



Quantum Convolutional Neural Networks

Citation

Cong, Iris, Soonwon Choi, and Mikhail Lukin. 2019. Quantum Convolutional Neural Networks. Nature Physics 15, no. 12: 1273–1278.

Permanent link

<http://nrs.harvard.edu/urn-3:HUL.InstRepos:42594545>

Terms of Use

This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA>

Share Your Story

The Harvard community has made this article openly available. Please share how this access benefits you. [Submit a story](#).

[Accessibility](#)

Quantum Convolutional Neural Networks

Iris Cong,¹ Soonwon Choi,^{1,2,*} and Mikhail D. Lukin¹

¹*Department of Physics, Harvard University, Cambridge, Massachusetts 02138, USA*

²*Department of Physics, University of California, Berkeley, CA 94720, USA*

We introduce and analyze a novel quantum machine learning model motivated by convolutional neural networks. Our quantum convolutional neural network (QCNN) makes use of only $O(\log(N))$ variational parameters for input sizes of N qubits, allowing for its efficient training and implementation on realistic, near-term quantum devices. The QCNN architecture combines the multi-scale entanglement renormalization ansatz and quantum error correction. We explicitly illustrate its potential with two examples. First, QCNN is used to accurately recognize quantum states associated with 1D symmetry-protected topological phases. We numerically demonstrate that a QCNN trained on a small set of exactly solvable points can reproduce the phase diagram over the entire parameter regime and also provide an exact, analytical QCNN solution. As a second application, we utilize QCNNs to devise a quantum error correction scheme optimized for a given error model. We provide a generic framework to simultaneously optimize both encoding and decoding procedures and find that the resultant scheme significantly outperforms known quantum codes of comparable complexity. Finally, potential experimental realization and generalizations of QCNNs are discussed.

Machine learning based on neural networks has recently provided significant advances for many practical applications¹. In physics, one natural application involves the study of quantum many-body systems, where the extreme complexity of many-body states often makes theoretical analysis intractable. This has led to a number of recent works using machine learning to study properties of quantum systems^{2–7}, using physical concepts to interpret machine learning^{8,9}, or using quantum computers to enhance conventional machine learning tasks^{10–13}.

In this work, motivated by the progress towards realizing quantum information processors^{14–17}, we bridge these approaches by proposing a quantum circuit model inspired by machine learning and illustrating its success for two important classes of quantum many-body problems. The first class of problems we consider is *quantum phase recognition* (QPR), which asks whether a given input quantum state ρ_{in} belongs to a particular quantum phase of matter. Critically, in contrast to many existing schemes based on tensor network descriptions^{18–20}, we assume ρ_{in} is prepared in a physical system without direct access to its classical description. The second class, *quantum error correction (QEC) optimization*, asks for an optimal QEC code for a given, *a priori* unknown error model such as dephasing or potentially correlated depolarization in realistic experimental settings.

The highly complex and intrinsically quantum nature of these problems makes them particularly difficult to solve using existing classical and quantum machine learning techniques. While conventional machine learning with large-scale neural networks can successfully solve analogous classical problems such as image recognition or improving classical error correction¹, the exponentially large many-body Hilbert space hinders efficiently translating such quantum problems into a classical framework without performing exponentially difficult quantum state or process tomography²¹. Quantum algorithms avoid this overhead, but the limited size and coherence times of near-term quantum devices prevent the use of large-scale

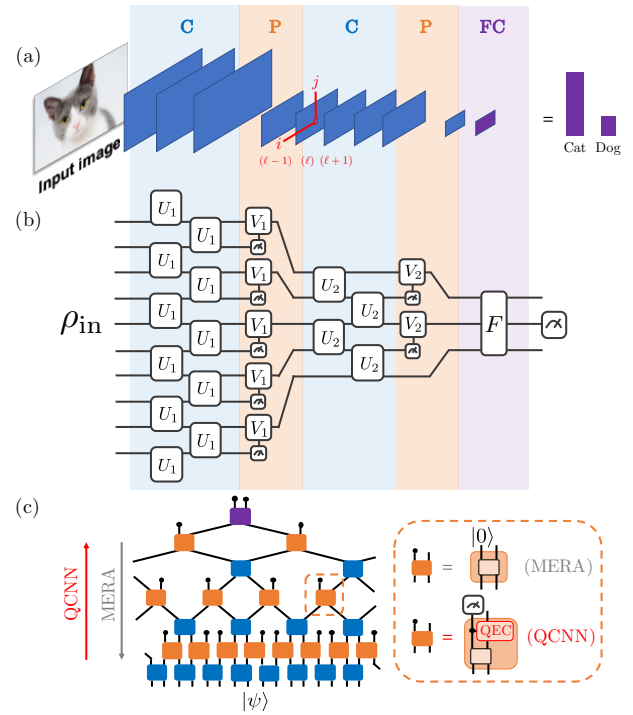


Figure 1: (a) Simplified illustration of CNNs. A sequence of image processing layers—convolution (C), pooling (P), and fully connected (FC)—transforms an input image into a series of feature maps (blue rectangles), and finally into an output probability distribution (purple bars). (b) QCNNs inherit a similar layered structure. (c) QCNN and MERA share the same circuit structure, but run in reverse directions.

networks; thus, it is vital to first theoretically understand the most important machine learning mechanisms that must be implemented. In this work, we introduce a quantum machine learning method for QPR and QEC optimization, provide both theoretical insight and numerical demonstrations for its success, and show its feasibility for near-term experimental implementation.

QCNN CIRCUIT MODEL

Convolutional neural networks (CNNs) provide a successful machine learning architecture for classification tasks such as image recognition^{1,22,23}. A CNN generally consists of a sequence of different (interleaved) layers of image processing; in each layer, an intermediate 2D array of pixels, called a feature map, is produced from the previous one (Figure 1a)²⁴. The convolution layers compute new pixel values $x_{ij}^{(\ell)}$ from a linear combination of nearby ones in the preceding map $x_{i,j}^{(\ell)} = \sum_{a,b=1}^w w_{a,b} x_{i+a,j+b}^{(\ell-1)}$, where the weights $w_{a,b}$ form a $w \times w$ matrix. Pooling layers reduce feature map size, e.g. by taking the maximum value from a few contiguous pixels, and are often followed by application of a nonlinear (activation) function. Once the feature map size becomes sufficiently small, the final output is computed from a function that depends on all remaining pixels (fully connected layer). The weights and fully connected function are optimized by training on large datasets. In contrast, variables such as the number of convolution and pooling layers and the size w of the weight matrices (known as hyperparameters) are fixed for a specific CNN¹. CNN's key properties are thus translationally invariant convolution and pooling layers, each characterized by a constant number of parameters (independent of system size), and sequential data size reduction (i.e., a hierarchical structure).

Motivated by this architecture we introduce a quantum circuit model extending these key properties to the quantum domain (Fig. 1b). The circuit's input is an unknown quantum state ρ_{in} . A convolution layer applies a single quasi-local unitary (U_i) in a translationally-invariant manner for finite depth. For pooling, a fraction of qubits are measured, and their outcomes determine unitary rotations (V_j) applied to nearby qubits. Hence, nonlinearities in QCNN arise from reducing the number of degrees of freedom. Convolution and pooling layers are performed until the system size is sufficiently small; then, a fully connected layer is applied as a unitary F on the remaining qubits. Finally, the outcome of the circuit is obtained by measuring a fixed number of output qubits. As in the classical case, circuit structures (i.e. QCNN hyperparameters) such as the number of convolution and pooling layers are fixed, and the unitaries themselves are learned.

A QCNN to classify N -qubit input states is thus characterized by $O(\log(N))$ parameters. This corresponds to doubly exponential reduction compared to a generic quantum circuit-based classifier¹² and allows for efficient learning and implementation. For example, given classified training data $\{(|\psi_\alpha\rangle, y_\alpha) : \alpha = 1, \dots, M\}$, where $|\psi_\alpha\rangle$ are input states and $y_\alpha = 0$ or 1 are corresponding binary classification outputs, one could compute the mean-squared error

$$\text{MSE} = \frac{1}{2M} \sum_{\alpha=1}^M (y_\alpha - f_{\{U_i, V_j, F\}}(|\psi_\alpha\rangle))^2. \quad (1)$$

Here, $f_{\{U_i, V_j, F\}}(|\psi_\alpha\rangle)$ denotes the expected QCNN output value for input $|\psi_\alpha\rangle$. Learning then consists of initializing all unitaries and successively optimizing them until convergence, e.g. via gradient descent.

