. Besides, each pixel can be attached to colors based on the phase of quantum state. The images obtained by the Qubism map of 1D QIM, Heisenberg XY, and XXZ models are shown in Figure S1 (a), (b) and (c), respectively. In all these cases, we take $L = 64$ and $L_b = 8$.

2 Convolutional Neural Network

CNN has shown the advantage in image recognition and many other challenging tasks. In general, CNN consists of several alternative convolutional and pooling layers, which together serve as a feature extractor. One or many fully-connected layers are then used to map the extracted features to the target output in classification and regression. The CNN we use in this work contains eight layers. The first convolutional layer filters the input images with 32 kernels of size 3×3 and a stride of size 1×1. The second convolutional layer take the output of the first convolutional layer as input with same setting as the first convolutional layer. The first max-pooling layer of size 2×2 follows which downsamples the size of features. Then the third and fourth convolutional layers are used both with 64 kernels of size 3×3 and a stride of size 1×1. All convolutional layers use padding around the images so the outputs has the same height/width dimension as the inputs. The second max-pooling layer follows with pool size 2×2. Next, the output of the second max-pooling layer is flattened and then input to two fully-connected layers with 128 and 32 neurons respectively. The output of the last fully connected layer is fed to one neuron which produce the final output of the physical parameter. Note that we choose two successive convolutional layers to improve the quality of features because of the more complex nonlinearity and larger receptive fields. The rectified linear unit (ReLU) [1, 2] is chosen as activation function for all the convolutional and fully-connected layers. A linear activation function is used for the output layer. The optimizer we use is RMSprop [3] with the initial learning rate 0.001. "He normal initialization" is employed as initialization method which has been proven to get good effect in CNN with the rectifier nonlinearities [4].

We split data into four datasets which are training, validation, testing and generalizing set. The training set is used to train the CNN model. The validation set is randomly taken from the training set with the percentage of 10%. The validation set has two roles

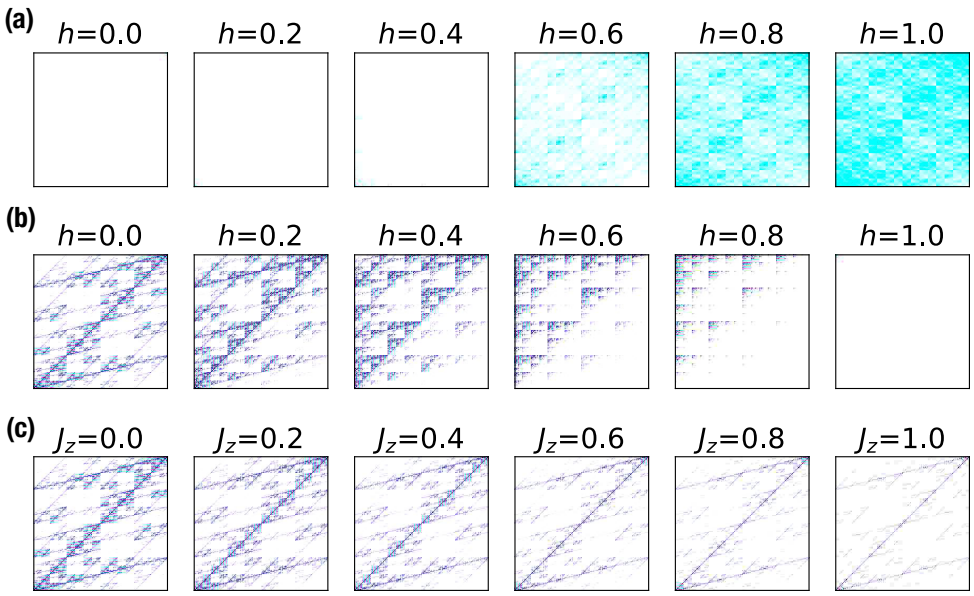

Figure S1: (Color online) Illustrations of the images by the Qubism map from the ground states of the QIM, XY, and XXZ models.

