Quantum Convolutional Neural Networks are Effectively Classically Simulable

Pablo Bermejo,[1,2,3] Paolo Braccia,[4] Manuel S. Rudolph,[5] Zoë Holmes,[5] Lukasz Cincio,[4] and M. Cerezo[1,*]

[1] *Information Sciences, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*
[2] *Donostia International Physics Center, Paseo Manuel de Lardizabal 4, San Sebastián E-20018, Spain*
[3] *Department of Applied Physics, Gipuzkoa School of Engineering, University of the Basque Country (UPV/EHU), Plaza Europa 1, San Sebastián 20018, Spain*
[4] *Theoretical Division, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*
[5] *Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne CH-1015, Switzerland*

(Received 22 October 2024; accepted 19 February 2026; published 6 April 2026)

Quantum convolutional neural networks (QCNNs) are widely regarded as a promising model for quantum machine learning (QML). In this work, we analyze the most widely used variants of these models (i.e., tracing out- and measurement-based QCNNs), and we relate their heuristic success to two facts. First, that when randomly initialized, they can only operate on the information encoded in low-bodyness measurements of their input states. And second, that they are commonly benchmarked on "locally easy" datasets whose states are precisely classifiable by the information encoded in these low-bodyness observables subspace. From these insights, we argue that the QCNN's action on this subspace should be efficiently classically simulable. Indeed, we construct and train a purely classical QCNN surrogate—based on low-bodyness Pauli propagation, tensor networks, and classical shadows—that matches or outperforms standard QCNNs on all benchmark datasets and on up-to 1024 qubits, thereby empirically realizing our simulability claims. Our results can then be understood as highlighting a deeper symptom of QML: Models could only be showing heuristic success because they are benchmarked on simple problems, for which their action can be classically simulated. This insight points to the fact that nontrivial datasets are a truly necessary ingredient for moving forward with QML. To finish, we discuss how our results can be extrapolated to classically simulate other architectures.

DOI: 10.1103/8qt9-72ts

I. INTRODUCTION

Quantum machine learning [1] (QML) harnessed significant attention by riding on the combined heuristic success of classical machine learning [2–5] and the computational promises of quantum computing [6–11]. In the last few years, however, it has been recognized that QML models are prone to encountering serious trainability barriers—such as barren plateaus and local minima—that prevent their large-scale implementations [12–25]. While several methods have been proposed to mitigate these issues [14,17,23,26–48], the recent work of Ref. [49] presents the intriguing connection between absence of barren plateaus and classical simulability. In particular, Ref. [49] hints at the fact that the very features that lead to provable absence of exponentially vanishing gradients, also serve as blueprints for classical algorithms that can simulate the action of the model, provided that one is given access to an initial data acquisition phase on a quantum computer.

Here it is important to highlight that the absence of barren plateaus is generally proved as an average statement (i.e., one shows that *most* of the landscape does not exhibit a barren plateau). Correspondingly, certain barren plateau-free models are simulable on average (i.e., we can classically estimate the loss function for *most* randomly sampled parameter settings). However, one could argue that the ability to classically simulate the model at random points of the landscape is not practically useful, as the sections of the landscape that encode the solution to a problem are atypical. The question thus becomes: "*Can we not only simulate randomly sampled points of a barren plateau-free model, but also replicate a successful training over a simulated landscape?*" Proving such a strong result becomes

*Contact author: cerezo@lanl.gov

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

a much harder task, as one has to approach the problem in a model- and even task-dependent way.

In this work, we analyze the previous question by focusing on quantum convolutional neural networks (QCNNs) [50]. QCNNs, either based on pooling layers where qubits are traced-out or measured, have shown heuristic success for supervised classification tasks based on classical [51–73] and quantum data [43,50,74–83], and currently regarded as some of the promising QML architectures [84–87]. Indeed, these models have been shown to be barren plateaus-free in Ref. [26] for low entangled datasets. However, while Ref. [49] gives some intuition as to why QCNNs should be classically simulable *on average* (with an explicit algorithm recently presented in our companion manuscript of Refs. [88,89]), there is no rigorous and systematic study of their simulability across *all* the relevant parts of the landscape which are accessed during training.

Here we argue that the action of both tracing out- and measurement-based QCNNs is simulable, not just on average but in the stronger sense of enabling successful training, by classical algorithms enhanced with classical shadows [49]. Indeed, we do not merely conjecture that such a classical surrogate could work—we explicitly construct it, train it on standard classical and quantum datasets (up to 1024 qubits), and observe comparable or improved classification accuracies compared to standard QCNNs. Importantly, the fact that such classical simulation requires classical shadows means that a quantum computer might still be needed (to perform measurements of the input data), but our construction fully avoids the need to train an actual QCNN architecture on a quantum computer with a hybrid quantum-classical loop. This substantially reduces both the depths of circuits and number of circuits that need to be run on the quantum device. In fact, a universal quantum computer may no longer even be needed if the quantum data can be obtained from an analog quantum simulator or a more conventional quantum experiment. Importantly, we will see that our results also suggest that likely no quantum computer is needed at all in the case of classical data.

To reach this conclusion, we will show the following results (schematically shown in Fig. 1). First, that randomly initialized tracing out- and measurement-based QCNNs avoid barren plateaus by only extracting and processing the information encoded in low-bodyness (i.e., low-weight) observables of their input states. Second, we will show that the datasets (classical and quantum) used in the literature to demonstrate the power of QCNNs are "*locally easy*." That is, that they can be classified by a QCNN that is randomly initialized and trained on the information encoded in this polynomially sized low-bodyness operator subspace. Then, we show that the action of the QCNN can be replaced by a classically surrogate via a new variant of the low-weight efficient simulation algorithm (LOWESA) algorithm [90,91] or by tensor networks [92],

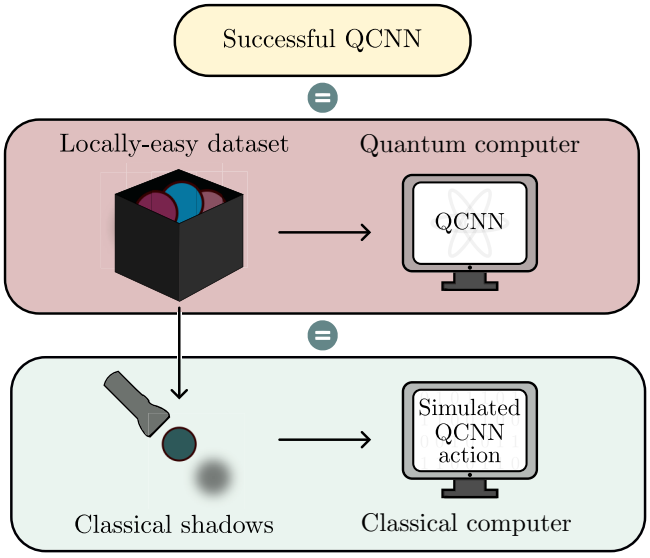


FIG. 1. Schematic representation of our main results. We conceptualize the success of quantum convolutional neural network as a consequence of two facts: (1) When randomly initialized, they operate on a polynomially sized subspace of low-bodyness observables. (2) They are benchmarked on locally easy datasets that are classifiable via the information encoded in low-bodyness measurements. The combination of these two facts allows us to show that there exist efficient classical algorithms that can simulate the action of the quantum convolutional neural network in this small subspace, provided that we are given access to Pauli classical shadows on the input data.

provided that we are given access to local Pauli classical shadows on the dataset's states [93–95]. Crucially, these insights for quantum data also point to the ever-increasing realization that classical algorithms equipped with measurements from quantum computers appear to be an extremely promising path forward for quantum machine learning [49,96,97].

While our results showcase that the heuristic success of QCNNs appears to be tied to locally easy datasets, this does not mean that examples could exist where the QCNN needs to leave the classically simulable regime to solve a given task. In fact, one can readily construct non-classically simulable datasets and QCNNs by embedding Shor's algorithm or some cryptographic assumption into the problem (see for instance Refs. [25,49,98]), but these are usually mathematical artifacts that are nowhere near related to realistic practical problems. Thus, the field of QML is in dire need of interesting, nontrivial datasets, in which models such as QCNNs can work without simultaneously being classically simulable. We present this as a challenge to the community and argue that until the viability of tracing out- and measurement-based QCNNs for such datasets can be demonstrated there is no reason to believe QCNNs will be useful.

II. FRAMEWORK

In this section we briefly introduce the considered supervised classification QML setting, as well as recall the basic ingredients of a QCNN, as defined in Ref. [50]. Then, we will present different notions of classical simulability, and discuss which one will be the focus of this work.

A. Supervised learning with QCNNs

To begin, let us recall that a supervised QML task is defined in terms of a data space $\mathcal{R}$, which belongs to the space of quantum states $\mathcal{R} \subseteq \mathcal{S}(\mathcal{H})$ of an n-qubit Hilbert space $\mathcal{H}$, an integer-valued label space $\mathcal{Y}$, and an unknown function $f : \mathcal{R} \mapsto \mathcal{Y}$ that assigns a label $y_i = f(\rho_i)$ to the states in $\mathcal{R}$. As such, our goal is to optimize a parameterized function $h_{\boldsymbol{\theta}} : \mathcal{R} \mapsto \mathcal{Y}$ to approximate the labels produced by f. The training of $h_{\boldsymbol{\theta}}$ is carried out by having repeated access to a dataset $T = \{(\rho_i, y_i)\}_i$, where ρ_i is drawn from $\mathcal{R}$ according to some probability distribution, and $y_i \in \mathcal{Y}$ are the associated labels. In this work we will consider two scenarios of interest corresponding to the classification of classical and quantum data [10]. For the case of classical data, we assume that the states in $\mathcal{R}$ were produced by encoding some real-valued classical vector $x_i \in \mathcal{X}$ into a quantum state by means of an encoding map $\mathcal{E} : \mathcal{X} \to \mathcal{R}$. Then, for quantum data, the states in $\mathcal{R}$ are produced by some quantum mechanical process of interest.

In this paper we will focus on QML schemes based on QCNN architectures [50]. As shown in Fig. 2(a), a QCNN is composed of a sequence of convolutional and pooling layers, with the first being used to coarse-grain the information of the input quantum states, and the latter aimed at reducing the QCNN's feature space by tracing out or measuring qubits. Following the definition in Ref. [50], we therefore can define a generic QCNN as the composition of quantum channels of the form

$$\Phi_{\boldsymbol{\theta},\boldsymbol{\lambda}} = \bigcirc_{l=1}^{L} \left(P_l^{\boldsymbol{\lambda}_l} \circ C_l^{\boldsymbol{\theta}_l} \right), \tag{1}$$

where C and P are parameterized by $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$, respectively, representing convolutional and pooling quantum maps. Convolutional maps $C_l^{\boldsymbol{\theta}_j} : \mathcal{S}(\mathcal{H}_l) \to \mathcal{S}(\mathcal{H}_l)$ preserve the size of the quantum registers they act on, while pooling maps reduce it, i.e., $P_l^{\boldsymbol{\lambda}_l} : \mathcal{S}(\mathcal{H}_l) \to \mathcal{S}(\mathcal{H}_{l+1})$ such that $\dim(\mathcal{H}_{l+1}) < \dim(\mathcal{H}_l)$. Typically, in a pooling layer half of the qubits are discarded or measured, meaning that the total number of layers satisfies $L \in \Theta(\log(n))$, i.e., for typical QCNN architectures the number of layers scales logarithmically with the system size.

While Eq. (1) allows for a wide variety of QCNNs, in this work we consider their most common instantiations: (1) tracing out-based QCNNs, where one uses unitary convolution maps and simply traces out of qubits (according to some pattern) in the pooling layers, and (2) measurement-based QCNNs, where qubits are measured in the pooling layers, and their results are used to control unitaries in neighboring sites. To begin, we will mostly focus on tracing out-based QCNNs. As we will see below in our numerical results section, this choice is motivated from the fact that these simpler architectures can be found to achieve similar, or even better, success than other more general ones (e.g., measurement-based QCNNs), indicating that additional levels of complexity might not be needed. Hence, in what follows we will assume that

$$\begin{aligned} C_l^{\boldsymbol{\theta}_l}(\cdot) &= U_l(\boldsymbol{\theta}_l)(\cdot)U_l^{\dagger}(\boldsymbol{\theta}_l), \\ P_l^{\boldsymbol{\lambda}_l}(\cdot) &= \mathrm{Tr}_{l_l}[\cdot], \end{aligned} \tag{2}$$

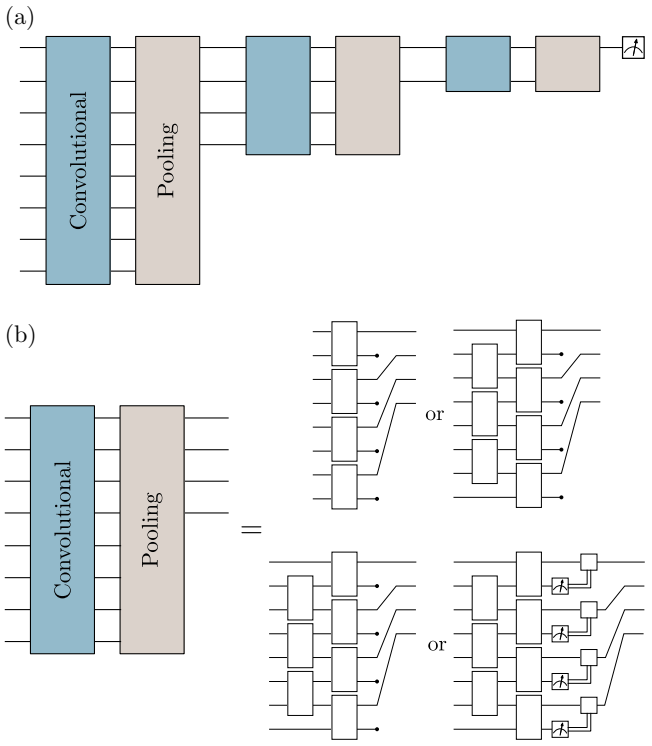


FIG. 2. Quantum convolutional neural network architecture. (a) A quantum convolutional neural networks is composed of alternating convolutional and pooling layers. In the convolutional layers, information is usually being processed by parametrized quantum gates. In the pooling layers, the dimension of the quantum convolutional neural network feature space is reduced by tracing out or measuring qubits. By design, quantum convolutional neural networks have a depth that only scales logarithmically with the number of qubits n, and the measurements at its output are local. (b) Examples of tracing out- and measurement-based quantum convolutional neural networks. Convolutional layers are composed of two-local gates acting on nearest neighbors. In a tracing out quantum convolutional neural network half of the qubits are being traced out during the pooling layer, whereas in a measurement-based quantum convolutional neural network, half of the qubits are measured and the outcomes control unitaries on one of their nearest neighbors.

where t_l determines the set of qubits being traced-out at the lth layer. Given that pooling layers are not parametrized we will denote the tracing out QCNN map as $\Phi_{\boldsymbol{\theta}}$.

At the output of this quantum neural network, we measure an observable O acting on $\mathcal{O}(1)$ qubits to compute the predicted label

$$\bar{y}_i(\boldsymbol{\theta}) = \mathrm{Tr}\left[\Phi_{\boldsymbol{\theta}}(\rho_i)O\right]. \tag{3}$$

These predictions are then used to evaluate a loss function $\mathcal{L}_{\boldsymbol{\theta}}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i)$ that quantifies how close the predicted labels are from the true ones on the training dataset T. For instance, one can use the mean-squared error or cross-entropy loss functions that are widely used in classical machine learning. Then, the QCNN's parameters are trained by leveraging the power of classical optimizers that solve the optimization task

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}\left(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i\right). \tag{4}$$

Once the optimal set of parameters $\boldsymbol{\theta}^*$ are found, these are used to define the trained model $h_{\boldsymbol{\theta}^*}$ [which is a function of the expectation values of Eq. (3)] to test its performance on new and previously unseen data points.

The success of the QML model in solving Eq. (4) hinges on several factors. For instance, one needs to efficiently navigate the (highly nonconvex) optimization landscape and reach its global minima. Here, it has been shown that several trainability barriers can exist in quantum neural network-based schemes such as the presence of barren plateaus (i.e., gradients that vanish exponentially with n) or suboptimal local minima where the optimizer can get trapped in Refs. [12–24]. Fortunately, it has been shown that QCNNs do not exhibit barren plateaus [26], and their heuristic success provides solid evidence that while local minima exist, they do not constitute a fundamental obstacle for QCNN trainability [43,50–72,74–82].