To gain physical insight into the mechanism underlying QCNNs and motivate their application to the problems under consideration, we now relate our circuit model to two well-known concepts in quantum information theory—the multiscale entanglement renormalization ansatz²⁵ (MERA) and quantum error correction (QEC). The MERA framework provides an efficient tensor network representation of many classes of interesting many-body wavefunctions, including those associated with critical systems^{25–27}. A MERA can be understood as a quantum state generated by a sequence of unitary and isometry layers applied to an input state (e.g. $|00\rangle$). While each isometry layer introduces a set of new qubits in a predetermined state (e.g. $|0\rangle$) before applying unitary gates on nearby ones, unitary layers simply apply quasi-local unitary gates to the existing qubits (Figure 1c). This exponentially growing, hierarchical structure allows for the long-range correlations associated with critical systems. The QCNN circuit has similar structure, but runs in the reverse direction. Hence, for any given state $|\psi\rangle$ with a MERA representation, there is always a QCNN that recognizes $|\psi\rangle$ with deterministic measurement outcomes; one such QCNN is simply the inverse of the MERA circuit.

For input states other than $|\psi\rangle$, however, such a QCNN does not generally produce deterministic measurement outcomes. These additional degrees of freedom distinguish QCNN from MERA. Specifically, we can identify the measurements as syndrome measurements in QEC²⁸, which determine error correction unitaries V_j to apply to the remaining qubit(s). Thus, a QCNN circuit with multiple pooling layers can be viewed as a combination of MERA — an important variational ansatz for many-body wavefunctions — and nested QEC — a mechanism to detect and correct local quantum errors without collapsing the wavefunction. This makes QCNN a powerful architecture to classify input quantum states or devise novel QEC codes. In particular, for QPR, the QCNN can provide a MERA realization of a representative state $|\psi_0\rangle$ in the target phase. Other input states within the same phase can be viewed as $|\psi_0\rangle$ with local errors, which are repeatedly corrected by the QCNN in multiple layers. In this sense, the QCNN circuit can mimic renormalization-group (RG) flow, a methodology which successfully classifies many families of quantum phases²⁹. For QEC optimization, the QCNN structure allows for simultaneous optimization of efficient encoding and decoding schemes with potentially rich entanglement structure.

DETECTING A 1D SPT PHASE

We first demonstrate the potential of QCNN explicitly by applying it to QPR in a class of one-dimensional

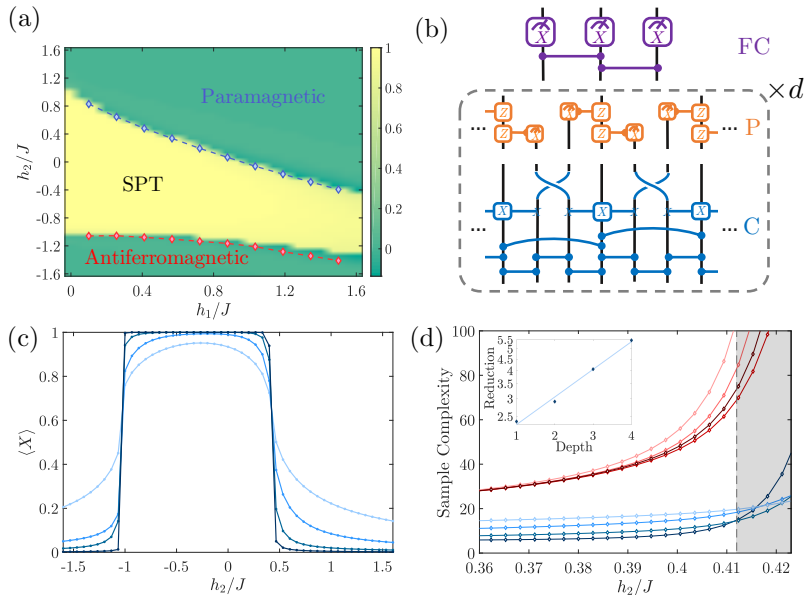


Figure 2: (a) The phase diagram of the Hamiltonian in the main text. The phase boundary points (blue and red diamonds) are extracted from infinite size DMRG numerical simulations, while the color represents the output from the exact QCNN circuit for input size $N = 45$ spins (see Methods). (b) Exact QCNN circuit to recognize a $\mathbb{Z}_2 \times \mathbb{Z}_2$ SPT phase. Blue line segments represent controlled-phase gates, blue three-qubit gates are Toffoli gates with the control qubits in the X basis, and orange two-qubit gates flip the target qubit's phase when the X measurement yields -1 . The fully connected layer applies controlled-phase gates followed by an X_i projection, effectively measuring $Z_{i-1}X_iZ_{i+1}$. (c) Exact QCNN output along $h_1 = 0.5J$ for $N = 135$ spins, $d = 1, \dots, 4$. (d) Sample complexity of QCNN at depths $d = 1, \dots, 4$ (blue) versus SOPs of length $N/2, N/3, N/5$, and $N/6$ (red) to detect the SPT/paramagnet phase transition along $h_1 = 0.5J$ for $N = 135$ spins. The critical point is identified as $h_2/J = 0.423$ using infinite size DMRG (bold line). Darkening colors show higher QCNN depth or shorter string lengths. In the shaded area, the correlation length exceeds the system size and finite-size effects can considerably affect our results. Inset: The ratio of SOP sample complexity to QCNN sample complexity is plotted as a function of depth d on a logarithmic scale for $h_1/J = 0.3918$. In the numerically accessible regime, this reduction of sample complexity scales exponentially as $1.73e^{0.28d}$ (trendline).

many-body systems. Specifically, we consider a $\mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry-protected topological (SPT) phase \mathcal{P} , a phase containing the $S = 1$ Haldane chain³⁰, and ground states $\{|\psi_G\rangle\}$ of a family of Hamiltonians on a spin-1/2 chain with open boundary conditions:

$$H = -J \sum_{i=1}^{N-2} Z_i X_{i+1} Z_{i+2} - h_1 \sum_{i=1}^N X_i - h_2 \sum_{i=1}^{N-1} X_i X_{i+1}. \quad (2)$$

X_i, Z_i are Pauli operators for the spin at site i , and the $\mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry is generated by $X_{\text{even(odd)}} = \prod_{i \in \text{even(odd)}} X_i$. Figure 2a shows the phase diagram as a function of $(h_1/J, h_2/J)$. When $h_2 = 0$, the Hamiltonian is exactly solvable via Jordan-Wigner transformation²⁹, confirming that \mathcal{P} is characterized by nonlocal order parameters. When $h_1 = h_2 = 0$, all terms are mutually commuting, and a ground state is the 1D cluster state. Our goal is to identify whether a given, unknown ground state drawn from the phase diagram belongs to \mathcal{P} .

As an example, we first present an exact, analytical QCNN circuit that recognizes \mathcal{P} , see Figure 2b. The convolution layers involve controlled-phase gates as well as Toffoli gates with controls in the X -basis, and pooling

layers perform phase-flips on remaining qubits when one adjacent measurement yields $X = -1$. This convolution-pooling unit is repeated d times, where d is the QCNN depth. The fully connected layer measures $Z_{i-1}X_iZ_{i+1}$ on the remaining qubits. Figure 2c shows the QCNN output for a system of $N = 135$ spins and $d = 1, \dots, 4$ along $h_2 = 0.5J$, obtained using matrix product state simulations. As d is increased, the measurement outcomes show sharper changes around the critical point, and the output of a $d = 2$ circuit already reproduces the phase diagram with high accuracy (Figure 2a). This QCNN can also be used for other Hamiltonian models belonging to the same phase, such as the $S = 1$ Haldane chain³⁰ (see Methods).