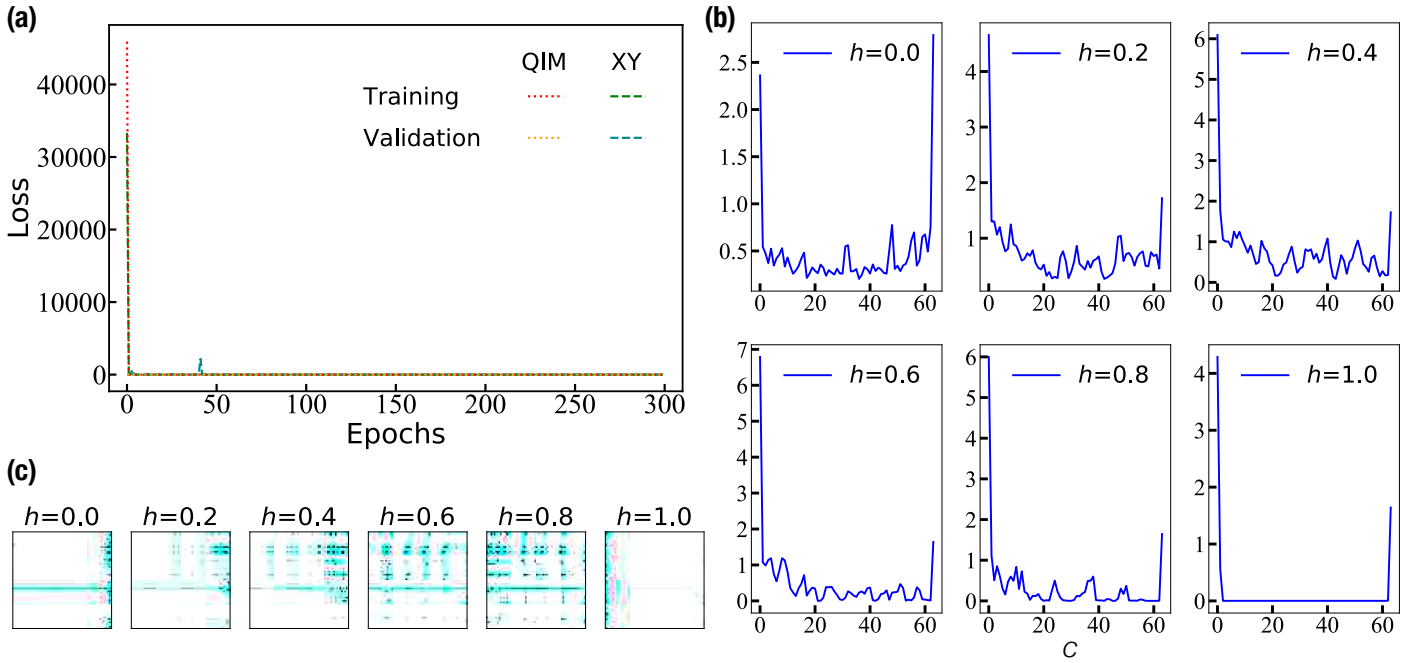

Figure S2: (Color online) The loss functions and extracted features of the CNN. (a) The decreases of the loss function with the training epochs on training and validation set for QIM, XY and XXZ models. (b) For the XY model, we show the average magnitude of the extracted features for different channels c. For different h, the dominant contribution is always from the first channel. (c) The extracted 64×64 features in the first channel. In general, we observe that the prominent features (illustrated by the dark dots) move from right to left as h increases.

in our work. One is choosing the proper hyper-parameters such as the number of neurons in each layer, the optimizer, and so on. The other is the early-stopping technique. The testing set and the generalizing set consist of data which never show in the training set. The difference is that data in the testing set have same range of the training set however the generalizing range is beyond the training range.