While these two features on their own have made QCNNs one of the most promising QML models, there is one aspect of their performance that has not been discussed enough: *Whether they can be classically simulated, or not [99].* Indeed, in a typical quantum neural network-based QML setting one assumes that the evaluation of the loss function $\mathcal{L}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i)$, or more precisely of the expectation values $\bar{y}_i(\boldsymbol{\theta})$, requires the use of a quantum computer, as it is presumed that the information processing capabilities of the quantum model map are hard to simulate. However, if a model is classically simulable, then the implementation of its parameterized quantum circuit on a quantum computer is unnecessary.

B. Notions of classical simulability

When studying the classical simulability of a QML model, one usually focuses on whether there exists a classical algorithm $\mathcal{A}$ that takes as input an efficient description of the problem (i.e., a gate sequence that prepares each ρ_i from a fiduciary state, a sequence of operations in the circuit, and a representation of the measurement operator) and returns an estimation $\hat{\mathcal{L}}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i)$ of the loss function [49,100–104]. In this framework, one generally focuses on the error $|\mathcal{L}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i) - \hat{\mathcal{L}}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i)|$ for a given set of parameters $\boldsymbol{\theta}$, or on the average error $\mathbb{E}_{\boldsymbol{\theta}}|\mathcal{L}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i) - \hat{\mathcal{L}}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i)|$ over the whole landscape. The results in our companion work of Ref. [88] can be used to provide bounds on the average error, showing that one can approximate the loss function to an arbitrarily small constant precision with polynomial time and sample complexity via low-bodyness (or low-Pauli-weight) approximations [102].

While the results in Ref. [88] show that random points in the QCNN's loss function landscape are easy to classically simulate with high probability, they do not imply that we can simulate the regions of the landscape that are relevant during training. To account for this limitation, we instead focus on a more pragmatic definition of classical simulability where the ultimate goal is to solve the task of interest [25]. In particular, we will follow the setting of "*classical simulation enhanced with quantum experiments*" (or more precisely, "*classical simulation enhanced with efficient shadow tomography*") defined in Ref. [49]. Here, one is allowed access to a quantum computer for an initial data acquisition phase during which one can prepare copies of the states in $\mathcal{R}$, apply some operations, and independently measure them via some efficient tomographic classical shadow techniques [93–95]. Once this initial data acquisition phase is over, one can no longer access the quantum computer. As such, we say that a model is classically simulable if there exists a classical algorithm $\mathcal{A}$ that takes as input an efficient description of the problem, as well as the data obtained from quantum devices in an initial data acquisition phase, and returns an estimation $\hat{\mathcal{L}}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i)$ such that the solution of

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \hat{\mathcal{L}}(\{\bar{y}_i(\boldsymbol{\theta})\}_i, \{y_i\}_i) \tag{5}$$

leads to a simulated trained model $\hat{h}_{\boldsymbol{\theta}^*_c}$ capable of achieving classification accuracies that are comparable to those of $h_{\boldsymbol{\theta}}$.

Note that this definition of classical simulability is intimately tied to the recently introduced notion of *dequantization in QML* (Definition 3 in Ref. [25]), with the addendum that we allow the classical algorithm access to measurements obtained from a quantum computer during an initial data acquisition phase. As such, our approach to classical simulability can be regarded as a substantial dequantization of QML whereby the usage of the quantum device is reduced to just collecting shadows of the training data. While this is not a full dequantization in the sense that

some form of quantum experiment may still be required, we stress that this experiment (state preparation followed by local Pauli measurements) is significantly simpler and so a full universal quantum computer may no longer be needed. As such, within the grander scheme of classical simulability of parametrized circuit-based QML models, we will say that if there is no need to run the parametrized quantum circuit on a quantum device, one can consider that the QML model's information processing abilities are dequantized.

III. CONCEPTUALIZING THE SUCCESS OF QCNNs

In this section we present the basic ingredients to understand what makes QCNNs successful. Then, we will argue that these same features can be exploited to classically simulate QCNN-based QML models via classical algorithms enhanced with classical shadows.

A. Randomly initialized QCNNs

To begin, let us begin by considering a unitary tracing out QCNN architecture as in Eq. (2), where the convolutional layers are composed of general parametrized two-qubit gates acting on nearest neighboring qubits in a pattern such as those of Fig. 2(b, top). Then, let $\Phi^{\dagger}_{\boldsymbol{\theta}}(O)$ denote the Heisenberg-evolved measurement operator, and we will expand this operator in the Pauli basis and define the bodyness of any given Pauli as the number of qubits it acts nontrivially on. In what follows, we will assume that when initializing the QCNN, we sample the parameters $\boldsymbol{\theta}$ such that each local gate forms an independent local 2-design over $\mathbb{U}(4)$ [105–110]. By means of the Weingarten calculus [111–113] we can find that the following result holds (see the Appendix for additional details):

Result 1 (Informal). Consider tracing out QCNNs where the convolutional layers are composed of general parametrized two-qubit gates acting on nearest neighboring qubits in a pattern such as those of Fig. 2(b, top, right). In average, the contribution in $\Phi^{\dagger}_{\boldsymbol{\theta}}(O)$ of a given Pauli with bodyness k, decays exponentially with k.

A direct implication of Result 1 is that when computing the inner product $\mathrm{Tr}[\rho\Phi^{\dagger}_{\boldsymbol{\theta}}(O)]$, the dominant contributions in the predicted labels of Eq. (3) will essentially only arise from $\mathcal{O}(1)$-bodyness measurements over the initial state. As such, we can see that, when randomly initialized, the QCNN will start its training by only processing the information encoded in the polynomially sized subspace of low-bodyness observables in their input data. In fact, these realizations can be directly linked to the proof of absence of barren plateaus in QCNNs, as a careful revision of Ref. [26] reveals that the large components in the loss function's gradients arise precisely from the local terms in the Heisenberg-evolved measurement operator $\Phi^{\dagger}_{\boldsymbol{\theta}}(O)$.

Importantly, we note that this conclusion is further strengthened by the results in our companion paper [88], where we show that, on average, the error in approximating the predicted labels via $\Phi^{\dagger}_{\boldsymbol{\theta}}(O)$ or via a low-k body approximation of this operator (where one truncates all the contributions from Paulis with bodyness higher than k after each layer, or after each gate), decays exponentially with k. Specifically, it decays as $\mathcal{O}((\frac{2}{3})^k)$. Indeed, while this observation is used in Ref. [88] to provide a classical algorithm that requires only polynomial time and sample complexity to simulate a QCNN on average, we note that the previous work does not perform end-to-end training on a simulated architecture.

At this point we find it important to highlight the fact while we have shown that the dominant terms of $\mathrm{Tr}[\rho\Phi^{\dagger}_{\boldsymbol{\theta}}(O)]$ arise from local observables over the initial state, these can be exponentially suppressed if the input data are too entangled (e.g., follows a volume law of entanglement [114,115]). In this case, the dominant information that the QCNN operates on might not be encoded in this small subspace, and the theoretical guarantees for the model are no longer valid. Namely, it follows from the results of Ref. [26] that QCNNs can exhibit barren plateaus when the data states are too entangled.

Then, we note that while Result 1 was theoretically derived for QCNNs where the convolutional layers are composed of general parametrized two-qubit gates acting on nearest neighboring qubits in a pattern such as those of Fig. 2(b), one can also generalize such result to other more architectures (e.g., where the convolutional layer has more gates) using the techniques in Ref. [116].

To finish, we note that the results in Result 1 are only an average statement, meaning that they only hold for randomly initialized QCNNs. Indeed, they do not provide a practical mean to simulate the action of the QCNN throughout training, when the parameters are not randomly sampled, but driven by an optimizer. Below we explicitly show how to construct a surrogate of the model and show that we can train on it to solve standard classification problems. Importantly, we note that our numerical simulations are a heuristic extension of the classical average-case simulability entailed by Result 1, and should not be considered a bridge between average-case and worst-case scenarios. Indeed, we note below that QCNN simulation will be successful only for specific datasets.

B. Locally easy datasets

In the previous sections we have argued that for random initializations the QCNN's training process is initially guided by the low-bodyness information over the input states, and that the loss function will exhibit large gradients if the initial states are not too entangled [26]. Hence,

we know that the QCNN will be able to—at least—take the first few training steps. From here, it is reasonable to expect that, if the landscape is sufficiently well-behaved (e.g., local minima do not prevent training) and a good solution lies also within this subspace, then the QCNN could accurately solve the task at hand. Evidently, analytically proving that all of these conditions are met could be an extremely hard task. However, we can define the following class of datasets for which QCNNs could perform well:

Definition 1 (Locally easy dataset). Given a dataset for a supervised QML classification task, we will say that it is locally easy if it can be classified using the expectation values of low-bodyness observables of the input quantum states.

Evidently, having a locally easy dataset is a necessary condition for the optimization curve to lie within the low-bodyness subspace, but it is nonetheless not a sufficient one. It is entirely possible that there exist initial points that are only connected to the global minima through paths that need to leave the subspace. However, as we will see below, we compiled classical and quantum datasets used in the literature to benchmark QCNNs, and we heuristically show not only that they are all locally easy, but the classification task can be solved by remaining in the low-bodyness subspace.

To finish, we find it important to note that in the appendixes we present results strengthening our claim that the considered datasets are locally easy. Namely, we consider the XXX bond-alternating dataset (see below for more details), generate $n = 100$ qubits states, estimate expectation values on all local Pauli operators—plus a few two-qubit ones—and perform perfect classification using a simple random forest algorithm. These results show that local observables contain all relevant information for classification, as per Definition 1.

C. Measurement-based QCNNs

To finish, let us now consider a measurement-based QCNN. For this purpose, we recall that the action of a local pooling operation on the lth layer, where qubit j is measured, and the outcome is used to control a trainable unitary $V(\boldsymbol{\lambda}, x)$ on qubit j' is given by

$$P_{l,j}^{\boldsymbol{\lambda}_l}(\cdot) = \sum_{x_j \in \{0,1\}} p(x_j)\, |x_j\rangle\langle x_j| \otimes V_{j'}(\boldsymbol{\lambda}, x)(\cdot)\, |x_j\rangle\langle x_j| \otimes V_{j'}^{\dagger}(\boldsymbol{\lambda}, x),$$

where $p(x) = \mathrm{Tr}[|x\rangle\langle x|\, \rho_{l,j}]$ and $\rho_{l,j}$ is the reduced state of the jth qubit at the lth layer. As such, the lth pooing layer takes the form

$$P_l^{\boldsymbol{\lambda}_l}(\cdot) = \prod_{j \in t_l} P_{l,j}^{\boldsymbol{\lambda}_l}(\cdot), \tag{6}$$

where we recall that t_l determines the set of qubits being measured at the lth layer.

By using again the Weingarten calculus we find that the following result holds (see the Appendix for a proof):

Result 2 (Informal). Consider measurement-based QCNNs where the convolutional layers are composed of general parametrized two-qubit gates acting on nearest neighboring qubits in a pattern such as those of Fig. 2(b, bottom, right). In average, the contribution in $\Phi^{\dagger}_{\boldsymbol{\theta},\boldsymbol{\lambda}}(O)$ of a given Pauli with bodyness k, decays exponentially with k.

Just as in tracing out-based QCNNs, Result 2 shows that randomly initialized measurement-based QCNN also start their training by only processing the information encoded in the polynomially sized subspace of low-bodyness observables in their input data. This realization is related to the absence of barren plateaus in this architecture. In fact, we can also show that (independently of the convolutional layer architecture)

$$\mathrm{Var}_{\boldsymbol{\theta}}\left[\mathrm{Tr}\left[\Phi_{\boldsymbol{\theta}}(\rho)O\right]\right] \leqslant \mathrm{Var}_{\boldsymbol{\theta},\boldsymbol{\lambda}}\left[\mathrm{Tr}\left[\Phi_{\boldsymbol{\theta},\boldsymbol{\lambda}}(\rho)O\right]\right], \tag{7}$$

where $\Phi_{\boldsymbol{\theta}}$ denotes the tracing out QCNN channel. Equation (7) indicates that the variance of measurement-based QCNNs is larger than that of tracing out ones, showing that adding measurements to the architecture decreases the model's expressive power [18,117].

IV. NUMERICAL RESULTS FOR CLASSICALLY SIMULATED QCNNs

As argued above, QCNNs simulability arises from the fact that they are initialized, explore and end their training in the polynomially large subspace of low-bodyness observables. We now use this fact to classically simulate QCNNs and show that its information processing capabilities can be emulated by classical computers for essentially all datasets used in the literature. As such, the purpose of this section is threefold:

(1) Proving that datasets commonly used in the literature as heuristic evidence for QCNNs are locally easy. In particular, we will see that simple Pauli classical shadows are sufficient to extract the relevant features of the data.
(2) Illustrating the classical simulation of QCNNs, as well as the scalability of the proposed techniques (we simulate and train QCNNs with upto 1024 qubits).

(3) Studying how the performance of the simulated QCNNs depends on the amount of measurements used during the shadow tomographic procedure.

The end-to-end simulations of tracing out-based QCNNs presented in this section are based on two techniques which are constructed to only process the information encoded in low-bodyness measurements of the input states. In particular, they accomplish this goal by truncating the Heisenberg-evolved measurement operators to bodyness $\mathcal{O}(1)$, which allows us to guarantee an efficient representation of this backward-evolved operator. As detailed in Appendix B, the first simulation method used to classify quantum data is based on the LOWESA algorithm [90,91]. Then, we classify classical data using a tensor network technique with restricted bodyness. In addition, in Appendix E, we show that tensor networks can also efficiently simulate measurement-based QCNNs for quantum data.

We will split our numerical results into quantum datasets and classical datasets, and we employ different simulation methods for each of these families. This will showcase the diversity of classical methods that enable the simulation of the QCNN action, as well as their broad applicability. In all cases, we find that the simulated QCNN reaches similar, or better performances than those reported in the literature for "full QCNNs" (i.e., not restricted to the subspace of low-bodyness observables). More importantly, we show that our finite sample shadow-based simulations can outperform full QCNNs trained with no finite sampling.

A. Quantum datasets

The problem of classifying quantum phases of matter is oftentimes considered as a key application of QML, since it inherently requires preparing quantum states of interest in a quantum device. However, despite the quantum nature of the data, we will argue that in the standard test cases used to demonstrate QCNNs the local information encoded in low-bodyness measurements suffices to classify the states. We establish this by showing that we can train a simulated QCNN [using the LOWESA-based Pauli propagation (PP) methods outlined in the Appendix] via Pauli classical shadows on the dataset states. Notably, the classical, weight-truncated QCNN is self-consistently trained to solve the classification tasks, not to faithfully emulate the training of an exact QCNN.

In particular, we focus on four popular classification tasks based on the one-dimensional Heisenberg bond-alternating XXX model [118], Haldane chain [119], axial next-nearest neighbor Ising (ANNNI) model [120], and Cluster Hamiltonian [121]. The datasets are composed of an evenly distributed number of ground states per phase [obtained via density-matrix renormalization group (DMRG) from ITensors.jl software library [122] for the Julia programming language [123]]. Then, the tomographic data are computed by performing standard Pauli classical shadows [93–95]. In all cases, we simulate the action of a QCNN as in Fig. 2(b, right), where each block is a general two-qubit unitary with 15 parameters, and where we trace out half of the remaining qubits in each pooling layer. The training of the simulated QCNN is performed by using a cross-entropy-based loss function and leveraging the LBFGS [124] optimizer for the parameter update.

Here we note that for the bond-alternating XXX model and the Haldane chain we will consider a binary classification task, and we will showcase the power of our simulations by considering spin systems of $n = 1024$ and $n = 512$ qubits, respectively. Then, for the ANNNI model and the Cluster Hamiltonian, we will focus on multiclass classification and restrict ourselves to QCNNs acting on $n = 32$ qubits, so that we can compare our results to those previously obtained in the literature.

1. Heisenberg bond-alternating XXX model

The Hamiltonian of the Heisenberg bond-alternating XXX model reads

$$H = \sum_{i=1}^{n-1} J_i \left(X_i X_{i+1} + Y_i Y_{i+1} + Z_i Z_{i+1}\right), \tag{8}$$

where X_i, Y_i and Z_i denote the Pauli operator acting on the ith qubit, and where $J_i = J_1(J_2) \geqslant 0$ for $i =$ even(odd). In the thermodynamic limit $n \to \infty$, the ground state of this model presents a phase transition controlled by the ratio J_2/J_1, such that $J_2 < J_1$ corresponds to a trivial phase whereas when $J_2 > J_1$ the ground space is topologically protected. See Fig. 3(a) for a sketch of the phase diagram.