Sample Complexity

The performance of a QPR solver can be quantified by sample complexity²¹: what is the expected number of copies of the input state required to identify its quantum phase? We demonstrate that the sample complexity of our exact QCNN circuit is significantly better than that of conventional methods. In principle, \mathcal{P} can be detected by measuring a nonzero expectation value of string order

parameters (SOP)^{31,32} such as

$$S_{ab} = Z_a X_{a+1} X_{a+3} \dots X_{b-3} X_{b-1} Z_b. \quad (3)$$

In practice, however, the expectation values of SOP vanish near the phase boundary due to diverging correlation length³²; since quantum projection noise is maximal in this vicinity, many experimental repetitions are required to affirm a nonzero expectation value. In contrast, the QCNN output is much sharper near the phase transition, so fewer repetitions are required.

Quantitatively, given some $|\psi_{\text{in}}\rangle$ and SOP S , a projective measurement of S can be modeled as a (generalized) Bernoulli random variable, where the outcome is 1 with probability $p = (\langle \psi_{\text{in}} | S | \psi_{\text{in}} \rangle + 1)/2$ and -1 with probability $1-p$ (since $S^2 = \mathbf{1}$); after M binary measurements, we estimate p . $p > p_0 = 0.5$ signifies $|\psi_{\text{in}}\rangle \in \mathcal{P}$. We define the sample complexity M_{min} as the minimum M to test whether $p > p_0$ with 95% confidence using an arcsine variance-stabilizing transformation³³:

$$M_{\text{min}} = \frac{1.96^2}{(\arcsin \sqrt{p} - \sqrt{\arcsin p_0})^2}. \quad (4)$$

Similarly, the sample complexity for a QCNN can be determined by replacing $\langle \psi_{\text{in}} | S | \psi_{\text{in}} \rangle$ by the QCNN output expectation value in the expression for p .

Figure 2d shows the sample complexity for the QCNN at various depths and SOPs of different lengths. Clearly, QCNN requires substantially fewer input copies throughout the parameter regime, especially near criticality. More importantly, although the SOP sample complexity scales independently of string length, the QCNN sample complexity consistently improves with increasing depth and is only limited by finite size effects in our simulations. In particular, compared to SOPs, QCNN reduces sample complexity by a factor which scales exponentially with the QCNN’s depth in numerically accessible regimes (inset). Such scaling arises from the iterative QEC performed at each depth and is not expected from any measurements of simple (potentially nonlocal) observables. We show in Methods that our QCNN circuit measures a *multiscale* string order parameter—a sum of products of exponentially many different SOPs which remains sharp up to the phase boundary.

MERA and QEC

Additional insights into the QCNN’s performance are revealed by interpreting it in terms of MERA and QEC. In particular, our QCNN is specifically designed to contain the MERA representation of the 1D cluster state ($|\psi_0\rangle$)—the ground state of H with $h_1 = h_2 = 0$ —such that it becomes a stable fixed point. When $|\psi_0\rangle$ is fed as input, each convolution-pooling unit produces the same state $|\psi_0\rangle$ with reduced system size in the unmeasured qubits, while yielding deterministic outcomes ($X = 1$) in the measured qubits. The fully connected layer measures

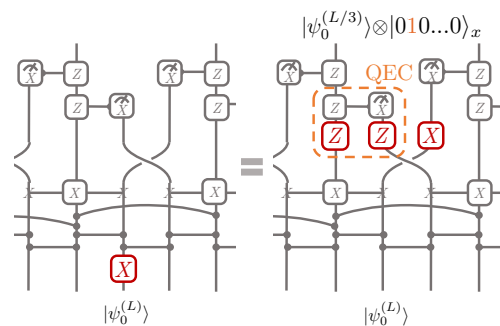


Figure 3: Action of cluster model QCNN convolution-pooling unit on a state with a single-qubit X error.

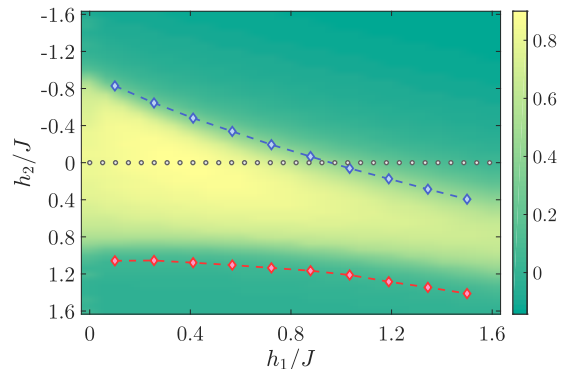


Figure 4: Output of a randomly initialized and trained QCNN for $N = 15$ spins and depth $d = 1$. Gray dots show (most of)³⁵ the training data points, which are 40 equally spaced points on a line where the Hamiltonian is solvable by Jordan-Wigner transformation ($h_2 = 0, h_1 \in [0, 2]$). The blue and red diamonds are phase boundary points extracted from infinite size DMRG numerical simulations, while the colors represent the expectation value of the QCNN output.

the SOP for $|\psi_0\rangle$. When an input wavefunction is perturbed away from $|\psi_0\rangle$, our QCNN corrects such “errors.” For example, if a single X error occurs, the first pooling layer identifies its location, and controlled unitary operations correct the error propagated through the circuit (Fig. 3). Similarly, if an initial state has multiple, sufficiently separated errors (possibly in coherent superpositions), the error density after several iterations of convolution and pooling layers will be significantly smaller³⁴. If the input state converges to the fixed point, our QCNN classifies it into the SPT phase with high fidelity. Clearly, this mechanism resembles the classification of quantum phases based on renormalization-group (RG) flow.

Obtaining QCNN from Training Procedure

Having analytically illustrated the computational power of the QCNN circuit model, we now demonstrate how a QCNN for \mathcal{P} can also be obtained using the learning procedure. In our example, the QCNN’s hyperparameters are chosen such that there are four convo-

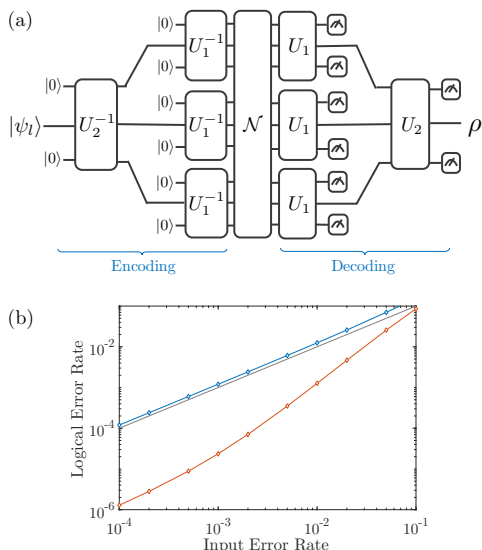


Figure 5: (a) Schematic diagram for using QCNNs to optimize QEC. The inverse QCNN encodes a single logical qubit $|\psi_l\rangle$ into 9 physical qubits, which undergo noise \mathcal{N} . QCNN then decodes these to obtain the logical state ρ . Our aim is to maximize $\langle\psi_l|\rho|\psi_l\rangle$. (b) Logical error rate of Shor code (blue) versus a learned QEC code (orange) in a correlated error model. The input error rate is defined as the sum of all probabilities p_μ and p_{xx} . The Shor code has worse performance than performing no error correction at all (identity, gray line), while the optimized code can still significantly reduce the error rate.

lution layers and one pooling layer at each depth, followed by a fully connected layer (see Methods). Initially, all unitaries are set to random values. Because classically simulating our training procedure requires expensive computational resources, we focus on a relatively small system with $N = 15$ spins and QCNN depth $d = 1$; there are a total of 1309 parameters to be learned (see Methods). Our training data consists of 40 evenly spaced points along the line $h_2 = 0$, where the Hamiltonian is exactly solvable by Jordan-Wigner transformation. Using gradient descent with the mean-squared error function (1), we iteratively update the unitaries until convergence (see Methods). The classification output of the resulting QCNN for generic h_2 is shown in Fig. 4. Remarkably, this QCNN accurately reproduces the 2D phase diagram over the entire parameter regime, even though the model was trained only on samples from a set of solvable points which does not even cross the lower phase boundary.