Two tricks on avoiding overfitting during the training process is employed. One is the dropout [5, 6] with rate of $p = 0.5$, meaning randomly masking 50% of neurons during training. It is a commonly used method making the NN robust. The other is validation. We always save the best model selected by the minimum loss of the validation set which consists of the samples randomly chosen 10% from the training set. The best model is then used on the testing and generalizing set. Note that the input images are normalized so that the features are in $[0, 1]$. We construct and train our CNN with Keras [7], a high-level open-source API running on top of TensorFlow. We train the CNN for 300 epochs for each case (one epoch means training by all the samples in training set once). The total training time is roughly 0.3 hours on a server equipped with NVIDIA P100 graphics processing unit when we use 1000 training samples.

Figure S2 (a) shows the loss functions versus epochs for the 1D QIM and XY models with $L = 64$, $L_b = 8$ and $\delta = 0.4$. The loss functions of the training and validation sets decrease rapidly and become quite small in a few epochs for all the three models. This indicates that QubismNet is trainable and feasible. Besides, the validation loss functions do not increase during the whole training process implying no overfitting.

Furthermore, we go deep inside the CNN and see extracted features by the convolutional/pooling part of the CNN. After the last pooling layer, each sample is mapped to a $64 \times 64 \times 64$ tensor as the extracted features where 64 is the number of channels. Figure S2 (b) shows the average magnitude of each channel c. We find the the dominant contribution is from the first channel in almost all the cases. In Figure S2 (c), we demonstrate the 64×64 features in the first channel. Interestingly, we observe that the prominent features (with larger values marked by the deeper dots) move in general from right to left with the increase of h. These results indicate that QubismNet capture some consecutive rule from the quantum states in its own way.

All codes used in the manuscript and this supplementary material can be found on GitHub [8].

3 RDM-based method for two-dimensional lattice

We follow the standard recipe of DMRG on solving the ground states of 2D quantum models [9]. The 2D lattice model with nearest-neighboring interactions is stretched into a chain with long-range interactions, as illustrated in Figure S3 (a). We set the size of lattice as 4×16 for both the 2D XY and XXZ models. The subsystem used to calculate RDM is chosen in the middle of lattice with size 4×2 as illustrated. For the purified state of ρ^2, we treat all the degrees of freedom in the *bra* space as one of the two dimensions of the 2D image, and treat those in the *ket* space as the other dimension of the image. This ordering is illustrated in Figure S3 (b). Several images obtained from the ground states of the 2D XY and XXZ models with different h or J_z are shown in Figure S3 (c) and (d) as examples.

4 QubismNet for breathing kagome antiferromagnet

We also tried QubismNet for the non-trivial breathing kagome antiferromagnet with the Hamiltonian

$$\hat{H} = J_{\triangle} \sum_{\langle i,j\rangle \in \triangle} (\hat{S}_i^x \hat{S}_j^x + \hat{S}_i^y \hat{S}_j^y + \hat{S}_i^z \hat{S}_j^z) + J_{\triangledown} \sum_{\langle i,j\rangle \in \triangledown} (\hat{S}_i^x \hat{S}_j^x + \hat{S}_i^y \hat{S}_j^y + \hat{S}_i^z \hat{S}_j^z), \tag{S2}$$

where $J_{\triangle}$ and $J_{\triangledown}$ represent the coupling strength in the up and down triangles, respectively. The illustration of the kagome lattice with periodic boundary condition is shown in Figure S4 (a). The number of columns is three and the image size after the qubism map is 512×512. The breathing anisotropy $J_{\triangledown}/J_{\triangle}$ is chosen to vary between 0.4 and 1.0. The ground states are magnetically disordered RVB states and solved by DMRG algorithm. Figure S4 (b) shows the results of predicting the breathing anisotropy on the testing and generalizing sets. The mean square errors are of the order of magnitude $O(10^{-4})$. This demonstrates the validity of the QubismNet on the nontrivial kagome model with magnetically disordered ground states.