Here we consider a chain of $n = 1024$ qubits. To our knowledge, this constitutes the largest QCNN implementation with classical shadows (see also Ref. [83] for an implementation with large-scale matrix-product states), and we find it important to highlight that the simulation was performed with resources available on a modern laptop. At the output of a QCNN we measure the Z_1 operator. Throughout the simulation we cap the maximum bodyness of the Heisenberg-evolved measurement operator at two. We use a dataset of 100 states and perform the training over 200 iterations, averaged over five different runs. To further strengthen our claim that just a small amount of information from the operator space is needed for the classification, we remark that only the first 400 operators with the largest variance across the shadows dataset were used to train this model (see the Appendix). The classification results are shown in Fig. 3(b), where we see that by taking only 100 classical shadows per data point one can consistently achieve test accuracies above 90%. This last fact is extremely important as the total number of measurements shots used to train our classical QCNN never

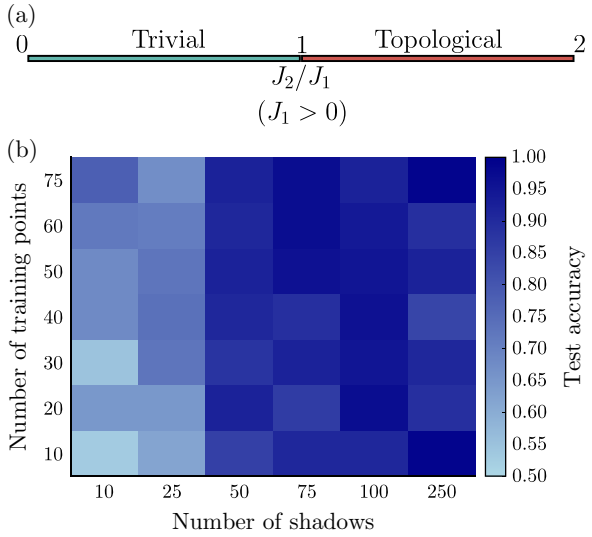


FIG. 3. Bond-alternating XXX model. (a) Phase diagram for the Hamiltonian in Eq. (8). (b) Test classification accuracy for the simulated quantum convolutional neural network acting only on the low-bodyness operator subspace. We show the accuracy as a function of the number of training points and Pauli classical shadows on each state of the dataset.

goes above 20 000 for the whole experiment. When compared against the resources needed to train a QCNN on a quantum computer, where at least 5000–10 000 shots are used to estimate the loss in each iteration step per data point, we can see that the amounts of quantum resources used in our scheme is order of magnitude smaller than standard QCNN training. Therefore, not only we can classically simulate the action of QCNNs, we can do so in an extremely measurement-frugal way.

2. Haldane chain

Next, we will consider the one-dimensional Haldane chain model [119] whose Hamiltonian reads

$$H = -J \sum_{i=1}^{n-2} Z_i X_{i+1} Z_{i+2} - h_1 \sum_{i=1}^{n} X_i - h_2 \sum_{i=1}^{n-1} X_i X_{i+1}. \quad (9)$$

Here, we take $J > 0$. As shown in Fig. 4 the model's phase diagram is characterized by the two ratios h_1/J and h_2/J. Inspecting the model in the thermodynamic limit, one can find out the emergence of three distinct phases: antiferromagnetic, paramagnetic and symmetry-protected (SPT). For the classification task, we will focus on classifying between the paramagnetic phase and the symmetry-protected phase, i.e., binary classification, leaving the multiclass classification for the ANNNI model and the cluster Hamiltonian (see below). As such, we fix $J = 1$ and $h_1 = 0.5$, and set the transition between phases to occur at $h_2 = 0.423$, given by the analysis in the thermodynamic limit [50].

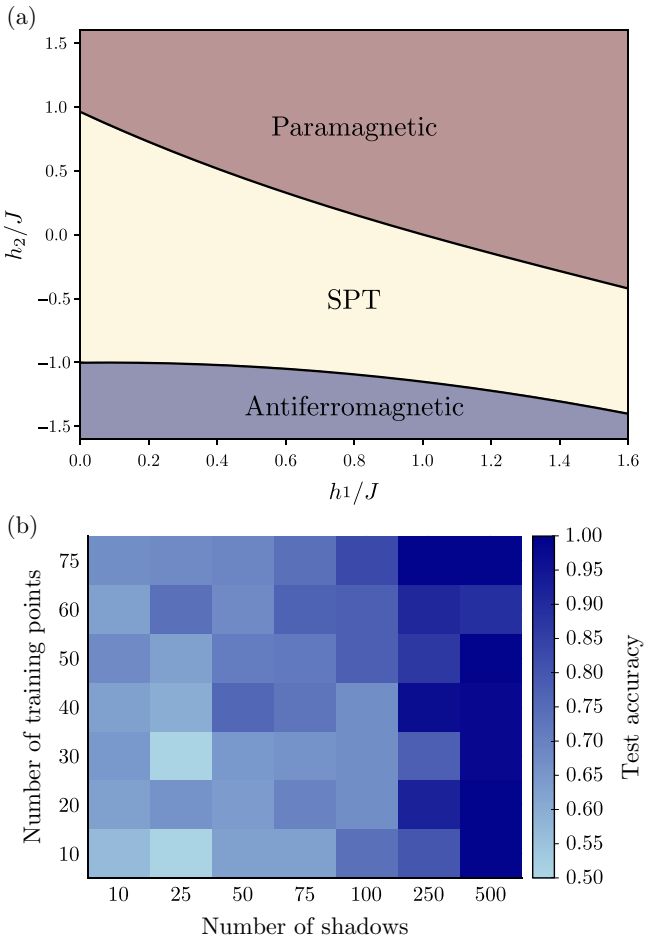


FIG. 4. Haldane chain. (a) Phase diagram for the Hamiltonian in Eq. (9). (b) Test classification accuracy for the simulated quantum convolutional neural network acting only on the low-bodyness operator subspace. We show the accuracy as a function of the number of training points and Pauli classical shadows on each state of the dataset.

Here we consider a chain of $n = 512$ qubits. Throughout the simulation we cap the maximum bodyness of the Heisenberg-evolved measurement operator at three, and we classify the phases only on the expectation values of the 3000 operators with the largest variance across the shadows dataset. We use again a dataset of 100 states and perform training over 200 iterations, averaged over five different runs. The classification results are shown in Fig. 4(b). Similar to the results obtained for the *XXX* model, we see that, as expected, using larger number of shadows leads to better classification accuracy. In this case, we see that the number of training points has a smaller effect on the model's performance, likely due to the fact that all the states in the same phase have large overlap. Here, at 10 training points and 500 shadows per state, our scheme requires only a total of 5000 measurements, which again constitutes a very small number of shots, as compared to running the QCNN on a quantum computer. For

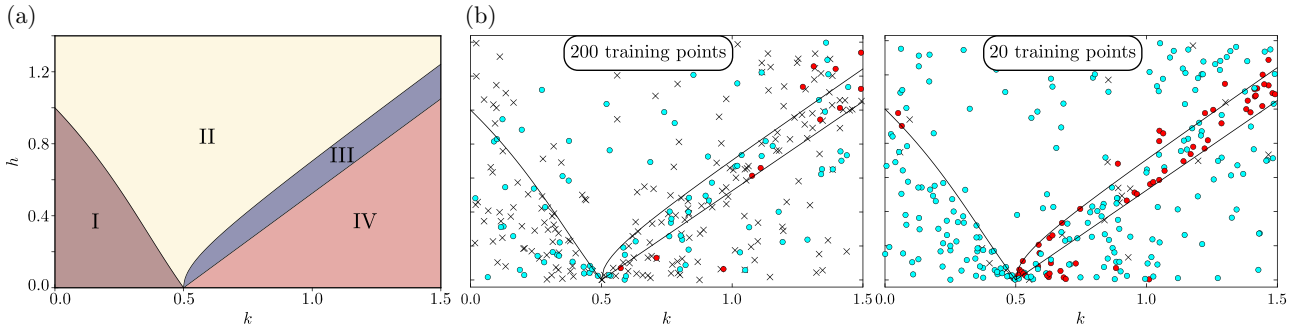


FIG. 5. Axial next-nearest neighbor Ising model. (a) Phase diagram for the Hamiltonian in Eq. (10). The phases are: (I) ferromagnetic, (II) paramagnetic, (III) floating, and (IV) antiphase. (b) Predicted phase diagram when training the simulated quantum convolutional neural network acting only on the low-bodyness operator subspace. The model is trained on with 200 and 20 states. The crosses mark the training points, while the circle the test ones. Blue circles means correct phase prediction, while a red color indicates that an incorrect phase was assigned.

a more detailed analysis of complexity scaling in the XXX bond-alternating model and the Haldane chain, as well as evidence for the absence of overfitting during training, we refer the reader to Appendix C.

3. ANNNI model

In this section, we consider the ANNNI model [120], which is described by the following Hamiltonian:

$$H = -J_1 \sum_{i=1}^{n-1} X_i X_{i+1} - J_2 \sum_{i=1}^{n-2} X_i X_{i+2} - B \sum_{i=1}^{n} Z_i. \quad (10)$$

The phases of this model are classified in terms of the dimensionless ratios $\kappa = -J_2/J_1$ and $h = B/J_1$. The phase diagram for this model is presented in Fig. 5. In particular, we will take $J_1 = 1$ and perform the classification based on the values κ and h of Ref. [75].

In this case we consider a system size of $n = 32$ qubits. Throughout the simulation we cap the maximum bodyness of the Heisenberg-evolved measurement operator at four, where we estimate the expectation of the initial states with nonidentity Pauli operators inside a sliding window of size 8 qubits. We refer the reader to Appendix B 1 b for more details on the truncation. Given that we are now dealing with multiclass classification, we assign phases by measuring two qubits at the QCNN's output and assigning labels using one-hot encoding. Our dataset is now composed of 300 states and we consider two scenarios where we train over 200 and 20 of them (taking 4000 and 2500 shadows per state, respectively). The training is averaged over five different runs. For 200 training points, we can achieve a train accuracy of 82.8%, and a test accuracy of 85.8%. Then, for 20 training points the train accuracy was of 87%, while the test of 80.2%. Thus we can reach similar results to those obtained in the literature, at a much smaller measurement budget.

4. Cluster Hamiltonian

To finish, we consider the cluster model defined by the following Hamiltonian:

$$H = \sum_{i=1}^{n} (Z_i - J_1 X_i X_{i+1} - J_2 X_{i-1} Z_i X_{i+1}). \quad (11)$$

Note that here we employ closed boundary conditions so that $X_{n+1} \equiv X_1$ and $X_{-1} \equiv X_n$. As shown in Fig. 6, the phase transitions are delimited by the ratios between J_1 and J_2.

Similarly to the previous ANNNI model, we consider a system size of $n = 32$ qubits. Throughout the simulation we cap the maximum bodyness of the Heisenberg-evolved measurement operator at four, where the nontrivial operators are taken from a sliding window of 8-qubits size. We again assign phases via one-hot encoding (i.e., measure two qubits in the computational basis and assigning a phase depending on what bistring probability is larger). The dataset now consists of 300 states and we consider training on subsets of 200 and 20 of them (taking 4000 shadows per state). The training is averaged over five different runs. For 200 training points, we find a train and test accuracies of 80.9% and 84% respectively. Then, for 20 training points we obtain a train accuracy of 85%, and a test one of 77.6%. In all cases, these accuracies are comparable and even higher than the ones achieved in previous attempts for the classification of the cluster Hamiltonian [77,78].

At this point we find it important to note that for both the ANNNI and cluster models most of the misclassification errors are located near the phase transitions. While this could arise due to the fact that phase transitions are precisely where the classification should become more difficult, we also note that there could be spurious errors arising from the fact that the "true" labels in our dataset are assigned by using the thermodynamic limit phase diagram,

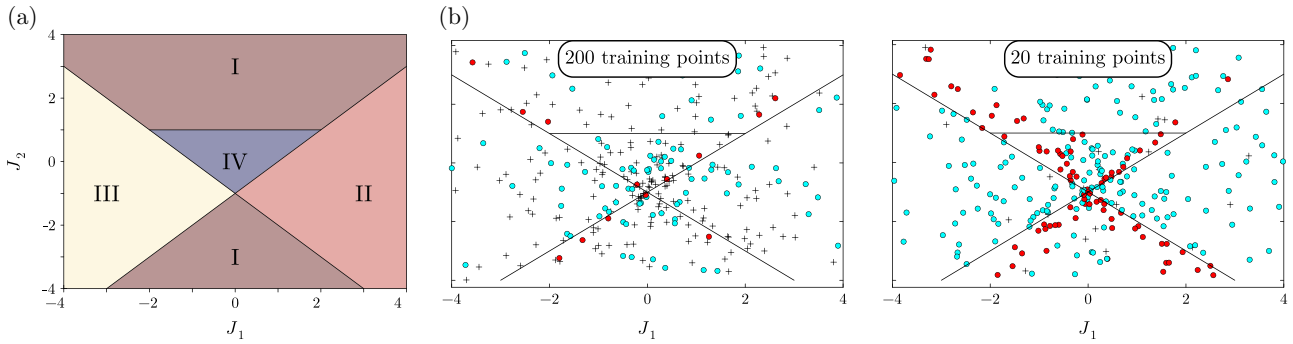


FIG. 6. Cluster model. (a) Phase diagram for the Hamiltonian in Eq. (11). The phases are: (I) Haldane, (II) ferromagnetic, (III) antiferromagnetic phase, and (IV) trivial. (b) Predicted phase diagram when training the simulated quantum convolutional neural networks acting only on the low-bodyness operator subspace. The model is trained on with 200 and 20 states. The crosses mark the training points, while the circle the test ones. Blue circles means correct phase prediction, while a red color indicates that an incorrect phase was assigned.

rather than the finite size one. This could lead to misalignments between data points and labels that increase as the system size decreases.

B. Classical datasets

The QCNN has been extensively employed in the literature to showcase the power of QML for classical data classification and characterization. In this section we will show that the most commonly used classical datasets in the literature, are locally easy. For that purpose, we will simulate the action of the QCNN for the following datasets: MNIST [51–62], Fashion-MNIST [51–59], EuroSAT [63–67] and GTSRB [60,61,68,69].

For all the datasets above we will study the simulation of the QCNN with both amplitude and angle encoding. For amplitude encoding, we load each image in the dataset at hand as a square matrix of floating point values, or triplets of them in case of colored data. Then, if needed (i.e., for the GTSRB and EuroSAT), we merge the color channels into a single gray-scale matrix. After resizing this matrix based on the dataset, it is reshaped as an unnormalized quantum state $|\psi(x)\rangle$ on $n = 8$ qubits. Last, we normalize $|\psi(x)\rangle$ and encode it in a matrix product state (MPS), by means of sequential singular value decompositions. Then, for angle encoding we will use a tensor product embedding, hardware-efficient embedding and the so-called classically hard embedding. We refer the reader to Ref. [125] for a detailed description of these data embedding schemes. Importantly, we note that we always take shallow embedding schemes (one and three layers for the hardware-efficient and classically hard) as it is well known that deep versions of these encoding are not useful [125], as they can lead to initial states that are too entangled and

TABLE I. Results for the classification of classical datasets with a simulated quantum convolutional neural network. Amplitude refers to amplitude encoding, while angle encodings are denoted by TPE (tensor product encoding), HEA 1/3 (hardware-efficient ansatz encoding with one or three layers) and CHE 1/3 (classically hard encoding with one or three layers). All the results are obtained with a constrained-bodyness of two operators, except for * which make use of three operators.

	Initial size	Train points	Test points	Amplitude (%)	TPE (%)	HEA 1/3 L (%)	CHE 1/3 L (%)
MNIST [51–62]							
0 / 1	28 × 28	40	100	100	100	100/98	98/100
0 / 9	28 × 28	60	100	91	99	93/87	91/91
Fashion-MNIST [51–59]							
Trouser/Sweater	28 × 28	60	100	98	97	96/95	96/93
Dress/Sneaker	28 × 28	60	100	97	100	99/95	98/100
EuroSAT [63–67]							
Industrial/Forest	64 × 64	200	400	98	94	94/92	92*/87*
Highway/Sealake	64 × 64	200	400	95	94	94/92	87/91
GTSRB [60,61,68,69]							
20 kmh/Stop	256 × 256	120	120	98	97	96/94	94/90
Bumpy road/Road work	256 × 256	160	120	94	98	94/90	86/82

thus void the QCNN's trainability guarantees [26]. A direct consequence of this fact is that all data-encoded states will all admit an efficient MPS decomposition. Finally, the simulation of the QCNN is conducted by means of the constrained-bodyness MPS algorithm described in the Appendix, and we use a mean-squared error loss function.