This example illustrates how the QCNN structure avoids overfitting to training data with its exponentially reduced number of parameters. While the training dataset for this particular QPR problem consists of solvable points, more generally, such a dataset can be obtained by using traditional methods (e.g. measuring SOPs) to classify representative states that can be efficiently generated either numerically or experimentally.^{36,37}

OPTIMIZING QUANTUM ERROR CORRECTION

As seen in the previous example, the QCNN’s architecture enables one to perform effective QEC. We next leverage this feature to design a new QEC code itself that is optimized for a given error model. More specifically, any QCNN circuit (and its inverse) can be viewed as a decoding (encoding) quantum channel between the physical input qubits and the logical output qubit. The encoding scheme introduces sets of new qubits in a predetermined state, e.g. $|0\rangle$, while the decoding scheme performs measurements (Fig. 5a). Given an error channel \mathcal{N} , our aim is therefore to maximize the recovery fidelity

$$f_q = \sum_{|\psi_l\rangle \in \{|\pm x, y, z\rangle\}} \langle\psi_l|\mathcal{M}_q^{-1}(\mathcal{N}(\mathcal{M}_q(|\psi_l\rangle\langle\psi_l|)))|\psi_l\rangle, \quad (5)$$

where $\mathcal{M}_q(\mathcal{M}_q^{-1})$ is the encoding (decoding) scheme generated by a QCNN circuit, and $|\pm x, y, z\rangle$ are the ± 1 eigenstates of the Pauli matrices. Thus, our method simultaneously optimizes both encoding and decoding schemes, while ensuring their efficient implementation in realistic systems. Importantly, the variational optimization can be carried out with an unknown \mathcal{N} since f_q can be evaluated experimentally.

To illustrate the potential of this procedure, we consider a two-layer QCNN with $N = 9$ physical qubits and 126 variational parameters (Figure 5a and Methods). This particular architecture includes the nested (classical) repetition codes and the 9-qubit Shor code³⁸; in the following, we compare our performance to the better of the two. We consider three different input error models: (1) independent single-qubit errors on all qubits with equal probabilities p_μ for $\mu = X, Y$, and Z errors or (2) anisotropic probabilities $p_x \neq p_y = p_z$, and (3) independent single-qubit anisotropic errors with additional two-qubit correlated errors $X_i X_{i+1}$ with probability p_{xx} .

Upon initializing all QCNN parameters to random values and numerically optimizing them to maximize f_q , we find that our model produces the same logical error rate as known codes in case (1), but can reduce the error rate by a constant factor in case (2), depending on the specific input error probability ratios (e.g. 14% for $p_x = 1.8p_y$, or 50% for $p_x = 0.4p_y$ —see Methods). More drastically, in case (3), the optimized QEC code performs significantly better than known codes (Figure 5b). Specifically, because the Shor code is only guaranteed to correct arbitrary single-qubit errors, it performs even worse than using no error correction, while the optimized QEC code performs much better. This example demonstrates the power of using QCNNs to obtain and optimize new QEC codes for realistic, a priori unknown error models.

EXPERIMENTAL REALIZATIONS

Our QCNN architecture can be efficiently implemented on several state-of-the-art experimental platforms. The key ingredients for realizing QCNNs include the efficient preparation of quantum many-body input states, the application of two-qubit gates at various length scales, and projective measurements³⁹. These capabilities have already been demonstrated in multiple programmable quantum simulators consisting of $N \geq 50$ qubits based on trapped neutral atoms and ions, or superconducting qubits^{40–43}.

As an example, we present a feasible protocol for near-term implementation of our exact cluster model QCNN circuit via neutral Rydberg atoms^{40,44}, where long-range dipolar interactions allow high fidelity entangling gates⁴⁵ among distant qubits in a variable geometric arrangement. The qubits can be encoded in the hyperfine ground states, where one of the states can be coupled to the Rydberg level to perform efficient entangling operations via the Rydberg-blockade mechanism⁴⁵; an explicit implementation scheme for every gate in Fig. 2b is provided in Methods. Our QCNN at depth d with N input qubits requires at most $\frac{7N}{2}(1-3^{1-d}) + N3^{1-d}$ multi-qubit operations and $4d$ single-qubit rotations. For a realistic effective coupling strength $\Omega \sim 2\pi \times 10 - 100$ MHz and single-qubit coherence time $\tau \sim 200$ μ s limited by the Rydberg state lifetime, approximately $\Omega\tau \sim 2\pi \times 10^3 - 10^4$ multi-qubit operations can be performed, and a $d = 4$ QCNN on $N \sim 100$ qubits feasible. These estimates are reasonably conservative as we have not considered advanced control techniques such as pulse-shaping⁴⁶, or potentially parallelizing independent multi-qubit operations.

OUTLOOK

These considerations indicate that QCNNs provide a promising quantum machine learning paradigm. Sev-

eral interesting generalizations and future directions can be considered. First, while we have only presented the QCNN circuit structure for recognizing 1D phases, it is straightforward to generalize the model to higher dimensions, where phases with intrinsic topological order such as the toric code are supported^{47,48}. Such studies could potentially identify nonlocal order parameters with low sample complexity for lesser-understood phases such as quantum spin liquids⁴⁹ or anyonic chains⁵⁰. To recognize more exotic phases, we could also relax the translation-invariance constraints, resulting in $O(N)$ parameters for system size N , or use ancilla qubits to implement parallel feature maps following traditional CNN architecture. Further extensions can incorporate optimizations for fault-tolerant operations on QEC code spaces. Finally, while we have used a finite-difference scheme to compute gradients in our learning demonstrations, the structural similarity of QCNN with its classical counterpart motivates adoption of more efficient schemes such as backpropagation¹.

Acknowledgment. The authors thank Xiao-Gang Wen, Ignacio Cirac, Xiaoliang Qi, Edward Farhi, John Preskill, Wen Wei Ho, Hannes Pichler, Ashvin Vishwanath, Chetan Nayak, and Zhenghan Wang for insightful discussions. I.C. acknowledges support from the Paul and Daisy Soros Fellowship, the Fannie and John Hertz Foundation, and the Department of Defense through the National Defense Science and Engineering Graduate Fellowship Program. S.C. acknowledges support from the Miller Institute for Basic Research in Science. This work was supported through the National Science Foundation (NSF), the Center for Ultracold Atoms, the Vannevar Bush Faculty Fellowship, and Google Research Award.

-
- * Electronic address: soonwon@berkeley.edu
- ¹ Y. LeCun, Y. Bengio, and G. Hinton, *Nature* **521**, 436 (2015).
 - ² G. Carleo and M. Troyer, *Science* **355**, 602 (2017).
 - ³ E. P. L. van Nieuwenburg, Y. h. Liu, and S. D. Huber, *Nat. Phys.* **13**, 435 (2017).
 - ⁴ J. Carrasquilla and R. G. Melko, *Nat. Phys.* **13** (2017).
 - ⁵ L. Wang, *Phys. Rev. B* **94** (2016).
 - ⁶ Y. Levine, N. Cohen, and A. Shashua, *Phys. Rev. Lett.* **122** (2019).
 - ⁷ Y. Zhang and E.-A. Kim, *Phys. Rev. Lett.* **118** (2017).
 - ⁸ H. W. Lin, M. Tegmark, and D. Rolnick, *J. Stat. Phys.* **168**, 1223 (2017).
 - ⁹ P. Mehta and D. J. Schwab, preprint, arXiv (2014), 1410.3831.
 - ¹⁰ J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, *Nature* **549**, 195 (2017).
 - ¹¹ V. Dunjko, J. M. Taylor, and H. J. Briegel, *Phys. Rev. Lett.* **117** (2016).
 - ¹² E. Farhi and H. Neven, preprint, arXiv (2018), 1802.06002.
 - ¹³ W. Huggins, P. Patil, B. Mitchell, K. B. Whaley, and E. M. Stoudenmire, *Quantum Science and Technology* **4** (2018).
 - ¹⁴ T. D. Ladd et al., *Nature* **464**, 45 (2010).
 - ¹⁵ C. Monroe and J. Kim, *Science* **339**, 1164 (2013).
 - ¹⁶ M. H. Devoret and R. J. Schoelkopf, *Science* **339**, 1169 (2013).
 - ¹⁷ D. D. Awschalom, L. C. Bassett, A. S. Dzurak, E. L. Hu, and J. R. Petta, *Science* **339**, 1174 (2013).
 - ¹⁸ C.-Y. Huang, X. Chen, and F.-L. Lin, *Phys. Rev. B* **88** (2013).
 - ¹⁹ S. Singh and G. Vidal, *Phys. Rev. B* **88** (2013).
 - ²⁰ I. Kim and B. Swingle, preprint, arXiv (2017), 1711.07500.