References

[1] V. Nair, G. E. Hinton, Rectified linear units improve restricted boltzmann machines, in: ICML, 2010.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Communications of the ACM 60 (6) (2017) 84–90.

[3] G. Hinton, N. Srivastava, K. Swersky, Neural networks for machine learning lecture 6a overview of mini-batch gradient descent, Cited on 14 (8) (2012).

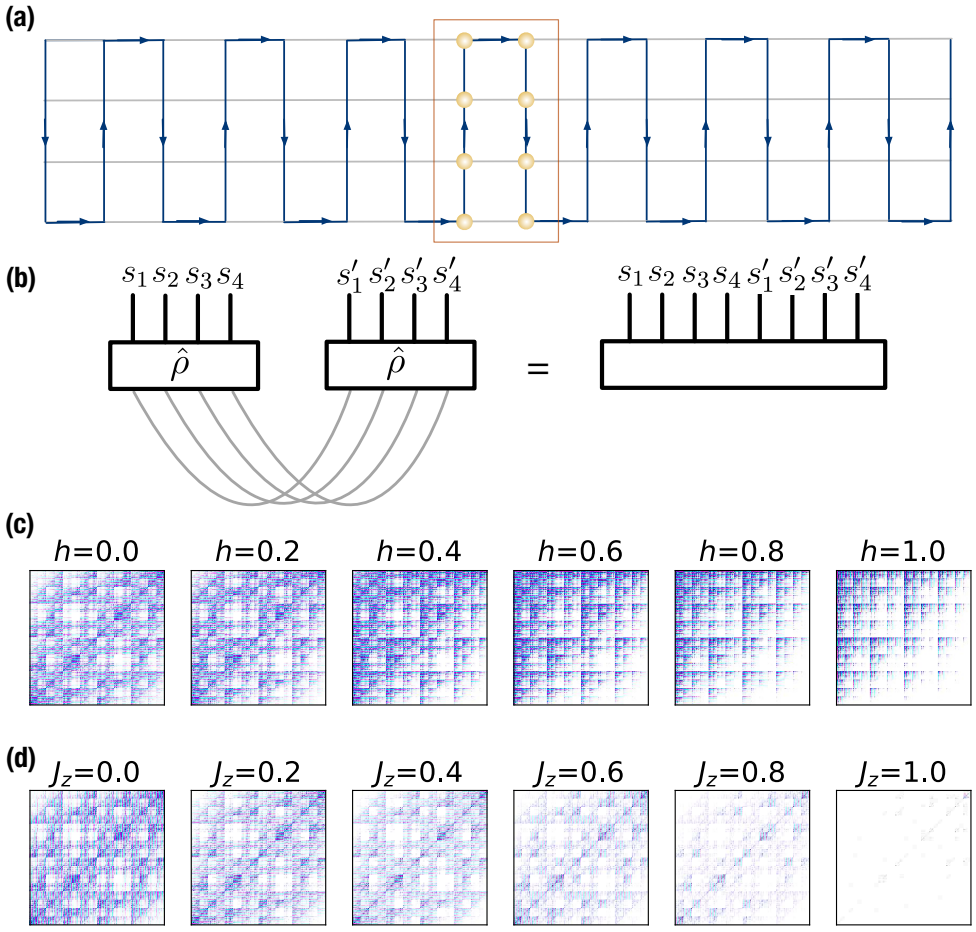

Figure S3: (Color online) An illustration of DMRG applied to the 2D system, and the images obtained from the 2D ground states. (a) An illustration of how the 2D lattice is stretched to a 1D chain in order to use DMRG to simulate the ground states. We choose a 2×4 sub-system in the middle to define the RDM. (b) An illustration of how the reduced density matrices are constructed. (c) The images obtained by applying the Qubism map to the ground states in the 2D XX model in different transverse fields h. (d) The images from the 2D XXZ model with different J_z.