In Table I we show results for all the classification tasks considered. Here we can see that the simulated QCNN result in high test accuracies (showing best out of five independent runs), comparable to, and even larger than, those found in the literature.

Here it is important to note that unlike the quantum dataset case, where the dataset state's preparation could be hard to simulate (thus requiring shadows), all of the QML models used here are fully classically simulable. That is, there is no need to perform measurements on the classical data-encoded states as the embedding scheme itself is classically simulable (this is again a consequence of the results in Ref. [125]). As such, it appears that no quantum resources are needed to simulate the QCNN. Concomitantly, this implies that using QCNN-based QML schemes for classical data appears to be an ill-motivated task.

V. DISCUSSION

In this work we have shown, using a proof by demonstration and explicit construction, that one-dimensional tracing out and measurement-based QCNNs can be classically simulated—in the sense that we can construct and efficiently classical surrogates. Our explicit simulations are obtained by conceptualizing the success of QCNNs and showing that they appear to only work on easy problems for which their action can be restricted to polynomially sized subspaces. Clearly, our work cannot, and does not intend to, prove that there is no scenario whatsoever where it may be necessary to train a QCNN on a quantum computer. However, the burden of proof now rests firmly in the hands of any proponent of QCNNs to identify such cases and until then it is good practice to maintain a healthy skepticism that such cases can be found. Hence we boldly claim: *There is currently no evidence that QCNNs will work on classically nontrivial tasks, and their place in the upper echelon of promising QML architectures should be seriously revised.*

With the previous being said, we now present several important caveats to our results. First, while we focus here on the two most popular instantiations of QCNNs used in the literature (one-dimensional tracing out and measurement-based architectures), it is clear that these are the easiest to classically simulate. One could, for instance, envision QCNNs in two or more dimensions, as well as more exotic topologies. Clearly, this would increase the simulation cost, potentially making it prohibitively expensive even at modest sizes. However, even here, Pauli propagation methods, and more general advanced tensor networks techniques (such as projected entangled pair states with belief propagation) can handle these QCNNs [126]. In this work, we decided not to pursue this route as these QCNN architectures have not been explored, and are not known to be useful in cases where simple one-dimensional ones fail. As before, we again leave the burden of proof to practitioners to find schemes that avoid classical simulability but also provide real tangible advantages in nontrivial datasets.

Then, it is important to note that our claims do not amount to a full dequantization of QCNNs. Fundamentally, at least for the case of quantum input data, a quantum computer is needed to obtain classical shadows. However, the resource requirements (state preparation and then single-qubit measurements) are substantially easier than running the QCNN on quantum hardware—so much easier that a universal digital quantum computer may no longer be required. Thus our results can be viewed positively as expanding the prospects of the near-term friendly framework of quantum measurement-enhanced machine learning [96]. Another positive spin of our results is a possible setting of "*classical training and quantum deployment*," where one trains a QCNN on a classical device via classical shadows, and then uses the optimal parameters to implement and test the model's performance in a quantum computer on new data. This approach has the benefit of bypassing the need to obtain classical shadows from the new data instances. Then, a second positive spin of our work is that we can now start to compare the classical resources needed to perform a simulation, versus the quantum resources needed to actually train a simulable model on a quantum computer. In this "just because we can classically simulate a model, does not mean it is efficient to do so" perspective, there is ample room for actually deploying models on quantum computers if the classical cost is too large. We also leave this open as a future research question.

While we here focused on QCNNs, we remark that the results and lessons learned for this model apply to QML more broadly. We strongly believe that the techniques introduced here can serve as blueprints to classically simulate the information processing capabilities of other quantum neural networks architectures composed of local parametrized gates. Indeed, one can envision that all of the circuit architectures that are average-case simulable via the results in Ref. [88], could also be classically trained via low-bodyness approximations (on locally easy datasets). Whether there exists good enough solutions within such subspace as they do for QCNNs is an open question which will likely require a case-by-case analysis via direct simulation.

Finally, we argue that our community is in dire need of nontrivial datasets as our exhaustive literature search did not produce a single example of a task that cannot be classified by simulating the action of the QCNN in

the small subspace of low-bodyness observables. Indeed, for the considered condensed matter quantum datasets, the order parameters are local and allow for classification even through phase transitions. In fact, our results indicate that for classical data, shadow tomography is not even needed, as the whole model is entirely classically simulable. This result pushes back on the hope that the encoding scheme can somehow create classically hard to simulate features and sheds serious doubt on whether it even makes sense at all to embed classical data on a quantum computer for coherent processing via parametrized quantum circuits such as QCNNs. Whether the fact that we use trivial datasets in our QML model benchmarking is a positive bias effect (i.e., only successful QCNN trainings make it into the published literature) or a more underlying phenomenon of physical problems is left as an open question. Hence, we also put the burden of proof on practitioners to find nonlocally easy datasets that quantum models can classify when operating on a quantum computers, but that their classical surrogates cannot solve. Regardless, we believe that an introspection is needed when it comes to choosing QML benchmarking tasks to avoid using trivial ones.

ACKNOWLEDGMENTS

P. Bermejo, P. Braccia and L. Cincio were supported by the Laboratory Directed Research and Development (LDRD) program of Los Alamos National Laboratory (LANL) under Projects No. 20230527ECR and No. 20230049DR. P. Bermejo acknowledges constant support from DIPC. M.S.R. thanks Tyson Jones for helpful suggestions and discussions surrounding the implementation of the LOWESA algorithm. Z.H. acknowledges support from the Sandoz Family Foundation-Monique de Meuron program for Academic Promotion. M.C. acknowledges support from LANL's ASC Beyond Moore's Law project.

APPENDIX A: PROOF OF RESULT 1

In this section we present a proof for Result 1. We begin by introducing the basic definitions and theoretical tools needed.

1. *k*-purities

Our main goal is to show that the Heisenberg-evolved measurement operator essentially only has support in $\mathcal{O}(1)$-bodyness Paulis. As such, let $\{P_j\}$ denote the set of Pauli operators, and let $k = |P_j|$ denote the bodyness, or weight, of P_j, i.e., the number of qubits that P_j acts nontrivially on. Given an operator $O \in \mathcal{B}(\mathcal{H})$, the set of bounded operators on the Hilbert space $\mathcal{H}$, we define its k-purity as the projection of O into all the Paulis with bodyness k. That is,

$$p_O^{(k)} = \frac{1}{4^n} \sum_{P_j : |P_j| = k} \mathrm{Tr}[P_j O]^2. \tag{A1}$$

When O is such that $\|O\|_2^2 = \mathbb{1}$, a condition that we will henceforth assume, then $\sum_{k=1}^{n} p_O^{(k)} = 1$ and, since by definition $p_O^{(k)} \geqslant 0$ for all k, then the k-purities form a probability distribution. In particular, we are interested in computing the quantities

$$\mathbb{E}_{\boldsymbol{\theta}} \left[p^{(k)}_{\Phi_{\boldsymbol{\theta}}^\dagger(O)} \right], \tag{A2}$$

where $\Phi_{\boldsymbol{\theta}}^\dagger(O)$ denotes the measurement operator obtained from Heisenberg evolving the local measurement through a unitary QCNN as in Eq. (2).

Here it is important to note that we have assumed that averaging over $\boldsymbol{\theta}$ is equivalent to randomly sampling each local gate in $\Phi_{\boldsymbol{\theta}}$ independently from the Haar measure over $\mathbb{U}(4)$. Then, since the QCNN is unitary according to Eqs. (2), we can express its action as

$$\Phi_{\boldsymbol{\theta}}(\cdot) = U(\boldsymbol{\theta})(\cdot)U^\dagger(\boldsymbol{\theta}), \tag{A3}$$

where

$$U(\boldsymbol{\theta}) = \prod_l U_l. \tag{A4}$$

Above, each U_l denotes a two-qubit gate in the circuit [see Fig. 2(b, right)] and we have omitted the explicit parameter dependence on the right-hand side. Therefore, we have that

$$\mathbb{E}_{\boldsymbol{\theta}} = \prod_l \int_{\mathbb{U}(4)} d\mu_l(U_l), \tag{A5}$$

and in order to compute the average k-purities we need to integrate local unitaries sampled randomly according to the Haar measure over $\mathbb{U}(4)$. In the next section we will present the basic Weingarten Calculus tools for performing such calculations.

2. Weingarten calculus

We here recall a few basic concepts from Weingarten calculus. We refer the reader to Refs. [42,113,127] for additional details.

To begin, let us note that computing the k-purities requires evaluating quantities of the form

$$\mathbb{E}_{\boldsymbol{\theta}}\left[p^{(k)}_{\Phi^\dagger_{\boldsymbol{\theta}}(O)}\right] = \frac{1}{4^n}\sum_{P_j:|P_j|=k}\mathbb{E}_{\boldsymbol{\theta}}\left[\mathrm{Tr}[P_j U^\dagger(\boldsymbol{\theta})OU(\boldsymbol{\theta})]^2\right]$$
$$= \frac{1}{4^n}\sum_{P_j:|P_j|=k}\prod_l\int_{\mathbb{U}(4)} d\mu_l(U_l)\mathrm{Tr}[P_j^{\otimes 2}(U_l^\dagger)^{\otimes 2}O^{\otimes 2}U_l^{\otimes 2}]$$
$$= \frac{3^k\binom{n}{k}}{4^n}\prod_l\int_{\mathbb{U}(4)} d\mu_l(U_l)\mathrm{Tr}[P_j^{\otimes 2}U_l^{\otimes 2}O^{\otimes 2}(U_l^\dagger)^{\otimes 2}]. \quad \text{(A6)}$$

In the last equation we have used the fact that we can always transform one Pauli with a given bodyness onto another one with the same bodyness by local rotations that can be absorbed into the first convolutional layer gate's Haar measure. Above, we take P_j to be any Pauli with bodyness equal to k. Importantly, we can vectorize this equation to obtain

$$\mathbb{E}_{\boldsymbol{\theta}}\left[p^{(k)}_{\Phi^\dagger_{\boldsymbol{\theta}}(O)}\right] = \frac{3^k\binom{n}{k}}{4^n}\langle\langle P_j^{\otimes 2}|\prod_l \hat{\tau}_l^{(2)}|O^{\otimes 2}\rangle\rangle, \quad \text{(A7)}$$

where we defined the second moment operator for the local U_l gate

$$\hat{\tau}_l^{(2)} = \int_{\mathbb{U}(4)} d\mu_l(U_l)U_l^{\otimes 2}\otimes(U_l^*)^{\otimes 2}. \quad \text{(A8)}$$

Here we recall that the vectorization takes an operator in $\mathcal{B}(\mathcal{H}^{\otimes t})$ and returns a vector in $\mathcal{H}^{\otimes t}\otimes(\mathcal{H}^*)^{\otimes t}$ while a channel from $\mathcal{B}(\mathcal{H}^{\otimes t})$ to $\mathcal{B}(\mathcal{H}^{\otimes t})$ is mapped to a matrix in $\mathcal{B}(\mathcal{H}^{\otimes t}\otimes(\mathcal{H}^*)^{\otimes t})$. Specifically, given some $X=\sum_{i,j=1}^{d^t}c_{ij}|i\rangle\langle j|$, its vectorized form is $|X\rangle\rangle=\sum_{i,j=1}^{d^t}c_{ij}|i\rangle\otimes|j\rangle$, while given a channel $\Phi(X)=\sum_{\nu=1}^{d^{2t}}K_\nu X J_\nu^\dagger$, we obtain $\hat{\Phi}=\sum_{\nu=1}^{d^{2t}}K_\nu\otimes J_\nu^*$. In particular, the inner product between two vectorized operators is given by $\langle\langle Y|\hat{\Phi}|X\rangle\rangle=\mathrm{Tr}[Y^\dagger\Phi(X)]$.

Equation (A7) reveals that computing the average k-purities requires evaluating the product of the moment operators $\hat{\tau}_l^{(2)}$. These can be computed via the Weingarten calculus as follows. We start by considering a compact unitary Lie group G with Haar measure $d\mu$ acting on a finite-dimensional Hilbert space $\mathcal{H}$. We are interested in computing the tth fold moment operator, which takes the form

$$\hat{\tau}_G^{(t)}(X) = \int_G d\mu(U)U^{\otimes t}\otimes(U^*)^{\otimes t}. \quad \text{(A9)}$$

It is well known that the moment operator is a projection onto the (vectorized) tth order commutant of G, i.e., the operator vector space $\mathrm{comm}^{(t)}(G)=\{M\in\mathcal{B}(\mathcal{H}^{\otimes t})\,|\,[M,U^{\otimes t}]=0\}$, of dimension $d_{G,t}=\dim(\mathrm{comm}^{(t)}(G))$. As such, given a basis $\{P_\mu\}_{\mu=1}^{d_{G,t}}$ of $\mathrm{comm}^{(t)}(G)$, one can express the moment operator as

$$\hat{\tau}_G^{(t)} = \sum_{\mu,\nu=1}^{d_{G,t}}(W_{G,t}^{-1})_{\nu\mu}|P_\nu\rangle\rangle\langle\langle P_\mu|. \quad \text{(A10)}$$

Here $W_{G,t}^{-1}$ is known as the Weingarten matrix with entries $(W_{G,t})_{\nu\mu}=\mathrm{Tr}[P_\nu^\dagger P_\mu]$. That is, $W_{G,t}^{-1}$ is the inverse of the commutant's Gram matrix.

We can then proceed to apply this framework to our case of interest. In particular, we will prove the following Lemma (see also Refs. [116,128,129] for proofs of similar statements).

Lemma A1. Let $U(\boldsymbol{\theta})=\prod_l U_l$ be a circuit composed of two-qubit gates as in Fig. 2(b,left) and assume that each U_l is sampled independently from a 2-design over $\mathbb{U}(4)$. Then, the associated second moment operator $\hat{\tau}_l^{(2)}$ associated to each U_l can be represented by the 4×4 dimensional matrix P,

$$P=\begin{pmatrix}1 & \frac{2}{5} & \frac{2}{5} & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & \frac{2}{5} & \frac{2}{5} & 1\end{pmatrix}, \quad \text{(A11)}$$

acting on the subspace spanned by $\{|i\rangle,|s\rangle\}^{\otimes 2}$, where $|i\rangle\equiv|\mathbb{1}\rangle\rangle$ and $|s\rangle\equiv|\mathrm{SWAP}\rangle\rangle$.