- ²¹ J. Haah, A. W. Harrow, Z. Ji, X. Wu, and N. Yu, IEEE Transactions on Information Theory **63**, 5628 (2017).
- ²² Y. LeCun and Y. Bengio, The handbook of brain theory and neural networks **3361**, 10 (1995).
- ²³ A. Krizhevsky, I. Sutskever, and G. E. Hinton, Advances in Neural Information Processing Systems (2012).
- ²⁴ More generally, CNN layers connect *volumes* of multiple feature maps to subsequent volumes; for simplicity, we consider only a single feature map per volume and leave the generalization to future works.
- ²⁵ G. Vidal, Phys. Rev. Lett. **101** (2008).
- ²⁶ M. Aguado and G. Vidal, Phys. Rev. Lett. **100** (2008).
- ²⁷ R. N. C. Pfeifer, G. Evenbly, and G. Vidal, Phys. Rev. A **79** (2009).
- ²⁸ J. Preskill (1998), lecture Notes for Physics 229, California Institute of Technology.
- ²⁹ S. Sachdev, *Quantum Phase Transitions* (Cambridge University Press, 2011).
- ³⁰ F. Haldane, Phys. Rev. Lett. **50** (1983).
- ³¹ J. Haegeman, D. Pérez-García, I. Cirac, and N. Schuch, Phys. Rev. Lett. **109** (2012).
- ³² F. Pollmann and A. M. Turner, Phys. Rev. B **86** (2012).
- ³³ L. D. Brown, T. T. Can, and A. DasGupta, Statistical Science **16** (2001).
- ³⁴ B. Zeng and D. L. Zhou, Eur. Phys. Lett. **113** (2016).
- ³⁵ We included some training data points outside the plotted 2D parameter regime (namely those with $h_1 \in [1.6, 2]$) to obtain an equal number of points inside and outside the SPT phase, and avoid bias in the prior distribution.
- ³⁶ M. Schwarz, K. Temme, and F. Verstraete, Phys. Rev. Lett. **108** (2012).
- ³⁷ Y. Ge, A. Molnar, and J. I. Cirac, Phys. Rev. Lett. **116** (2016).
- ³⁸ P. Shor, Phys. Rev. A **52** (1995).
- ³⁹ We emphasize that the measurements of intermediate qubits and feed-forwarding can be replaced by controlled two-qubit unitary operations so that measurements are only performed at the end of an experimental sequence, as in stabilizer-based QEC.
- ⁴⁰ H. Bernien, S. Schwartz, A. Keesling, H. Levine, A. Omran, H. Pichler, S. Choi, A. S. Zibrov, M. Endres, M. Greiner, et al., Nature **551**, 579 (2017).
- ⁴¹ J. Zhang, G. Pagano, P. W. Hess, A. Kyprianidis, P. Becker, H. Kaplan, A. V. Gorshkov, Z.-X. Gong, and C. Monroe, Nature **551**, 601 (2017).
- ⁴² T. Brydges, A. Elben, P. Jurcevic, B. Vermersch, C. Maier, B. P. Lanyon, P. Zoller, R. Blatt, and C. F. Roos, preprint, arXiv (2018), 1806.05747.
- ⁴³ R. Harris et al., Science **361**, 162 (2018).
- ⁴⁴ H. Labuhn, D. Barredo, S. Ravets, S. de D. Léséleuc, M. Macri, T. Lahaye, and A. Browaeys, Nature **534**, 667 (2016).
- ⁴⁵ H. Levine, A. Keesling, A. Omran, H. Bernien, S. Schwartz, A. S. Zibrov, M. Endres, M. Greiner, V. Vuletic, and M. D. Lukin, Phys. Rev. Lett. **121** (2018).
- ⁴⁶ R. Freeman, Progress in Nuclear Magnetic Resonance Spectroscopy **32**, 59 (1998).
- ⁴⁷ A. Y. Kitaev, Ann. Phys. **303** (2003).
- ⁴⁸ M. A. Levin and X.-G. Wen, Phys. Rev. B. **71** (2005).
- ⁴⁹ L. Savary and L. Balents, Rep. Prog. Phys **80** (2017).
- ⁵⁰ A. Feiguin, S. Trebst, A. W. W. Ludwig, M. Troyer, A. Kitaev, Z. Wang, and M. H. Freedman, Phys. Rev. Lett. **98** (2007).

Methods

Phase Diagram and QCNN Circuit Simulations

The phase diagram in the main text (Fig. 2a) was numerically obtained using the infinite size density-matrix renormalization group (DMRG) algorithm. We generally follow the method outlined in Ref. 51 with the maximum bond dimension 150. To extract each data point in Fig. 2a, we numerically obtain the ground state energy density as a function of h_2 for fixed h_1 and computed its second order derivative. The phase boundary points are identified from sharp peaks.

The simulation of our QCNN in Fig. 2b also utilizes matrix product state representations. We first obtain the input ground state wavefunction using finite-size DMRG⁵¹ with bond dimension $D = 130$ for a system of $N = 135$ qubits. Then, the circuit operations are performed by sequentially applying SWAP and two-qubit gates on nearest neighboring qubits⁵². Each three-qubit gate is decomposed into two-qubit unitaries⁵³. We find that increasing bond dimension to $D = 150$ does not lead to any visible changes in our main figures, confirming a reasonable convergence of our method. The color plot in Fig. 2a is similarly generated for a system of $N = 45$ qubits.

QCNN for the $S = 1$ Haldane Chain

As discussed in the main text, the (spin-1/2) 1D cluster state belongs to an SPT phase protected by $\mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry, a phase which also contains the celebrated $S = 1$ Haldane chain³⁰. It is thus natural to ask whether this circuit can be used to detect the phase transition between the Haldane phase and an $S = 1$ paramagnetic phase, which we numerically demonstrate here.

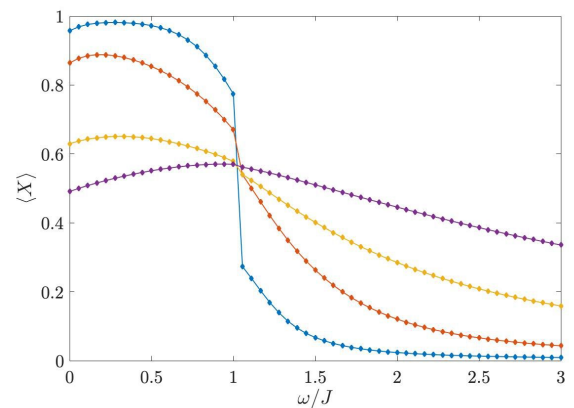


Figure 7: Exact QCNN output (at depth $d = 1, \dots, 4$) for the Haldane chain Hamiltonian with $N = 54$ spins.

The one-parameter family of Hamiltonians we con-

sider for the Haldane phase is defined on a one-dimensional chain of N spin-1 particles with open boundary conditions³⁰:

$$H_{\text{Haldane}} = J \sum_{j=1}^N \mathbf{S}_j \cdot \mathbf{S}_{j+1} + \omega \sum_{j=1}^N (S_j^z)^2 \quad (6)$$

In this equation, \mathbf{S}_j denotes the vector of $S = 1$ spin operators at site j . The system is protected by a $\mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry generated by global π -rotations of every spin around the X and Y axes: $R_x = \prod_j e^{i\pi S_j^x}$, $R_y = e^{i\pi S_j^y}$. When ω is zero or small compared to J , the ground state belongs to the SPT phase, but when ω/J is sufficiently large, the ground state becomes paramagnetic³⁰.

To apply our QCNN circuit to this Haldane phase, we must first identify a quasi-local isometric map U between the two models, because their representations of the symmetry group are distinct. More specifically, since the cluster model has a $\mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry generated by $X_{\text{even(odd)}} = \prod_{i \in \text{even(odd)}} X_i$, we require $UR_xU^\dagger = X_{\text{odd}}$ and $UR_yU^\dagger = X_{\text{even}}$. Such a map can be constructed following Ref. 54. Intuitively, it extends the local Hilbert space of a spin-1 particle by introducing a spin singlet state $|s\rangle$ and mapping it to a pair of spin-1/2 particles: $|x\rangle \mapsto |+-\rangle$, $|y\rangle \mapsto -|-+\rangle$, $|z\rangle \mapsto -i|--\rangle$, $|s\rangle \mapsto |++\rangle$. Here, $|\pm\rangle$ denote the ± 1 eigenstates of the (spin-1/2) Pauli matrix X . $|\mu\rangle$ denotes a spin-1 state defined by $R_\nu |\mu\rangle = (-1)^{\delta_{\mu,\nu}+1} |\mu\rangle$ ($\mu, \nu \in \{x, y, z\}$). The QCNN circuit for the Haldane chain thus consists of applying U followed by the circuit presented in the main text.