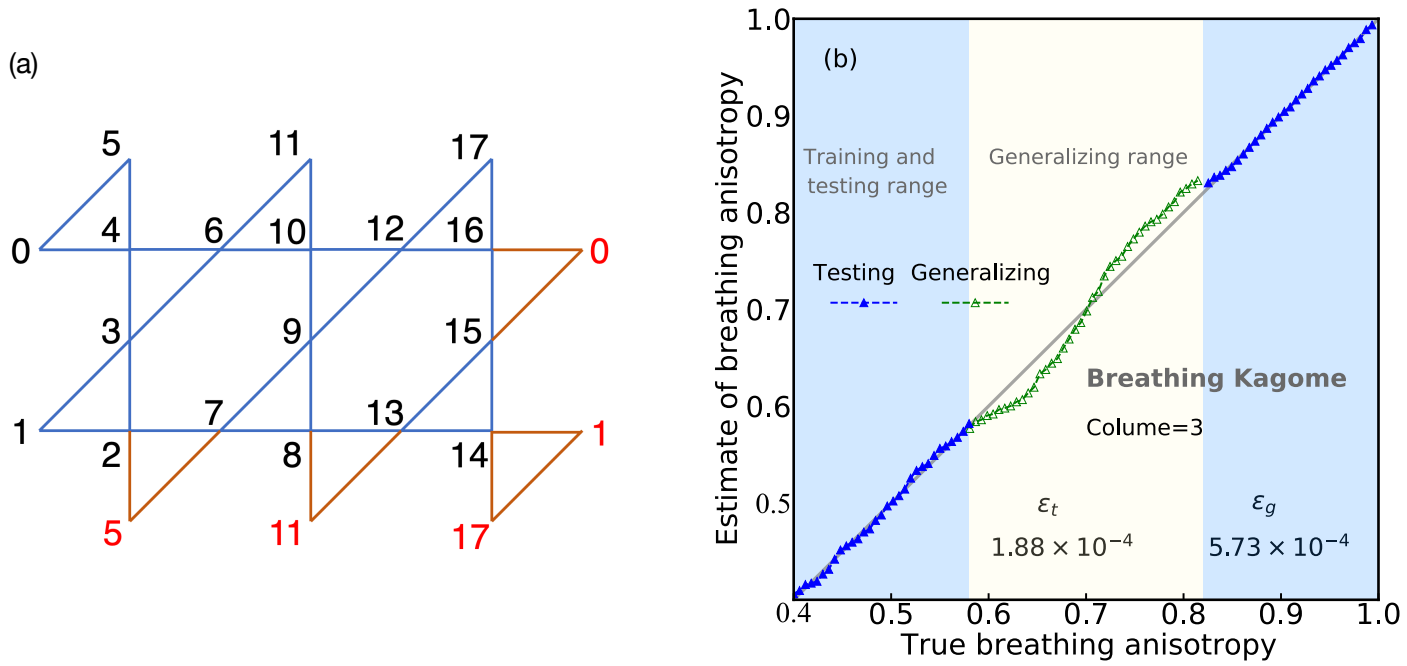

Figure S4: (Color online) (a) An illustration of the breathing kagome antiferromagnet, where the couplings of the upper and lower triangles are antisymmetric. The number of columns is three. Periodic boundary condition is chosen as shown in the red edges. (b) shows the estimations of the breathing anisotropy versus its ground truth. We use the RDM trick with $L_b = 8$.

[4] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.

[5] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, arXiv preprint arXiv:1207.0580 (2012).

[6] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, The Journal of Machine Learning Research 15 (1) (2014) 1929–1958.

[7] F. Chollet, et al., Keras, https://keras.io (2015).

[8] https://github.com/JoyeBNU/QubismNet.

[9] E. M. Stoudenmire, S. R. White, Studying two-dimensional systems with the density matrix renormalization group, Ann. Rev. Cond. Matter Phys. 3 (2012) 111–128.