Proof. let us begin by taking a two-qubit gate U_l acting on qubits j and j'. Since U_l is sampled from a 2-design over $\mathbb{U}(4)$, we recall that the twofold commutant of the unitary group is $\mathrm{comm}^{(2)}(\mathbb{U}(4))=\mathrm{span}_{\mathcal{C}}\{\mathbb{1}_j\otimes\mathbb{1}_{j'},\mathrm{SWAP}_j\otimes\mathrm{SWAP}_{j'}\}$ [113]. Here $\mathbb{1}_j$ denotes the identity on the two copies of the j th qubit Hilbert spaces, while SWAP_j the operation that interchange these two Hilbert spaces (and similarly for $\mathbb{1}_{j'}$ and $\mathrm{SWAP}_{j'}$. From Eq. (A10) we know that $\hat{\tau}_l^{(2)}:=\hat{\tau}_{\mathbb{U}(4)}^{(2)}$ will be a projector onto $|\mathbb{1}_j\otimes\mathbb{1}_{j'}\rangle\rangle=|\mathbb{1}_j\rangle\rangle\otimes|\mathbb{1}_{j'}\rangle\rangle$ and $|\mathrm{SWAP}_j\rangle\rangle\otimes|\mathrm{SWAP}_{j'}\rangle\rangle$. When two consecutive two-qubit gates act as in Fig. 2(b, right), say U_l acting on qubits j and j' and U_{l+1} acting on qubits j and j'', we can see that one of the qubits is shared. In this case, $\hat{\tau}_l^{(2)}$ and $\hat{\tau}_{l'}^{(2)}$ will be projectors onto different subspaces. However, their joint action can be studied by expanding them onto the vector space spanned by the overlaps of their commutants. For instance, $\hat{\tau}_l^{(2)}$ can be fully studied by its action on a four-dimensional vector space

spanned by the basis vectors

$$\{|i\rangle, |s\rangle\}^{\otimes 2}, \tag{A12}$$

where $|i\rangle \equiv |\mathbb{1}\rangle\rangle$ and $|s\rangle \equiv |\mathrm{SWAP}\rangle\rangle$. In particular, we can explicitly find that

$$\begin{aligned} &\hat{\tau}_l^{(2)}|ii\rangle = |ii\rangle, \quad \hat{\tau}_l^{(2)}|is\rangle = \frac{2}{5}|is\rangle + \frac{2}{5}|si\rangle, \\ &\hat{\tau}_l^{(2)}|si\rangle = \frac{2}{5}|is\rangle + \frac{2}{5}|si\rangle, \quad \hat{\tau}_l^{(2)}|ss\rangle = |ss\rangle, \end{aligned} \tag{A13}$$

indicating that the second moment, $\hat{\tau}_l^{(2)}$, is given by the 4×4 matrix P, which we dub P-gate,

$$P = \begin{pmatrix} 1 & \frac{2}{5} & \frac{2}{5} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{2}{5} & \frac{2}{5} & 1 \end{pmatrix}. \tag{A14}$$

■

Combining Eqs. (A7) and (A14) we find that the average k-purity can be obtained by projecting $|O^{\otimes 2}\rangle\rangle$ and $|P_j^{\otimes 2}\rangle\rangle$ into the i and s basis and evolving them through a circuit composed of P gates respecting the same topology as that of the QCNN. We refer the reader to Refs. [116,129,130] for additional details on this general procedure.

3. Exact derivation of the k-purities for a prototypical QCNN ansatz

To begin, let us recall the claim of Result 1:

Result 3 (Informal). Consider QCNNs where the convolutional layers are composed of general parametrized two-qubit gates acting on nearest neighboring qubits in a pattern such as those of Fig. 2(b, right). In average, the contribution in $\Phi_{\boldsymbol{\theta}}^{\dagger}(O)$ of a given Pauli with bodyness k, decays exponentially with k.

In what follows, we will prove the following theorem, which constitutes a formal version of the previous claim:

Theorem A1. Consider a QCNN acting on $n = 2^{\eta}$ qubits, with $\eta \in \mathbb{N}$, as in Fig. 2(b, right), or alternatively as in Fig. 7(b,left). That is,

$$\Phi_{\boldsymbol{\theta},\boldsymbol{\lambda}} = \bigcirc_{l=1}^{\eta-1}\left(P_l^{\boldsymbol{\lambda}_l} \circ C_l^{\boldsymbol{\theta}_l}\right), \tag{A15}$$

where $P_l^{\boldsymbol{\lambda}_l}(\cdot) = \mathrm{Tr}_{S_l}[\cdot]$ is a pooling layer where the qubits in the subset S_l are traced out and where $C_l^{\boldsymbol{\theta}_l} = U_l(\theta_l)(\cdot)U_l(\theta_l)$ is a convolutional layer such that

$$U_l(\boldsymbol{\theta}_l) = \bigotimes_{t=1}^{n_l/2} U_l^t(\boldsymbol{\theta}_l^t), \tag{A16}$$

where U_l^t is a two-qubit gate acting on qubits $2t$ and $2t-1$. Above, $n_l = 1,\ldots,2^{\eta-l}$ and $S_l = \{1,3,5,\ldots,\eta-1\}$. Then, assuming that U_l^t forms an independent 2-design over the unitary group $\mathbb{U}(4)$ and denoting the average $\mathbb{E}_{\boldsymbol{\theta}} = \prod_{l=1}^{\eta-1}\prod_{t=1}^{n_l/2}\mathbb{E}_{\boldsymbol{\theta}_l^t}$ one obtains that

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\theta}}\left[p^{(k)}_{\Phi_{\boldsymbol{\theta}}^{\dagger}(O)}\right] = \frac{2}{3}\left(\frac{3}{2}\right)^k \sum_{k_L^s=1}^{2k_{L-1}^s}\left(\frac{1}{2}\right)^{2k_L^s-k}\sum_{k_{L-1}^s=1}^{2k_{L-2}^s}\cdots \\ \times \sum_{k_3^s=1}^{2k_2^s}\sum_{k_2^s=1}^{2k_1^s}\binom{2k_1^s}{k_2^s}\binom{2k_2^s}{k_3^s}\cdots \\ \times \binom{2k_{L-1}^s}{k_L^s} a^{1+2\sum_{l=1}^{L-1}k_l^s}, \end{aligned} \tag{A17}$$

where O is a single-qubit Pauli operator, and $p^{(k)}_{\Phi_{\boldsymbol{\theta}}^{\dagger}(O)}$ is the k-purity as defined in Eq. (A1).

We proceed to apply the machinery introduced in the previous section to compute the average k-purities of an operator that is Heisenberg-evolved throughout the QCNN architecture shown in Fig. 7(b,left). By construction, the gates in each layer of the QCNN do not cross, i.e. they act on independent pairs of qubits. We choose to study this particular configuration because it reduces the mixing of operators, keeping the exact calculation of the operator bodyness tractable. Moreover, it provides a reliable low-bound for the operator mixing in alternative QCNN architectures. When the convolutional layer is composed of two layer of single-qubit gates as in Fig. 2(b, right), then the proof of Result 1 can be found in Ref. [26].

Proof. As previously mentioned, the measurement operator O is assumed to be a single Pauli operator (acting, e.g., on the topmost remaining qubit). As per Lemma A1, we are required to work in the $\{|i\rangle, |s\rangle\}^{\otimes n}$ basis, so we will rewrite this operator as $|O^{\otimes 2}\rangle\rangle = \frac{2}{3}|s\rangle|i\rangle^{\otimes n-1}$. Given that each gate on our ansatz maps $|ii\rangle$ to $|ii\rangle$, we are interested in studying the propagation of $|si\rangle$ throughout the QCNN, spreading from $|s\rangle|i\rangle^{\otimes n-1}$. In Fig. 7, we exemplify the spread of $|si\rangle$ throughout the first 2 layers of the QCNN, keeping track of the coefficients picked up from each gate.

By construction, the topology of our QCNN ansatz enforces each P-gate to act on, at least, one $|i\rangle$. Hence, we can restrict our study to the remaining input operator, which can be either $|s\rangle$ or $|i\rangle$. We are interested in characterizing the support of the output of the network of P-gates

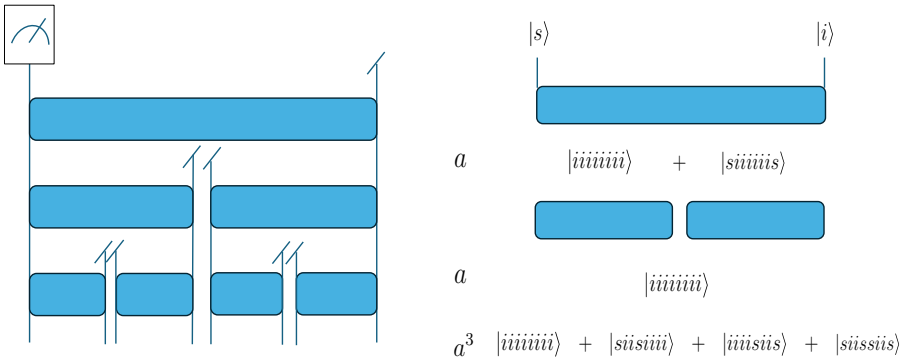


FIG. 7. Schematic representation of the proof technique. In this figure we show a quantum convolutional neural network architecture (left) and the spread of the vectorized measurement operator as it propagates through the P gates of the first two layers of the quantum convolutional neural network. The initial state is given by $|s\rangle$ on the first qubit and $|i\rangle$ on the remaining qubits. Here, we also include the coefficient a defining the contribution of each operator to the k-purities, which is obtained after applying the P-gate on each gate and equals $2/5$, according to Eq. (A14). Note that the quantum convolutional neural network on the left is exactly the same as that in Fig. 2(b,left) up to an unimportant swapping of the qubits at the output of each unitary in the convolutional layers.

over the nontrivial components of $\{|i\rangle, |s\rangle\}^{\otimes n}$, grouped by their $|s\rangle$ content. Noticing that each operator $|s\rangle$ entering a P-gate will either generate a trivial output $|ii\rangle$ or a nontrivial one $|ss\rangle$, we deduce that, after each layer, the number of $|s\rangle$ terms appearing in each possible basis state reached by our ansatz is even. Thus, we now introduce a variable k_l^s, which we refer to as the s-content, counting the number of pairs of $|s\rangle$ operators appearing in the output states after layer l is applied. It is easy to see that given a state with k_{l-1}^s its outputs after layer l will contain all k_l^s from zero, corresponding to each gate mapping its $|si\rangle$ input to $|ii\rangle$, to $2k_{j-1}^s$, corresponding to the case where all the outputs are $|ss\rangle$. Each of the possible states on layer l with s-content k_l^s appears exactly $\binom{2k_{l-1}^s}{k_l^s}$ times.

Assuming the number of qubits n is such that the QCNN comprises L layers, we can then calculate all the possible appearances of nontrivial basis states with fixed s-content k_L^s at the end of the circuit, $N(k_L^s)$, as

$$N_p(k_L^s) = \sum_{k_{L-1}^s=1}^{2k_{L-2}^s} \cdots \sum_{k_3^s=1}^{2k_2^s} \sum_{k_2^s=1}^{2k_1^s} \binom{2k_1^s}{k_2^s}\binom{2k_2^s}{k_3^s} \cdots \binom{2k_{L-1}^s}{k_L^s} = \sum_{k_{L-1}^s=1}^{2k_{L-2}^s} \cdots \sum_{k_3^s=1}^{2k_2^s} \sum_{k_2^s=1}^{2k_1^s} n(k_1^s, \ldots, k_L^s), \tag{A18}$$

subjected to $k_1^s = 1$. This constraint follows from the initial state $|s\rangle\,|i\rangle^{\otimes n-1}$ only branching into a state with a single $|ss\rangle$ and the completely trivial state (which we discard for the purpose of the analysis). Notice that we also introduced the number of configurations with a fixed evolution of the s-content $n(k_1^s, \ldots, k_L^s)$. Now that we have classified the support of the average Heisenberg-evolved $|O^{\otimes 2}\rangle\rangle$ in the basis $\{|i\rangle, |s\rangle\}^{\otimes n}$ by its s-content k_L^s at the end of the circuit, we can proceed to analyze the contribution of each of these terms to the distribution of k-purities. First of all, we need to incorporate the coefficients they pick up when each gate is applied. Considering that the first layer always generates a coefficient $a = 2/5$, and each $k_{l>1}^s$ does also carry a multiplicative factor $a^{2k_l^s}$, each operator at the end of the circuit will pick up a coefficient $a^{m(k_1^s,\ldots,k_{L-1}^s)}$ depending on the path it followed thorough the network of P-gates, where the exponent $m(k_1^s, \ldots, k_{L-1}^s)$ is obtained as

$$m(k_1^s, \ldots, k_{L-1}^s) = 1 + 2\sum_{l=1}^{L-1} k_l^s. \tag{A19}$$

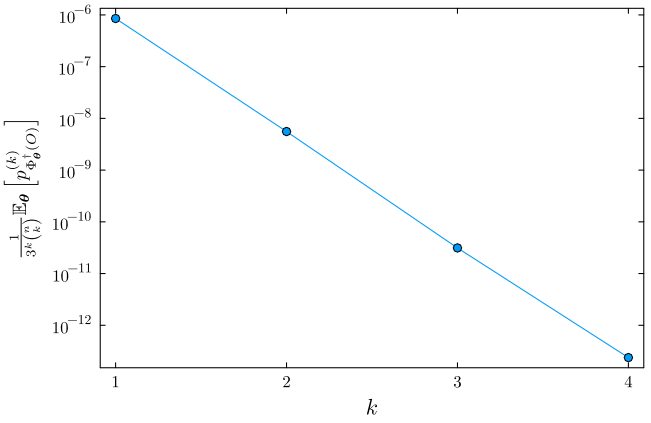


FIG. 8. Average contribution per bodyness for different problem sizes. In this figure we plot $(1/3^k\binom{n}{k})\mathbb{E}_{\boldsymbol{\theta}}\left[p^{(k)}_{\Phi^\dagger_{\boldsymbol{\theta}}(O)}\right]$ for a system of $n = 128$ qubits. This quantity determines the average contribution of a given Pauli with bodyness k on the Heisenberg-evolved measurement operator.

Last, we map the s-content to Pauli bodyness and obtain the k-purities. To do this, we recall that the operator $|s\rangle$ is defined as $|s\rangle = (|\mathbb{1}^{\otimes 2}\rangle\rangle + |X^{\otimes 2}\rangle\rangle + |Y^{\otimes 2}\rangle\rangle + |Z^{\otimes 2}\rangle\rangle)/2$, which we can schematically rewrite as $|s\rangle = \frac{1}{2}|\mathbb{1}^{\otimes 2}\rangle\rangle + \frac{3}{2}|P^{\otimes 2}\rangle\rangle$ since there is no distinction between Paulis P in the analysis of Pauli weight. We can thus see that each state with s-content k_L^s splits into k-bodied Paulis where k goes from zero to $2k_L^s$, and where one associates to each of these Paulis a coefficient $(\frac{3}{2})^k\,(\frac{1}{2})^{2k_L^s-k}$, since the remaining identities $|i\rangle$ do also map to identities $|\mathbb{1}^{\otimes 2}\rangle\rangle$ and pick up no coefficient.

Putting everything together, and reintroducing the initial coefficient $\alpha = 2/3$, we finally arrive at the exact expression for our average QCNN k-purities,

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\theta}}\left[p^{(k)}_{\Phi^\dagger_{\boldsymbol{\theta}}(O)}\right] &= \frac{2}{3}\left(\frac{3}{2}\right)^k \sum_{k_L^s=1}^{2k_{L-1}^s}\left(\frac{1}{2}\right)^{2k_L^s-k}\sum_{k_{L-1}^s=1}^{2k_{L-2}^s}\cdots\sum_{k_3^s=1}^{2k_2^s}\sum_{k_2^s=1}^{2k_1^s} n(k_1^s,\ldots,k_L^s)\,a^{m(k_1^s,\ldots,k_{L-1}^s)}\\ &= \frac{2}{3}\left(\frac{3}{2}\right)^k \sum_{k_L^s=1}^{2k_{L-1}^s}\left(\frac{1}{2}\right)^{2k_L^s-k}\sum_{k_{L-1}^s=1}^{2k_{L-2}^s}\cdots\sum_{k_3^s=1}^{2k_2^s}\sum_{k_2^s=1}^{2k_1^s}\binom{2k_1^s}{k_2^s}\binom{2k_2^s}{k_3^s}\cdots\binom{2k_{L-1}^s}{k_L^s} a^{1+2\sum_{l=1}^{L-1}k_l^s}. \end{aligned} \tag{A20}$$

■

We can evaluate Eq. (A20) and plot the ensuing results in Fig. 8 where we plot the average contribution of each Pauli with bodyness k to the k-purities (i.e., we plot $\frac{1}{3^k\binom{n}{k}}\mathbb{E}_{\boldsymbol{\theta}}\left[p^{(k)}_{\Phi^\dagger_{\boldsymbol{\theta}}(O)}\right]$). Here we can see that the Heisenberg-evolved measurement operator through a randomly initialized QCNN has support essentially only on Paulis with bodyness in $\mathcal{O}(1)$. Indeed, we can also verify via the results in Ref. [116] that a QCNN with a topology such as that in Fig. 2(b, right) follows exactly the same behavior.

APPENDIX B: CLASSICAL SIMULABILITY OF QCNNs

Having realized that the success of QCNNs arises from the fact that they are initialized, explore and end their training in the polynomially large subspace of low-bodyness observables, we proceed to describe methods to classically simulate their action on this subspace. All techniques are based on the idea of only processing the information encoded in low-bodyness measurements of the input states. In particular, the methods presented here truncate the Heisenberg-evolved measurement operators to low bodyness [i.e., bodyness in $\mathcal{O}(1)$], which allows us to guarantee an efficient representation of this backward-evolved operator. Then, the overlap with the initial states are obtained by projecting each ρ_i into the low-bodyness subspace (e.g., by first taking Pauli classical shadows in the case of quantum datasets). The first simulation method we use is based on the LOWESA algorithm [90,91], which belongs to the class of *Pauli propagation surrogates*, within the family of *Pauli propagation* algorithms. For the second method, we employ tensor networks with restricted bodyness.