Figure 7 shows the QCNN output for an input system of $N = 54$ spin-1 particles at depths $d = 1, \dots, 4$, obtained using matrix product state simulations with bond dimension $D = 160$. For this system size, we numerically identified the critical point as $\omega/J = 1.035 \pm 0.005$, by using DMRG to obtain the second derivative of energy density as a function of ω and J . The QCNN provides accurate identification of the phase transition.

Multiscale String Order Parameters

We examine the final operator measured by our circuit that recognizes the SPT phase in the Heisenberg picture. Although a QCNN performs non-unitary measurements in the pooling layers, similar to QEC circuits²⁸, one can postpone all measurements to the end and replace pooling layers by unitary controlled gates acting on both measured and unmeasured qubits. In this way, the QCNN is equivalent to measuring a non-local observable

$$\mathcal{O} = (U_{\text{CP}}^{(d)} \dots U_{\text{CP}}^{(1)})^\dagger Z_{i-1} X_i Z_{i+1} (U_{\text{CP}}^{(d)} \dots U_{\text{CP}}^{(1)}) \quad (7)$$

where i is the index of the measured qubit in the final layer and $U_{\text{CP}}^{(l)}$ is the unitary corresponding to the convolution-pooling unit at depth l . A more explicit expression of \mathcal{O} can be obtained by commuting U_{CP} with

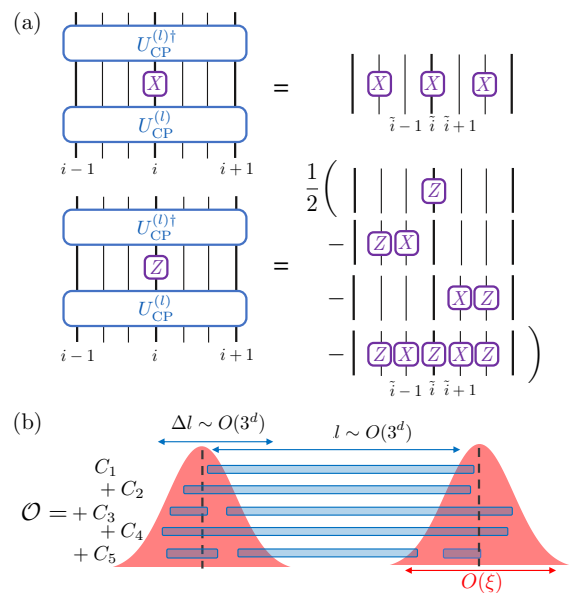


Figure 8: (a) Recursion relations for computing Pauli operators in the Heisenberg picture. $U_{\text{CP}}^{(l)}$ is the unitary corresponding to the convolution-pooling unit at depth l , and the different indices i, \tilde{i} reflect different numbers of unmeasured qubits at different layers. Qubits measured at depth l are indicated by lighter lines, while the remaining ones are shown in bold. (b) Measured operator in the Heisenberg picture is a sum of exponentially many products of string operators, with coefficients determined by Eq. (10).

the Pauli operators, which yields recursive relations:

$$U_{\text{CP}}^\dagger X_i U_{\text{CP}} = X_{\tilde{i}-2} X_{\tilde{i}} X_{\tilde{i}+2} \quad (8)$$

$$U_{\text{CP}}^\dagger Z_i U_{\text{CP}} = \frac{1}{2} (Z_{\tilde{i}} - Z_{\tilde{i}-2} X_{\tilde{i}-1} - X_{\tilde{i}+1} Z_{\tilde{i}+2} - Z_{\tilde{i}-2} X_{\tilde{i}-1} Z_{\tilde{i}} X_{\tilde{i}+1} Z_{\tilde{i}+2}) \quad (9)$$

\tilde{i} enumerates every qubit at depth $l-1$, including those measured in the pooling layer (Fig. 8a). It follows that an SOP of the form $ZX X \dots XZ$ at depth l transforms into a weighted linear combination of 16 products of SOPs at depth $l-1$. Thus, instead of measuring a single SOP, our QCNN circuit measures a sum of products of exponentially many different SOPs (Fig. 8b):

$$\mathcal{O} = \sum_{ab} C_{ab}^{(1)} \mathcal{S}_{ab} + \sum_{a_1 b_1 a_2 b_2} C_{a_1 b_1 a_2 b_2}^{(2)} \mathcal{S}_{a_1 b_1} \mathcal{S}_{a_2 b_2} + \dots, \quad (10)$$

\mathcal{O} can be viewed as a *multiscale* string order parameter with coefficients computed recursively in d using Eqs. (8,9). This allows the QCNN to produce a sharp classification output even when the correlation length is as long as 3^d .

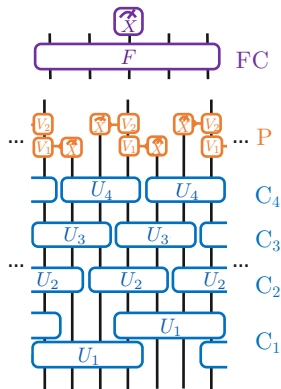


Figure 9: Circuit parameterization for training a QCNN to solve QPR. Our circuit involves 4 different convolution layers ($C_1 - C_4$), a pooling layer, and a fully connected layer. The unitaries are initialized to random values, and learned via gradient descent.

Demonstration of Learning Procedure for QPR

To perform our learning procedure in a QPR problem, we choose the hyperparameters for the QCNN as shown in Fig. 9. This hyperparameter structure can be used for generic (1D) phases, and is characterized by a single integer n that determines the reduction of system size in each convolution-pooling layer, $L \rightarrow L/n$. (Fig. 9 shows the special case where $n = 3$). The first convolution layer involves $(n+1)$ -qubit unitaries starting on every n^{th} qubit. This is followed by n layers of n -qubit unitaries arranged as shown in Fig. 9. The pooling layer measures $n - 1$ out of every contiguous block of n qubits; each of these is associated with a unitary V_j applied to the remaining qubit, depending on the measurement outcome. This set of convolution and pooling layers is repeated d times, where d is the QCNN depth. Finally, the fully connected layer consists of an arbitrary unitary on the remaining N/n^d qubits, and the classification output is given by the measurement output of the middle qubit (or any fixed choice of one of them). For our example, we choose $n = 3$ because the Hamiltonian in Eq. (2) involves three-qubit terms.

In our simulations, we consider only $N = 15$ spins and depth $d = 1$, because simulating quantum circuits on classical computers requires a large amount of resources. We parameterize unitaries as exponentials of generalized $a \times a$ Gell-Mann matrices $\{\Lambda_i\}$, where $a = 2^w$ and w is the number of qubits involved in the unitary⁵⁵: $U = \exp\left(-i \sum_j c_j \Lambda_j\right)$.

This parameterization is used directly for the unitaries in the convolution layers $C_2 - C_4$, the pooling layer, and the fully connected layer. For the first convolution layer C_1 , we restrict the choice of U_1 to a product of six two-qubit unitaries between each possible pair of qubits: $U_1 = U_{(23)}U_{(24)}U_{(13)}U_{(14)}U_{(12)}U_{(34)}$, where $U_{(\alpha\beta)}$ is a two-qubit unitary acting on qubits indexed by α and β . Such a decomposition is useful when considering ex-

perimental implementation.

In the QCNN learning procedure, all parameters c_μ are set to random values between 0 and 2π for the unitaries $\{U_i, V_j, F\}$. In every iteration of gradient descent, we compute the derivative of the mean-squared error function (Eq. (1) in the main text) to first order with respect to each of these coefficients c_μ by using the finite-difference method:

$$\frac{\partial \text{MSE}}{\partial c_\mu} = \frac{1}{2\epsilon} (\text{MSE}(c_\mu + \epsilon) - \text{MSE}(c_\mu - \epsilon)) + O(\epsilon^2). \quad (11)$$

Each coefficient is thus updated as $c_\mu \mapsto c_\mu - \eta \frac{\partial \text{MSE}}{\partial c_\mu}$, where η is the learning rate for that iteration. We compute the learning rate using the bold driver technique from machine learning, where η is increased by 5% if the error has decreased from the previous iteration, and decreased by 50% otherwise⁵⁶. We repeat the gradient descent procedure until the error function changes on the order of 10^{-5} between successive iterations. In our simulations, we use $\epsilon = 10^{-4}$ for the gradient computation, and begin with an initial learning rate of $\eta_0 = 10$.

Construction of QCNN Circuit

To construct the exact QCNN circuit in Fig. 2b, we followed the guidelines discussed in the main text. Specifically, we designed the convolution and pooling layers to satisfy the following two important properties:

1. Fixed-point criterion: If the input is a cluster state $|\psi_0\rangle$ of L spins, the output of the convolution-pooling layers is a cluster state $|\psi_0\rangle$ of $L/3$ spins, with all measurements deterministically yielding $|0\rangle$.
2. QEC criterion: If the input is not $|\psi_0\rangle$ but instead differs from $|\psi_0\rangle$ at one site by an error which commutes with the global symmetry, the output should still be a cluster state of $L/3$ spins, but at least one of the measurements will result in the state $|1\rangle$.

These two properties are desirable for any quantum circuit implementation of RG flow for performing QPR.

In the specific case of our Hamiltonian, the ground state (1D cluster state) is a graph state, which can be efficiently obtained by applying a sequence of controlled phase gates to a product state. This significantly simplifies the construction of the MERA representation for the fixed-point criterion. To satisfy the QEC criterion, we treat the ground state manifold of the unperturbed Hamiltonian $H = -J \sum_i Z_i X_{i+1} Z_{i+2}$ as the code space of a stabilizer code with stabilizers $\{Z_i X_{i+1} Z_{i+2}\}$. The remaining degrees of freedom in the QCNN convolution and pooling layers are then specified such that the circuit detects and corrects the error (i.e. measures at least one $|1\rangle$ and prevents propagation to the next layer) when a single-qubit X error is present.

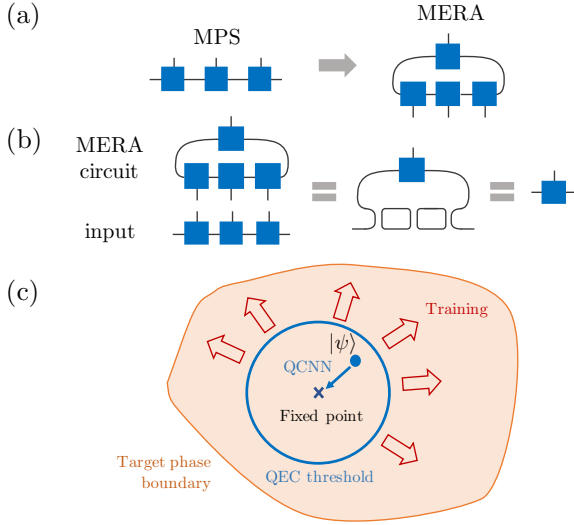


Figure 10: (a) Given a state with a translationally invariant, isometric matrix product state representation (e.g. a fixed point state for a 1D SPT phase), we explicitly construct an isometry for the MERA representation of this state. Blue squares are the matrix product state tensors, while black lines are the legs of the tensor. While we have illustrated a 3-to-1 isometry, the generalization to arbitrary n -to-1 isometries is straightforward. (b) Diagrammatic proof showing that a MERA constructed from the above tensor maps the fixed-point state back to a shorter version of itself. The first equality uses the definition of isometric tensor, and loops in the middle diagram simplify to a constant number unity. The generalization of this isometry to higher dimensions is discussed in Ref. 57. (c) One helpful initial parameterization for QPR problems consists of a MERA for the fixed point state $|\psi_0(\mathcal{P})\rangle$ and a choice of nested QEC, so that states within the QEC threshold flow toward $|\psi_0(\mathcal{P})\rangle$. Training procedures then expand this threshold boundary to the phase boundary.

QCNN for General QPR Problems

Our interpretation of QCNNs in terms of MERA and QEC motivates their application for recognizing more generic quantum phases. For any quantum phase \mathcal{P} whose RG fixed-point wavefunction $|\psi_0(\mathcal{P})\rangle$ has a tensor network representation in isometric or G -isometric form⁵⁸ (Fig. 10a), one can systematically construct a corresponding QCNN circuit. This family of quantum phases includes all 1D SPT and 2D string-net phases^{48,58,59}. In these cases, one can explicitly construct a commuting parent Hamiltonian for $|\psi_0(\mathcal{P})\rangle$ and a MERA structure in which $|\psi_0(\mathcal{P})\rangle$ is a fixed-point wavefunction (Fig. 10a for 1D systems). The diagrammatic proof of this fixed-point property is given in Fig. 10b. Furthermore, any “local error” perturbing an input state away from $|\psi_0(\mathcal{P})\rangle$ can be identified by measuring a fraction of terms in the parent Hamiltonian, similar to syndrome measurements in stabilizer-based QEC²⁸. Then, a QCNN for \mathcal{P} simply consists of the MERA for $|\psi_0(\mathcal{P})\rangle$ and a nested QEC scheme in which an input state with

error density below the QEC threshold⁶⁰ “flows” to the RG fixed point. Such a QCNN can be optimized via our learning procedure.

While our generic learning protocol begins with completely random unitaries, as in the classical case¹, this initialization may not be the most efficient for gradient descent. Instead, motivated by deep learning techniques such as pre-training¹, a better initial parameterization would consist of a MERA representation of $|\psi_0(\mathcal{P})\rangle$ and one choice of nested QEC. With such an initialization, the learning procedure serves to optimize the QEC scheme, expanding its threshold to the target phase boundary (Fig. 10c).

Experimental Resource Analysis

To compute the gate depth of the cluster model QCNN circuit in a Rydberg atom implementation, we analyze each gate shown in Figure 2b. By postponing pooling layer measurements to the end of the circuit, the multi-qubit gates required are

$$C_z Z_{ij} = e^{i\pi(-1+Z_i)(-1+Z_j)/4} \quad (12)$$

$$C_x Z_{ij} = e^{i\pi(-1+X_i)(-1+Z_j)/4} \quad (13)$$

$$C_x C_x X_{ijk} = e^{i\pi(-1+X_i)(-1+X_j)(-1+X_k)/8}. \quad (14)$$

By using Rydberg blockade-mediated controlled gates⁵⁷, it is straightforward to implement $C_z Z_{ij}$ and $C_z C_z Z_{ijk} = e^{i\pi(-1+Z_i)(-1+Z_j)(-1+Z_k)/8}$. The desired $C_x Z_{ij}$ and $C_x C_x X_{ijk}$ gates can then be obtained by conjugating $C_z Z_{ij}$ and $C_z C_z Z_{ijk}$ by single-qubit rotations. For input size of N spins, the k^{th} convolution-pooling unit thus applies $4N/3^{k-1}$ $C_z Z_{ij}$ gates, $N/3^{k-1}$ $C_x C_x X_{ijk}$ gates, and $2N/3^{k-1}$ layers of $C_x Z_{ij}$ gates. The depth of single-qubit rotations required is $4d$, as these rotations can be implemented in parallel on all N qubits. Finally, the fully connected layer consists of $N3^{1-d}$ $C_z Z_{ij}$ gates. Thus, the total number of multi-qubit operations required for a QCNN of depth d operating on N spins is $\frac{7N}{2}(1-3^{1-d}) + N3^{1-d}$. Note that we need not use SWAP gates since the Rydberg interaction is long-range.