1. LOWESA

To introduce the novel variant of LOWESA, which we classify as a *Pauli propagation surrogate* (PPS), we begin by giving an overview of the underlying PP simulation. Then we will move on to the PPS we use for QCNN classification.

a. Pauli propagation

PP refers to the simulation of an expectation value in the form

$$\langle O\rangle = \mathrm{Tr}[U(\boldsymbol{\theta})\rho U^\dagger(\boldsymbol{\theta})O] = \mathrm{Tr}\left[\rho U^\dagger(\boldsymbol{\theta})OU(\boldsymbol{\theta})\right], \tag{B1}$$

via a Pauli path integral approach [131]. Specifically, we apply the parametrized quantum circuit $U(\boldsymbol{\theta})$ to the observable O in the Pauli transfer matrix (PTM) formalism [132] and then individually compute the trace of each resulting Pauli operator with the initial state ρ. We highlight that our following PP notation assumes *normalized Pauli operators* such that $\mathrm{Tr}[P_i^2] = 1$.

First, let us define

$$U[\cdot] := U^\dagger(\cdot)U \tag{B2}$$

as the action of an operator in the PTM formalism. Furthermore, we define the quantum circuit as

$$U(\boldsymbol{\theta}) = \prod_{i=1}^{m} U_i(\theta_i), \tag{B3}$$

where $U_i(\theta) = e^{-i\theta P_i}$ is a unitary generated by the Pauli operator P_i with parameter θ_i. The circuit could also contain Clifford operators like H or CNOT, or channels like

Pauli noise that are diagonal in PTM representation at very low computational cost. However, the examples of QCNNs we consider in this work can be fully rewritten in terms of Pauli gates.

We can write the action of a Pauli gate on a (normalized) Pauli operator P_j as

$$U_i(\theta)[P_j] = e^{i\theta P_i} P_j e^{-i\theta P_i} \tag{B4}$$

$$= \left(\cos(\frac{\theta}{2})I + i\sin(\frac{\theta}{2})P_i\right) P_j \times \left(\cos\left(\frac{\theta}{2}\right) I - i\sin\left(\frac{\theta}{2}\right) P_i\right). \tag{B5}$$

This general formula simplifies depending on whether P_i and P_j commute (i.e., $[P_i, P_j] = 0$) or anticommute (i.e., $\{P_i, P_j\} = 0$). Using trigonometric identities, we find that

$$U_i(\theta)[P_j] = \begin{cases} P_j, & \text{if } [P_i, P_j] = 0, \\ \cos(\theta)P_j - i\sin(\theta)P_iP_j, & \text{if } \{P_i, P_j\} = 0. \end{cases} \tag{B6}$$

With the product of Pauli operators given as $P_iP_j = i\epsilon_{ijk}P_k$ for $P \neq I$, we see that the application of a Pauli gate to a Pauli operator either leaves the Pauli operator unchanged, or it creates two different Pauli operators with *real-valued* trigonometric coefficients. This implies that simulating such a quantum circuit can create a number of Pauli operators that is exponential in the number of gates m.

After applying the entire quantum circuit, we receive a weighted sum of Pauli operators,

$$U(\boldsymbol{\theta})[O] = \sum_\alpha c_\alpha(\boldsymbol{\theta}) P_\alpha, \tag{B7}$$

where $c_\alpha(\boldsymbol{\theta})$ are trigonometric polynomials. This expression goes over *unique* Pauli operators because we *merge* Pauli paths, i.e., add their coefficients if Pauli operators become equal at some stage of the quantum circuit. Using this Pauli path formulation, the original expectation function in Eq. (B1) becomes

$$\langle O\rangle = \mathrm{Tr}[\rho U^\dagger(\boldsymbol{\theta}) O U(\boldsymbol{\theta})] \tag{B8}$$

$$= \sum_\alpha c_\alpha(\boldsymbol{\theta}) \mathrm{Tr}[\rho P_\alpha] \tag{B9}$$

$$= \sum_\beta \sum_\alpha c_\beta c_\alpha(\boldsymbol{\theta}) \mathrm{Tr}[P_\beta P_\alpha], \tag{B10}$$

where $\rho = \sum_\beta c_\beta P_\beta$ is the decomposition of the initial state in the Pauli basis.

We can restrict the simulation to the polynomially small operator subspace of low-body operators by truncating a Pauli path if the propagating Pauli operator crosses a truncation threshold. This strongly reduces the sum over α to a classically manageable subset of low-body Pauli operators.

The crucial insight in Eq. (B10), which leads to the PP surrogates, is that the initial states ρ are known at the time of simulation. This means that both the magnitude of the coefficients c_β and overlaps $\mathrm{Tr}[P_\beta P_\alpha]$ could in principle be computed before starting the optimization of the parameters $\boldsymbol{\theta}$. One could therefore invest precomputation time to identify which indices β are most important to the task at hand, and only evaluate along the Pauli paths leading to those Pauli operators P_β. This is the idea behind PP surrogates.

b. Pauli propagation surrogate

PPS are a family of PP algorithms that trade precompute time and memory for drastically faster evaluation of the expectation function. As such, PPS algorithms can be classified as *classical surrogates* [133]. The PPS algorithm we employ in this work is a novel variant of the LOWESA presented in Refs. [90,91]. As hinted at in the previous section, instead of numerically evaluating all coefficients c_α during the *Pauli propagation*, we only store a graphical representation of the *splitting* and *merging* of Pauli paths during an initial *surrogation* pass. We then identify which paths lead to the most important P_β operators and only re-evaluate the computed graph along those paths. A schematic depiction of this is shown in Fig. 9.

Consider the simple but common case of $\rho = |0\rangle\langle 0| = (I + Z)^{\otimes n}$, where the individual traces $\mathrm{Tr}[\rho P_\alpha] \in \{0, 1\}$ are either 0 or 1 depending on if P_α contains X, Y operators on any qubit or not, respectively. Again, note the fact that we are using normalized Pauli operators and can thus drop the $\frac{1}{2}$ factor per qubit. The number of operators that contain only I, Z operators is generally much lower than the number of all propagated Pauli operators, but only they contribute to the expectation value. This means that the graph representation of the expectation value can be re-evaluated only on the paths leading to those operators, resulting in a significant speedup.

In this work, we do not consider such simple initial states but instead classical shadow representations [93] of more intricate quantum states. That is, we use measurements of ρ in random Pauli bases to estimate the values $c_\beta = \mathrm{Tr}[\rho P_\beta]$. Replacing ρ with its classical shadow $\rho_s = \sum_\beta c_{\beta,s} P_\beta$, we receive the expectation estimator

$$\langle O\rangle \approx \sum_\alpha c_\alpha(\boldsymbol{\theta}) \mathrm{Tr}[\rho_s P_\alpha] \tag{B11}$$

$$= \sum_\beta \sum_\alpha c_{\beta,s} c_\alpha(\boldsymbol{\theta}) \mathrm{Tr}[P_\beta P_\alpha]. \tag{B12}$$

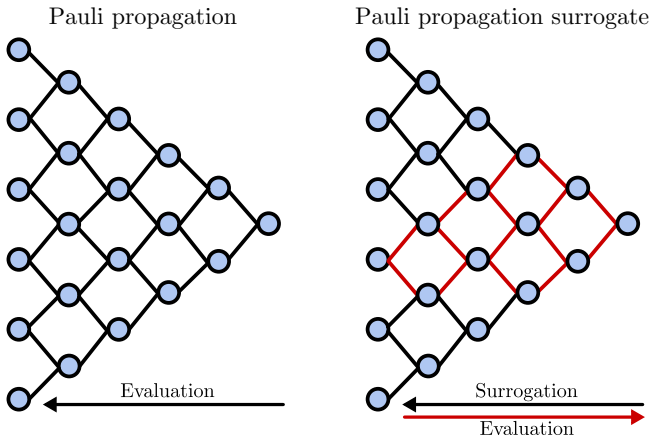


FIG. 9. Schematic comparison between Pauli propagation and Pauli propagation surrogate methods. Both methods are based on propagating Pauli operators through a quantum circuit which will in general create new Pauli operators via splitting of paths. If we employ a *breadth-first* approach to the propagation, we can identify identical Pauli operators and merge them. The splitting and merging pattern depicted here is strongly simplified. In conventional Pauli propagation, one would numerically evaluate the coefficients throughout the circuit and then calculate the overlap with the initial state to estimate the expectation value. In contrast, Pauli propagation surrogate first compute a graph representation of the Pauli paths, which we call the *surrogation* step. Evaluation of the expectation value is then performed only along the paths that result in operators that are deemed important for the task.

Propagation truncations—To maintain the efficiency of the simulation, we employ truncations of the Pauli propagation during the initial surrogation. First, we restrict the propagation to low-bodyness operators. Depending on the difficulty of the problem, we choose a truncation threshold k, and if a Pauli operator crosses this threshold at any part of the circuit, it is not propagated further. As such, the operators P_α in Eqs. (B11) and (B12) have at most this bodyness k. This truncation is proven to be highly effective for average-case simulability in the accompanying Ref. [88]. Furthermore, we employ a so-called *frequency* truncation which affects the coefficients $c_\alpha(\boldsymbol{\theta})$. We define the frequency of a path as the number of sines and cosines it has picked up. The average coefficient of a path with ℓ sines and cosines over the entire parameter landscape is $(\frac{1}{2})^\ell$, which gives frequency truncation the intuitive meaning of an average coefficient truncation. Both truncations have been used in Ref. [91] with good practical success.

Surrogate truncations—While the complexity of the initial surrogation can be controlled by the propagation truncation above, PP surrogates allow further truncations that speed up expectation and gradient evaluations. First, we do not always choose to estimate the expectation values of all operators P_β up to bodyness k. This would imply that we expect the underlying correlations in the training data to be *all-to-all*. Instead, we can choose to evaluate operators that only contain nonidentity Pauli operators in a *sliding window* between 1D neighboring qubits, i.e., nonidentity Pauli operators are taken from a subset of adjacent qubits, which is displaced along the qubit chain. If the data are 1D structured, nontrivial Pauli operators on adjacent qubits are expected to carry most of the discriminating information. Other subsets of low-bodyness operators can be chosen on a case-by-case basis. Another strategy we employ to sparsify the evaluation graph is based on the variance over the input states. We can estimate the variance of the coefficients $c_{\beta,s}$ across the dataset (potentially inside the sliding window) and only evaluate along the paths with high variance. This leverages the heuristic that high-variance paths are likely the ones that are most useful for classifying the states, as opposed to paths that have the same coefficient irrespective of the class label.

2. Constrained-bodyness tensor network

Tensor networks stand out for their convenient properties to represent vector states based on their local features. Moreover, they serve as a suitable platform to efficiently perform certain computations that would otherwise result in prohibitive computational costs. This is exemplified in a recent study of random quantum circuits [116], where the authors introduced an efficient MPS based algorithm to perform the projection of a vectorized operator into the subspace of k-bodyness Paulis. This setting paves the way to control the bodyness of the operators in an efficient manner, enabling tensor networks to deal with polynomially sized operator subspaces and simulate efficiently a wide variety of ansätze under this operator constraint. In this work, we made use of this construction to show that the classical datasets considered in the main text are indeed classifiable with only access to the reduced subspace of low-bodyness operators during the entire training.

This method is currently applicable to binary classification tasks and fares best with one-dimensional quantum systems, we leave for future work the extension to multiclass problems and higher-dimensional topologies. We now proceed to explain the details of the technique.

In outline, the method consists of three steps. First we preprocess the input data, be it quantum or classical, in such a way that only the low-bodyness information is kept. Then, we solve the binary classification task by finding the optimal classifying operator as represented by an MPS in the low-bodyness operator subspace. Last, we compile the quantum circuit ansatz at hand such that the Heisenberg-evolved measurement operator is as close as possible to the one found at the previous step.

The preprocessing of the input data works as follows. Consider we are given a set of quantum states $\{\rho_i\}$, which could be either the result of some purely quantum experiment or the result of encoding classical data into a quantum computer. We can resort to shadow tomography techniques

[93,95,134] to efficiently reconstruct the components of each state over the subspace of small-weight Paulis, resulting in a truncated, classical dataset $\{\tilde{\rho}_i\}$. Of course, when the data are classical to start with, we can straightforwardly encode it into the low-body space. For instance, one can simulate the encoding circuit via MPS methods and later project the resulting states via the projectors ϕ_k introduced in Ref. [116]. No matter the path taken, we then vectorize the states $\tilde{\rho}_i$ and normalize them to obtain a new dataset $|\tilde{\rho}_i\rangle\rangle$ where the truncated density matrices are encoded as quantum states in MPS form. Notice that in doing this, the physical dimension of the quantum systems constituting the quantum computational register goes from d to d^2, namely, for the qubits systems we deal with, from 2 to 4.

In this formalism quantities such as $\mathrm{Tr}[\rho O]$ can be readily expressed as inner products $\langle\langle\rho|O\rangle\rangle$, which in turn are easy to compute when the states involved can be efficiently represented as MPSs. Hence, we can carry out the training by optimizing overlaps between MPSs, where the particular optimization scheme depends on the loss function chosen to guide the learning task. As an example, we can formalize this description in the case of the mean-squared error, which is a common loss function for supervised learning tasks, and the one used in the problems considered in this manuscript. We recall that the mean-squared error loss function is defined by

$$\mathrm{MSE}(\theta) = \frac{1}{N_t}\sum_{i=1}^{N_t}\left(y_i - \mathrm{Tr}\left(\tilde{\rho}_i U^\dagger(\boldsymbol{\theta})\mathrm{MU}(\boldsymbol{\theta})\right)\right)^2.$$

Here, the sum runs over the whole training set, $\tilde{\rho}_i$ stands for the density matrix representation of our truncated ith training sample, y_i stands for the known ith label, M is the measurement operator and $U^\dagger(\boldsymbol{\theta})(\cdot)U(\boldsymbol{\theta})$ represents the Heisenberg-evolved operator parametrized by the parameters $\boldsymbol{\theta}$. Using our MPS setting, one would reformulate this expression as

$$\mathrm{MSE}(|O\rangle\rangle) = \frac{1}{N_t}\sum_{i=1}^{N_t}\left(y_i - \langle\langle\tilde{\rho}_i|O\rangle\rangle\right)^2,$$

where $|O\rangle\rangle$ replaces the Heisenberg-evolved operator $|U^\dagger(\boldsymbol{\theta})\mathrm{MU}(\boldsymbol{\theta})\rangle\rangle$ in the vectorized picture, and where now the optimization is carried out on the tensors defining the MPS $|O\rangle\rangle$.

Indeed, optimizing MPS overlaps can be done in an efficient manner, without resorting to parametric gradient computation [135–137], resulting in a considerable speedup with respect to gradient based methods. We employ this technique to swiftly find the best operator $|O\rangle\rangle$ that solves the classification task at hand. Notice that not only the data $|\tilde{\rho}_i\rangle\rangle$ lives in the low-bodyness subspace, but the MPS representing the evolved measurement operator is also constrained to the optimization within the low-bodyness subspace. This constraint reproduces the behavior of an average QCNN under random initialization and proves to be sufficient to solve the task at hand. Hence, after optimization, the second phase returns an optimal MPS representing a low-weight measurement operator that is able to classify the training set. We can now proceed with the third and last step, which is optional with respect to the classification task, but which is needed to find the optimal parameters of the circuital ansatz we started with. This step simply corresponds to compiling the ansatz $U(\theta)$ at hand, realized as a tensor network $\hat{U}(\theta)$ acting on the MPS representation of some final measurement operator $|M\rangle\rangle$, to be as close as possible to the found optimal $|O\rangle\rangle$. In the case considered in this manuscript, a 1-d QCNN having logarithmic number of layers, the corresponding tensor network representation of the QCNN can never increase the bond dimension of the MPS it acts on by an exponential (in the number of qubits n) amount. Hence the full simulation and compilation of the QCNN is always efficient. In general, when dealing with hard-to-simulate ansätze we can employ strategies akin to those described for the PP and PPS methods, basically interleaving the layers of the ansatz with projectors onto the low-body subspace. In any case, the compilation can be carried out optimizing the overlap $\langle\langle O|\hat{U}(\theta)|M\rangle\rangle$ between the optimal and parametrized Heisenberg evolved measurement operator. To this end, either numerical gradient methods or more precise tensor network techniques can be used [135,137,138].