Demonstration of Learning Procedure for QEC

To obtain the QEC code considered in the main text, we consider a QCNN with $N = 9$ input physical qubits and simulate the circuit evolution of its $2^N \times 2^N$ density matrix exactly. Strictly speaking, our QCNN has three layers: a three-qubit convolution layer U_1 , a 3-to-1 pooling layer, and a 3-to-1 fully connected layer U_2 . Without loss of generality, we may ignore the optimization over the pooling layer by absorbing its effect into the first convolution layer, leading to the effective two-layer

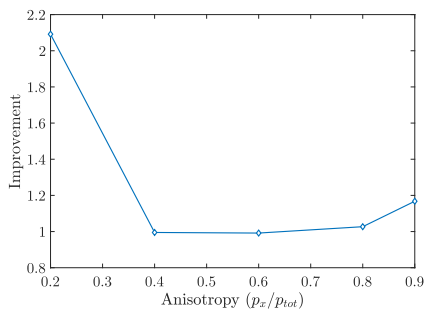


Figure 11: Ratio between the logical error rate of the Shor code and that of the QCNN code for the anisotropic depolarization error model. We fix the total input error rate $p_{\text{tot}} = p_x + p_y + p_z = 0.001$ and $p_y = p_z$, while varying the ratio p_x/p_{tot} .

structure shown in Fig. 5a. The generic three-qubit unitary operations U_1 and U_2 are parameterized using 63 Gell-Mann coefficients each.

As discussed in the main text, we consider three different error models: (1) independent single-qubit errors on all qubits with equal probabilities p_μ for $\mu = X, Y$, and Z errors, (2) independent single-qubit errors on all qubits, with anisotropic probabilities $p_x \neq p_y = p_z$, and (3) independent single-qubit anisotropic errors with additional two-qubit correlated errors $X_i X_{i+1}$ with probability p_{xx} . More specifically, the first two error models are realized by applying a (generally anisotropic) depolarization quantum channel to each of the nine physical qubits:

$$\mathcal{N}_{1,i} : \rho \mapsto (1 - \sum_{\mu} p_{\mu})\rho + \sum_{\mu} p_{\mu} \sigma_i^{\mu} \rho \sigma_i^{\mu} \quad (15)$$

with Pauli matrices σ_i^{μ} for $i \in \{1, 2, \dots, 9\}$ (the qubit indices are defined from bottom to top in Fig. 5a). For the anisotropic case, we trained the QCNN on various different error models with the same total error probability $p_x + p_y + p_z = 0.001$, but different relative ratios; the resulting ratio between the logical error probability of the Shor code and that of the QCNN code is plotted as a function of anisotropy in Fig. 11. For strongly anisotropic models, the QCNN outperforms the Shor code, while for nearly isotropic models, the Shor code is optimal and QCNN can achieve the same logical error rate.

For the correlated error model, we additionally apply a quantum channel:

$$\mathcal{N}_{2,i} : \rho \mapsto (1 - p_{xx})\rho + p_{xx} X_i X_{i+1} \rho X_i X_{i+1} \quad (16)$$

for pairs of nearby qubits, i.e. $i \in \{1, 2, 4, 5, 7, 8\}$. Such a geometrically local correlation is motivated from experimental considerations. In this case, we train our QCNN circuit on a specific error model with parameter choices

$p_x = 5.8 \times 10^{-3}$, $p_y = p_z = 2 \times 10^{-3}$, $p_{xx} = 2 \times 10^{-4}$ and evaluate the logical error probabilities for various physical error models with the same relative ratios, but different total error per qubit $p_x + p_y + p_z + p_{xx}$. In general, for an anisotropic logical error model with probabilities p_μ for σ_μ logical errors, the overlap f_q is $(1 - 2 \sum_{\mu} p_{\mu}/3)$, since $\langle \pm\nu | \sigma_{\mu} | \pm\nu \rangle = (-1)^{\delta_{\mu,\nu}+1}$. Because of this, we compute the total logical error probability from f_q as $1.5(1 - f_q)$. Hence, our goal is to maximize the logical state overlap f_q defined in Eq. (5). If we naively apply the gradient descent method based on f_q directly to both U_1 and U_2 , we find that the optimization is easily trapped in a local optimum. Instead, we optimize two unitaries U_1 and U_2 sequentially, similar to the layer-by-layer optimization in backpropagation for conventional CNN¹.

A few remarks are in order. First, since U_1 is optimized prior to U_2 , one needs to devise an efficient cost function C_1 that is independent of U_2 . In particular, simply maximizing f_q with an assumption $U_2 = \mathbf{1}$ may not be ideal, since such choice does not capture a potential interplay between U_1 and U_2 . Second, because U_1 captures arbitrary single qubit rotations, the definition of C_1 should be basis independent. Finally, we note that the tree structure of our circuit allows one to view the first layer as an independent quantum channel:

$$\mathcal{M}_{U_1} : \rho \mapsto \text{tr}_a[U_1 \mathcal{N}(U_1^\dagger(|0\rangle\langle 0| \otimes \rho \otimes |0\rangle\langle 0|)U_1)U_1^\dagger], \quad (17)$$

where $\text{tr}_a[\cdot]$ denotes tracing over the ancilla qubits that are measured in the intermediate step. From this perspective, \mathcal{M}_{U_1} describes an effective error model to be corrected by the second layer.

With these considerations, we optimize U_1 such that the effective error model \mathcal{M}_{U_1} becomes as classical as possible, i.e. \mathcal{M}_{U_1} is dominated by a “flip” error along a certain axis with a strongly suppressed “phase” error. Only then, the remnant, simpler errors will be corrected by the second layer. More specifically, one may represent \mathcal{M}_{U_1} using a map $M_{U_1} : \mathbf{r} \mapsto M\mathbf{r} + \mathbf{c}$, where $\mathbf{r} \in \mathbb{R}^3$ is the Bloch vector for a qubit state $\rho \equiv \frac{1}{2}\mathbf{1} + \mathbf{r} \cdot \boldsymbol{\sigma}$ ⁵³. The singular values of the real matrix M encode the probabilities $p_1 \geq p_2 \geq p_3$ for three different types of errors. We choose our cost function for the first layer as $C_1 = p_1^2 + p_2 + p_3$, which is relatively more sensitive to p_2 and p_3 than p_1 and ensure that the resultant, optimized channel \mathcal{M}_{U_1} is dominated by one type of error (with probability p_1). We note that M can be efficiently evaluated from a quantum device without knowing \mathcal{N} , by performing quantum process tomography for a single logical qubit. Once U_1 is optimized, we use gradient decent to find an optimal U_2 to maximize the fidelity f_q . As with QPR, gradients are computed via the finite-difference method, and the learning rate is determined by the bold driver technique¹.

-
- * Electronic address: soonwon@berkeley.edu
- ⁵¹ I.P. McCulloch. *Infinite size density matrix renormalization group, revisited*. arXiv preprint arXiv:0804.2509 (2008).
- ⁵² G. Vidal. *Efficient Classical Simulation of Slightly Entangled Quantum Computations*. Phys. Rev. Lett. **91**(14), 147902 (2003).
- ⁵³ M.A. Nielsen and I. Chuang. *Quantum computation and quantum information*. Cambridge University Press (2000).
- ⁵⁴ R. Verresen, R. Moessner, and F. Pollman. *One-dimensional symmetry-protected topological phases and their transitions*. Phys. Rev. B **96**, 165124 (2017).
- ⁵⁵ R.A. Bertlmann and P. Krammer. *Bloch vectors for qudits*. J. Phys. A.: Math. Theor. **41** 235303 (2008).
- ⁵⁶ G. Hinton. *Lecture Notes for CSC2515: Introduction to Machine Learning*. University of Toronto, 2007.
- ⁵⁷ N. Schuch, D. Pérez-García, and J.I. Cirac. *PEPS as ground states: Degeneracy and topology*. Ann. Phys. **325**, 2153 (2010).
- ⁵⁸ N. Schuch, D. Pérez-García, and J.I. Cirac. *Classifying quantum phases using matrix product states and projected entangled pair states*. Phys. Rev. B **84**, 165139 (2011).
- ⁵⁹ X. Chen, Z.-C. Gu, and X.-G. Wen. *Classification of gapped symmetric phases in one-dimensional spin systems*, Phys. Rev. B **83**, 035107 (2011).
- ⁶⁰ D. Aharonov and M. Ben-Or, in *Proceedings of the twenty-ninth annual ACM symposium on the theory of computing* (ACM, 1997), pp. 176-188.
- ⁵⁷ M. Saffman, T. Walker, and K. Molmer, *Quantum information with Rydberg atoms*. Rev. Mod. Phys. **82**, 2313 (2010).