APPENDIX C: ANALYSIS OF SCALING AND OVERFITTING DURING TRAINING

We choose the XXX model and the Haldane chain to showcase how the number of shadows required to perform successful training varies as a function of the system size. Moreover, in order to inspect as well the potential effect of overfitting, we modify the amount of samples employed in the training and testing datasets. In the main text, we used a dataset composed of 100 samples, using up to 75 of them as the training dataset for the XXX and Haldane models. Here, we enlarge the dataset to 500 samples, out of which 400 are employed in the training stage (Figs. 10 and 11).

As we argued in the manuscript, which is further evidenced in these complementary results, the complexity of the classification does not substantially increase with system size. The number of shadows to perform successful classification in the XXX model increases sublinearly with system size, while there it seems to be independent of the system size in the case of the Haldane model.

One does not expect the intrinsic features of the data samples to change with system size, so that the kind of operators (in our case, amount and type of Pauli strings) employed to perform classification should not vary either. One can visualize this idea by considering a ferromagnetic

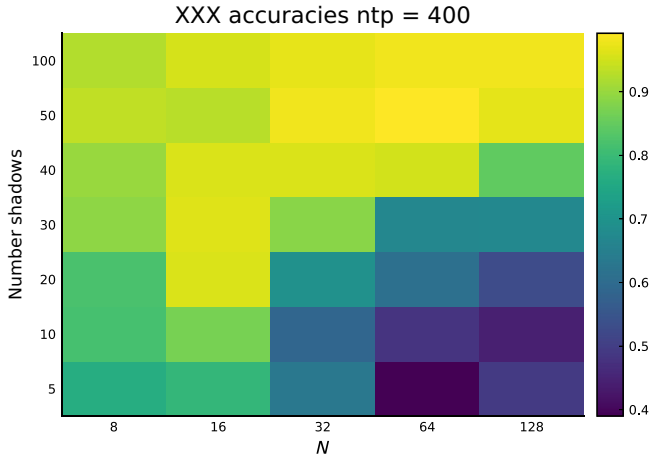


FIG. 10. Bond-alternating XXX model. Classification accuracy as a function of number of qubits and number of shadows.

material, where all spins are pointing at a given direction, and an antiferromagnetic one. A faithful measurement of 2-body operators (which can tell the difference in local magnetization) should be enough to tell phases apart.

We would also like to point out that significantly more hypertuning can be conducted in the optimization process. In the case of the XXX model, where only 2-body operators were used for the classification, it could be the case that system size effects are stronger, thus explaining the sublinear increase in the number of shadows with system size. However, it is also possible that further hypertuning of the model (preselection of relevant Pauli strings, increase in number of iterations, etc.) lead to a classification process which barely depends on system size.

Last, we want to remark that no overfitting seems to be taking place in the optimization process, since we increase the number of training points and the number of shadows required to perform classification stays barely the same compared to the results shown in the main text with significantly lower number of training points.

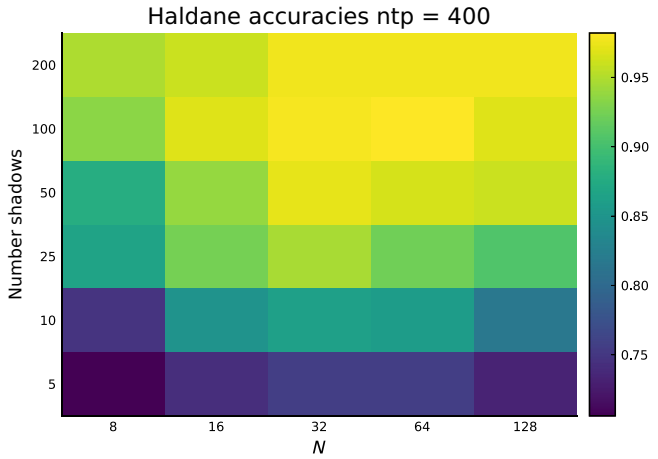


FIG. 11. Bond-alternating Haldane model. Classification accuracy as a function of number of qubits and number of shadows.

APPENDIX D: RANDOM FOREST CLASSIFICATION OF THE XXX BOND-ALTERNATING MODEL

In this complementary analysis, we study the classification of the two ground states of the XXX bond-alternating model using the random forest algorithm, a widely employed method in the machine learning literature. These results are aimed at showcasing that the dataset is locally easy, as we can perfectly classify the phases of matter by just considering single-qubit Paulis and a few two-qubit ones.

We generate 500 ground states from an $n = 100$ qubit XXX bond-alternating model via DMRG(for $J_1 = 1, 250$ states belong to $0 < J_2 < 1$, and 250 states fall within $1 < J_2 < 2$). The features fed to the decision trees in the random forest are constructed as follows: the overlap of each ground state with all operators containing a single nontrivial (Pauli) operator on the string, plus 500 operators containing two nontrivial operators (with no preference for which positions the nontrivial operators occupy). We use 350 states for training (evenly split between phases and randomly distributed within each phase) and 150 states for testing. Since the number of features is moderate, we employ as many trees in the random forest as there are input features. Finally, we average the predicted labels over 200 independent trainings. The resulting classification is shown in Fig. 12.

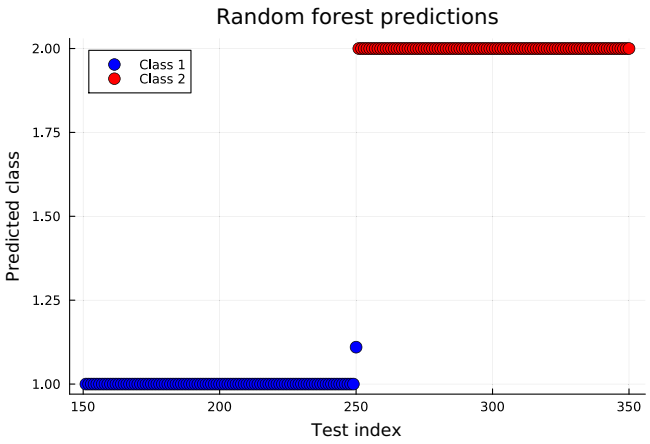


FIG. 12. Classification of ground states of the XXX bond-alternating model with $n = 100$ qubits using a random forest. Features include overlaps of the ground states with operators containing a single nontrivial (Pauli) operator (and identity elsewhere), plus 500 operators composed of two nontrivial operators.

As clearly observed in Fig. 12, the classification achieves 100% accuracy. This indicates that the information contained in the module of a single nontrivial operator, complemented by some information from operators with two nontrivial operators, is sufficient to distinguish the two families. This result, entirely independent of the QCNN, demonstrates the intrinsic features of the datasets, making them easily classifiable.

APPENDIX E: MEASUREMENT-BASED QCNN

In this section we will study QCNNs where at each pooling layer, qubits are measured rather than traced out, and where the measurement outcomes are used to control unitaries applied to neighboring qubits [50].

1. Concentration properties and Heisenberg-evolved measurement operator

First, let us define the two-qubit channel consisting of a random two-qubit unitary U followed by a measurement on the first qubit, and a controlled unitary $V(x)$ on the second qubit [see Fig. 13(a)]. As such, if the first qubit is measured on the state $x = 0$ (or 1), then we apply a unitary $V(0)$ (or $V(1)$) onto the second qubit. The action of this channel on a two-qubit state ρ is given by

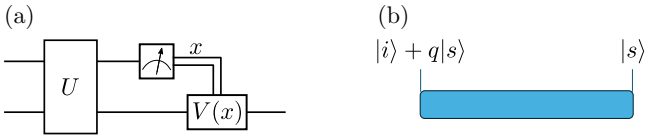


FIG. 13. (a) Unitary (from a convolutional layer) and the basic unit of the pooling layer: a measurement on a qubit, followed by a controlled unitary. (b) Adding the measurement in the pooling layer changes the input to the P gate from $|i\rangle|s\rangle$ to $(|i\rangle + q|s\rangle)|s\rangle$.

$$\mathcal{N}(\rho) = \sum_{x\in\{0,1\}} p(x)(|x\rangle\langle x| \otimes V(x))U\rho U^\dagger(|x\rangle\langle x| \otimes V^\dagger(x)), \tag{E1}$$

where $p(x) = \mathrm{Tr}[(|x\rangle\langle x| \otimes \mathbb{1})\rho\,|x\rangle\langle x| \otimes \mathbb{1}]$. Via vectorization, we can express the previous channel as

$$\hat{\mathcal{N}} = \sum_{x\in\{0,1\}} p(x)(|x\rangle\langle x| \otimes V(x)) \otimes (|x\rangle\langle x| \otimes V^*(x)) \cdot \left(U \otimes U^*\right). \tag{E2}$$

Assuming that U is sampled according to the Haar measure over $\mathbb{U}(4)$, and taking the expectation value $\mathbb{E}[\cdot] = \int_{\mathbb{U}(4)} d\mu(U)[\cdot]$, we obtain the moment operator

$$\begin{aligned}
\mathbb{E}[\hat{\mathcal{N}}] &= \int_{\mathbb{U}(4)} d\mu(U) \sum_{x\in\{0,1\}} p(x)(|x\rangle\langle x| \otimes V(x)) \otimes (|x\rangle\langle x| \otimes V^*(x)) \cdot \left(U \otimes U^*\right) \\
&= \sum_{x\in\{0,1\}} p(x)(|x\rangle\langle x| \otimes \mathbb{1}) \otimes (|x\rangle\langle x| \otimes \mathbb{1}) \cdot \int_{\mathbb{U}(4)} d\mu(U) \left(U \otimes U^*\right) \\
&= \sum_{x\in\{0,1\}} p(x)(|x\rangle\langle x| \otimes \mathbb{1}) \otimes (|x\rangle\langle x| \otimes \mathbb{1}) \frac{|\mathbb{1}\otimes\mathbb{1}\rangle\rangle\langle\langle \mathbb{1}\otimes\mathbb{1}|}{4},
\end{aligned}$$

where in the second equality we have used the right- and left-invariance of the Haar measure and in the third we used Eq. (A10). From the previous, we can see if we seek to estimate the expectation value of a traceless operator O on the second qubit, then we need to compute $\langle\langle \mathbb{1}\otimes\mathbb{1}|\mathbb{1}\otimes O\rangle\rangle = 0$, which matches the expected value of zero as in the tracing out QCNN.

Then, the second moment operator, denoted as $\mathbb{E}[\hat{\mathcal{N}}^{(2)}]$, can be found as [139]

$$\begin{aligned}
\mathbb{E}[\hat{\mathcal{N}}^{(2)}] = &\left(\sum_{x,y\in\{0,1\}} p(x)p(y)(|x\rangle\langle x| \otimes V(x)) \otimes (|y\rangle\langle y| \otimes V(y)) \otimes (|x\rangle\langle x| \otimes V^*(x)) \otimes (|y\rangle\langle y| \otimes V^*(y))\right) \\
&\times \left(\int_{\mathbb{U}(4)} d\mu(U) U^{\otimes 2} \otimes (U^*)^{\otimes 2}\right).
\end{aligned} \tag{E3}$$

Note that here we cannot use the left- and right-invariance of the Haar measure to absorb the action of the unitaries $V(x)$ and $V(y)$. For instance, when $x \neq y$ we obtain terms of the form

$$\int_{\mathbb{U}(4)} d\mu(U)(V(x)U) \otimes (V(y)U) \otimes (V^*(x)U^*) \otimes (V^*(y)U^*) \neq \int_{\mathbb{U}(4)} d\mu(U) U^{\otimes 2} \otimes (U^*)^{\otimes 2}. \tag{E4}$$

Instead, by using Eq. (A10) we find (up to reordering of indexes)

$$\int_{\mathbb{U}(4)} d\mu(U) U^{\otimes 2} \otimes (U^*)^{\otimes 2} = \frac{1}{15}\Big(|\mathbb{1}_1 \otimes \mathbb{1}_2\rangle\rangle\langle\langle \mathbb{1}_1 \otimes \mathbb{1}_2| + |\mathrm{SWAP}_1 \otimes \mathrm{SWAP}_2\rangle\rangle\langle\langle \mathrm{SWAP}_1 \otimes \mathrm{SWAP}_2| - \frac{1}{4}(|\mathbb{1}_1 \otimes \mathbb{1}_2\rangle\rangle\langle\langle \mathrm{SWAP}_1 \otimes \mathrm{SWAP}_2| + |\mathrm{SWAP}_1 \otimes \mathrm{SWAP}_2\rangle\rangle\langle\langle \mathbb{1}_1 \otimes \mathbb{1}_2|)\Big), \tag{E5}$$

where we recall that $\mathbb{1}_j$ and SWAP_j denote the identity and swap operators acting on the two copies of the j th qubit. From the previous, we see that we need to evaluate the terms

$$\begin{aligned}&(|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (V(x) \otimes V(y)) \otimes (|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (V^*(x) \otimes V^*(y))|\mathbb{1}_1 \otimes \mathbb{1}_2\rangle\rangle \\ &= (|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (\mathbb{1} \otimes \mathbb{1}) \otimes (|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (\mathbb{1} \otimes \mathbb{1})|\mathbb{1}_1 \otimes \mathbb{1}_2\rangle\rangle\end{aligned}$$

and

$$\begin{aligned}&(|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (V(x) \otimes V(y)) \otimes (|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (V^*(x) \otimes V^*(y))|\mathrm{SWAP}_1 \otimes \mathrm{SWAP}_2\rangle\rangle \\ &= (|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (\mathbb{1} \otimes \mathbb{1}) \otimes (|x\rangle\langle x| \otimes |y\rangle\langle y|) \otimes (\mathbb{1} \otimes \mathbb{1})|\mathrm{SWAP}_1 \otimes \mathrm{SWAP}_2\rangle\rangle.\end{aligned}$$

The previous equations show that when acting on the second Haar moment operator, the action of the $V(x)$ and $V(y)$ disappears thanks to the presence of the projectors on the first qubit. Hence, we need to evaluate the controlled unitary-independent term

$$\mathbb{E}[\hat{\mathcal{N}}^{(2)}] = \sum_{x,y\in\{0,1\}} p(x)p(y)(|x\rangle\langle x| \otimes \mathbb{1}) \otimes (|y\rangle\langle y| \otimes \mathbb{1}) \otimes (|x\rangle\langle x| \otimes \mathbb{1}) \otimes (|y\rangle\langle y| \otimes \mathbb{1}) \left(\int_{\mathbb{U}(4)} d\mu(U) U^{\otimes 2} \otimes (U^*)^{\otimes 2}\right). \tag{E6}$$

At this point, we find it convenient to consider the previous term in the context of a Heisenberg-evolved measurement operator. For simplicity let us assume that the channel $\mathcal{N}$ is applied at the end of the circuit to the last two qubits prior to the measurement of the expectation value of a Pauli operator O on the second qubit. As seen in Fig. 13(b), this means that going into the second leg of the P gate is an $|s\rangle$ operator coming from projecting O into the $|i\rangle$ and $|s\rangle$ basis (where we recall that $|i\rangle = |\mathbb{1} \otimes \mathbb{1}\rangle\rangle$ and $|s\rangle = |\mathrm{SWAP}\rangle\rangle$). Then, the input to the first leg is the projection of $|x\rangle\langle x| \otimes |y\rangle\langle y|$ onto the $|i\rangle$ and $|s\rangle$ basis. One can readily find that

$$|\,|x\rangle\langle x| \otimes |y\rangle\langle y|\rangle\rangle = |i\rangle + \delta_{x,y}\,|s\rangle\,, \tag{E7}$$

when adding the probabilities, we find that the input to the first leg is then

$$\begin{aligned}\sum_{x,y\in\{0,1\}} p(x)p(y)\left(|i\rangle + \delta_{x,y}\,|s\rangle\right) &= |i\rangle + (p(0)^2 + p(1)^2)\,|s\rangle \\ &= |i\rangle + q\,|s\rangle\,,\end{aligned} \tag{E8}$$

where we defined $q = p(0)^2 + p(1)^2$ with $0 \leqslant q \leq 1$ [where the upper bound is saturated if and only if $p(x) = 1$ for some x, indicating that the measured qubit is a computational basis state, i.e., no entanglement between the two qubits].

Using similar arguments as the ones derived above, we now need to study how the input $\frac{2}{3}\,|s\rangle \bigotimes_{j=1}^{n-1}(|i\rangle + q_j\,|s\rangle)$ spreads through the QCNN. Here, we defined $q_j = p_j(0)^2 + p_j(1)^2$ and $p_j(x)$ is the probability of measuring qubit j on state $x \in \{0,1\}$. At this point, we thus find it important to make several remarks. First, we can see that, adding measurements to the QCNN simply changes the measurement operator to be back-propagated as

$$\underbrace{\frac{2}{3}\,|s\rangle\,|i\rangle^{\otimes n-1}}_{\text{without measurements}} \quad\rightarrow\quad \underbrace{\frac{2}{3}\,|s\rangle \bigotimes_{j=1}^{n-1}(|i\rangle + q_j\,|s\rangle)}_{\text{with measurements}}. \tag{E9}$$

Next, since the action of the convolutional layers remains the same with or without measurements [as per the rightmost term of Eq. (E6)] the second moment of expectation

values simply changes as

$$\underbrace{\frac{2}{3}\langle\langle\rho^{\otimes 2}|\mathcal{P}\,|s\rangle\,|i\rangle^{\otimes n-1}}_{\text{without measurements}} \rightarrow \underbrace{\frac{2}{3}\langle\langle\rho^{\otimes 2}|\mathcal{P}\left(|s\rangle\bigotimes_{j=1}^{n-1}(|i\rangle+q_j\,|s\rangle)\right)}_{\text{with measurements}}, \tag{E10}$$

where $\mathcal{P}$ denotes the moment operator composed of products of P gates defined in Eq. (A14). In particular, it is clear that since the expectation value of $\langle\langle\rho^{\otimes 2}|$ with any operator in $\{|i\rangle, |s\rangle\}$ is positive (as it simply corresponds to a purity in a reduced subsystem on the qubits indicated by the s-indexes) then we find

$$\frac{2}{3}\langle\langle\rho^{\otimes 2}|\mathcal{P}\,|s\rangle\,|i\rangle^{\otimes n-1} \leqslant \frac{2}{3}\langle\langle\rho^{\otimes 2}|\mathcal{P}\left(|s\rangle\bigotimes_{j=1}^{n-1}(|i\rangle+q_j\,|s\rangle)\right) \tag{E11}$$

and hence, if all gates in the convolutional layer form independent two designs,

$$\mathrm{Var}_{\boldsymbol{\theta}}\left[\mathrm{Tr}[\Phi_{\boldsymbol{\theta}}^{\mathrm{tr}}(\rho_i)O]\right] \leqslant \mathrm{Var}_{\boldsymbol{\theta}}\left[\mathrm{Tr}[\Phi_{\boldsymbol{\theta}}^{\mathrm{meas}}(\rho_i)O]\right], \tag{E12}$$

where we defined the channels implementing the QCNN with tracing out as $\Phi_{\boldsymbol{\theta}}^{\mathrm{tr}}$, and with measurements as $\Phi_{\boldsymbol{\theta}}^{\mathrm{meas}}$. Equation (E12) shows that the QCNN with measurements concentrates less than the QCNN without them, implying that adding measurements decreases the expressive power of the QCNN. In particular, we note that the specific details of the convolutional layers are encoded into $\mathcal{P}$, and that Eq. (E12) holds irrespective of how the trainable gates are arranged (i.e., one-dimensional QCNN, two-dimensional QCNN, etc.).

Another consequence of Eq. (E9) is that when studying the bodyness in $\mathbb{E}_{\boldsymbol{\theta}}\left[p^{(k)}_{\Phi_{\boldsymbol{\theta}}^{\mathrm{meas}\dagger}(O)}\right]$, only the local terms will dominate, implying that randomly initialized measurement-based QCNNs can also only see local information in the input state. This is due to the fact that in $\mathcal{P}\,|s\rangle\bigotimes_{j=1}^{n-1}(|i\rangle+q_j\,|s\rangle) = \mathcal{P}\left(\sum_{x\in\{0,1\}^{\otimes n-1}}\mathcal{Q}^x\,|s\rangle\otimes_{j=1}^{n-1}|s^{x_j}\rangle\right)$ with $\mathcal{Q}^x = q_1^{x_1}\times\cdots q_{n-1}^{x_{n-1}}$ and $|s^0\rangle = |i\rangle$, the coefficients decay exponentially with the number of s in the Heisenberg-evolved operator. At this point, one may wonder about the case when $q_j = 1\ \forall j = 1,\ldots,n-1$, as here the input to $\mathcal{P}$ is simply $|s\rangle^{\otimes n}$, which satisfies the property $\mathcal{P}\,|s\rangle^{\otimes n} = |s\rangle^{\otimes n}$ (as per the definition of the P gates). In this pathological case, a global operator is able to move through the P-gates without becoming exponentially suppressed, and the QCNN could see the initial state globally. However, it is easy to see that this case does not appear, as prior to the measurement, a Haar random two-qubit gate is applied. Thus, in average, the output of such gate is the maximally mixed state $(\mathbb{1}\otimes\mathbb{1})/4$ from which one obtains $\mathbb{E}[p_j(0)] = \mathbb{E}[p_j(1)] = \frac{1}{2}$, and concomitantly $\mathbb{E}[q_j] = \frac{1}{2}$ for all measured qubits. Moreover, beyond the average-case scenario, one only obtains $q_j = 1$ for all j if the input state to all measurements in an unentangled computational basis state, indicating that the QCNN is able to disentangle the input state via its simple convolutional layers. This implies an entanglement structure that can be readily reproduced by tensor networks, so that both the QCNN and the input data are classically simulable. In the next section, we discuss how tensor networks can still be used to simulate a measurement-based QCNN on more general input states.

2. Classical simulability

In the previous section, we have seen that randomly initialized measurement-based QCNNs will only "see" the local information in the initial state. Given that this information can be captured with Pauli classical shadows, we here explore whether classical shadows plus tensor networks enable for the efficient simulation of such QCNN architectures. Note that the fact that we use measurement information for feed-forward operations precludes the use of Heisenberg-evolution type techniques such as LOWESA or other Pauli propagation techniques. On the other hand, as measurements destroy entanglement in the system (pooling layers are in the class of local operations and classical communication, and thus, they cannot increase entanglement on average [140]), and as Pauli shadows lead to separable initial tensor product states, this combination actually enables efficient simulation vis standard MPS techniques.

In fact, we have explicitly performed the aforementioned simulation for a characteristic QCNN architecture, similar to the one employed for the numerical results of Sec. IV. Namely, the architecture consists of convolutional layers where two-qubit gates act on pairs of neighboring qubits in a brick-like fashion. Then, in the pooling layer, the qubits that will no longer be part of a convolutional operation are measured, and the measurement outcome controls a single qubit unitary on one of its nearest neighbors (e.g., if qubit i is measured, then the controlled unitary is applied to qubit $i+1$).

In Fig. 14 we show how the maximum bond dimension χ of the MPS scales for different system sizes as the initial state evolves through the QCNN. In this example, the initial state is a random product state, and the unitaries forming the convolutional layers, together with the unitaries from the controlled measurements, are randomly sampled from $\mathbb{U}(4)$ and $\mathbb{U}(2)$, respectively. From the plots, one can observe an initial increase in the bond dimension

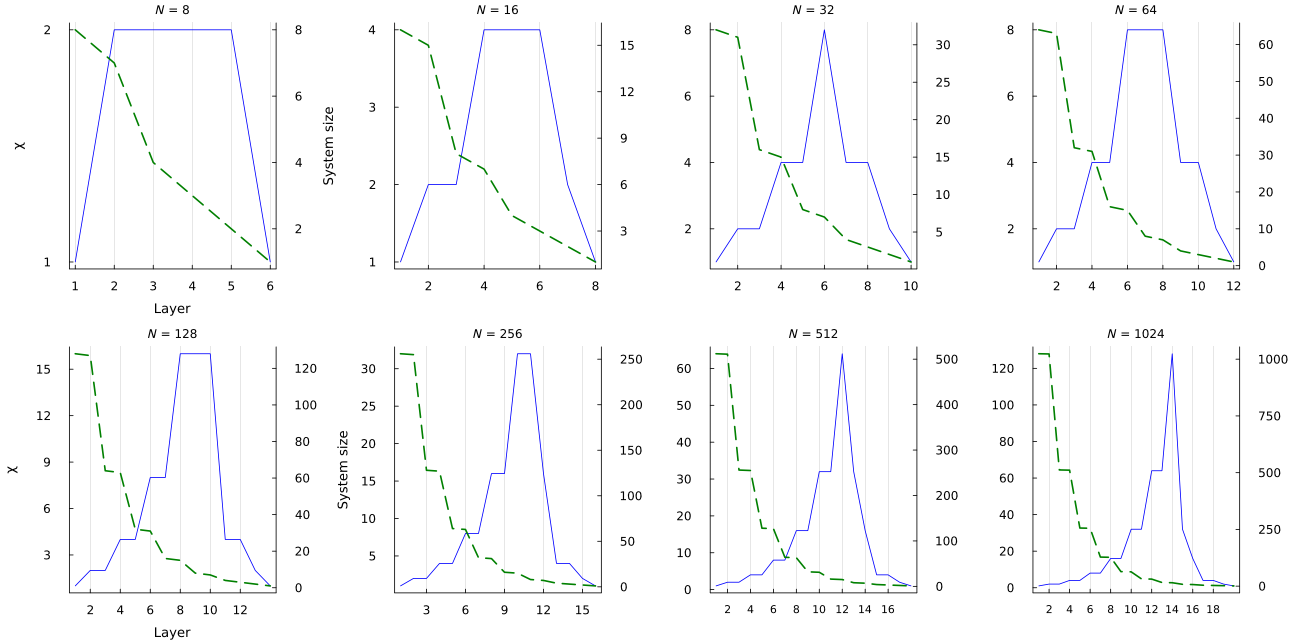


FIG. 14. Average bond dimension χ and systems size for a product state evolving through a typical n-qubit measurement-based quantum convolutional neural network. The horizontal axis depicts the layer number $l = 0, \ldots \log(n)$, the solid blue curve the bond dimension, and the dashed green curve the system size of the quantum state at the lth layer. Results are shown for $n = 2^m$ for $m = 3, 4, \ldots, 10$. All unitaries in the convolutional layer, as well as the control unitaries, were randomly sampled from $\mathbb{U}(4)$ and $\mathbb{U}(2)$, respectively.

as the state is evolved throughout the convolutional layers, and initial pooling layers. Such growth, is eventually stopped by the fact that after each pooling layer half of the qubits are measured-out, effectively reducing the system size. After the crossover point, correlations start to decrease and are limited by the system size. At the end of the QCNN, a single-qubit state remains and the bond dimension is equal to its starting value of one.

We find that, at most, the bond dimension scales as $\chi \sim n/8$. Since the MPS simulation cost scales as $\mathcal{O}(\chi^3)$, determined by the SVD performed after applying each gate, the total cost scales as $\mathcal{O}(n^3)$, i.e., polynomially in n. With the previous being said, it is worth mentioning that in practice even modest polynomial scaling can quickly become intractable. However, state-of-the-art methods allow MPS simulations on a laptop with bond dimension up to $\chi \sim 4096$, meaning that a laptop could simulate a QCNN with up to 32 768 qubits, far beyond the capabilities of current quantum computers. It is also worth noting that the previous simulation used random quantum gates in order to reproduce the worst-case scenario, whereas in practice one would expect smaller values of χ during a typical training process.

[1] S. Y. Chang and M. Cerezo, A primer on quantum machine learning, arXiv:2511.15969.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT, 2016).

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

[4] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Netw. **61**, 85 (2015).

[5] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, arXiv:2104.13478.

[6] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature **549**, 195 (2017).

[7] M. Schuld and F. Petruccione, *Supervised Learning with Quantum Computers* (Springer, 2018), Vol. 17.

[8] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, Variational quantum algorithms, Nat. Rev. Phys. **3**, 625 (2021).

[9] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke *et al.*, Noisy intermediate-scale quantum algorithms, Rev. Mod. Phys. **94**, 015004 (2022).

[10] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles, Challenges and opportunities in quantum machine learning, Nat. Comput. Sci. **2**, 567 (2022).

[11] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, Hybrid quantum-classical algorithms and quantum error mitigation, J. Phys. Soc. Jpn. **90**, 032001 (2021).

[12] M. Larocca, S. Thanasilp, S. Wang, K. Sharma, J. Biamonte, P. J. Coles, L. Cincio, J. R. McClean, Z. Holmes,

and M. Cerezo, A review of barren plateaus in variational quantum computing, Nat. Rev. Phys. **3**, 625 (2025).
[13] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nat. Commun. **9**, 1 (2018).
[14] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nat. Commun. **12**, 1 (2021).
[15] M. Cerezo and P. J. Coles, Higher order derivatives of quantum neural networks with barren plateaus, Quantum Sci. Technol. **6**, 035006 (2021).
[16] A. Arrasmith, Z. Holmes, M. Cerezo, and P. J. Coles, Equivalence of quantum barren plateaus to cost concentration and narrow gorges, Quantum Sci. Technol. **7**, 045015 (2022).
[17] M. Ragone, B. N. Bakalov, F. Sauvage, A. F. Kemper, C. Ortiz Marrero, M. Larocca, and M. Cerezo, A Lie algebraic theory of barren plateaus for deep parameterized quantum circuits, Nat. Commun. **15**, 7172 (2024).
[18] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, PRX Quantum **3**, 010313 (2022).
[19] L. Bittel and M. Kliesch, Training variational quantum algorithms is NP-hard, Phys. Rev. Lett. **127**, 120502 (2021).
[20] E. Fontana, M. Cerezo, A. Arrasmith, I. Rungger, and P. J. Coles, Non-trivial symmetries in quantum landscapes and their resilience to quantum noise, Quantum **6**, 804 (2022).
[21] E. R. Anschuetz and B. T. Kiani, Quantum variational algorithms are swamped with traps, Nat. Commun. **13**, 7760 (2022).
[22] E. R. Anschuetz, in *International Conference on Learning Representations* (2022).
[23] M. Larocca, P. Czarnik, K. Sharma, G. Muraleedharan, P. J. Coles, and M. Cerezo, Diagnosing barren plateaus with tools from quantum optimal control, Quantum **6**, 824 (2022).
[24] P. Bermejo, B. Aizpurua, and R. Orús, Improving gradient methods via coordinate transformations: Applications to quantum machine learning, Phys. Rev. Res. **6**, 023069 (2024).
[25] E. Gil-Fuster, C. Gyurik, A. Pérez-Salinas, and V. Dunjko, On the relation between trainability and dequantization of variational quantum learning models, arXiv:2406.07072.
[26] A. Pesah, M. Cerezo, S. Wang, T. Volkoff, A. T. Sornborger, and P. J. Coles, Absence of barren plateaus in quantum convolutional neural networks, Phys. Rev. X **11**, 041011 (2021).
[27] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, Quantum-assisted quantum compiling, Quantum **3**, 140 (2019).
[28] C. Zhao and X.-S. Gao, Analyzing the barren plateau phenomenon in training quantum neural networks with the ZX-calculus, Quantum **5**, 466 (2021).
[29] Z. Liu, L.-W. Yu, L.-M. Duan, and D.-L. Deng, The presence and absence of barren plateaus in tensor-network based machine learning, Phys. Rev. Lett. **129**, 270501 (2022).
[30] Q. Miao and T. Barthel, Isometric tensor network optimization for extensive Hamiltonians is free of barren plateaus, Phys. Rev. A **109**, L050402 (2024).
[31] L. Monbroussou, E. Z. Mamon, J. Landman, A. B. Grilo, R. Kukla, and E. Kashefi, Trainability and expressivity of Hamming-weight preserving quantum circuits for machine learning, Quantum **9**, 1745 (2025).
[32] S. Raj, I. Kerenidis, A. Shekhar, B. Wood, J. Dee, S. Chakrabarti, R. Chen, D. Herman, S. Hu, P. Minssen *et al.*, Quantum deep hedging, Quantum **7**, 1191 (2023).
[33] E. Fontana, D. Herman, S. Chakrabarti, N. Kumar, R. Yalovetzky, J. Heredge, S. H. Sureshbabu, and M. Pistoia, Characterizing barren plateaus in quantum ansätze with the adjoint representation, Nat. Commun. **15**, 7171 (2024).
[34] N. L. Diaz, D. García-Martín, S. Kazi, M. Larocca, and M. Cerezo, Showcasing a barren plateau theory beyond the dynamical Lie algebra, arXiv:2310.11505.
[35] M. T. West, J. Heredge, M. Sevior, and M. Usman, Provably trainable rotationally equivariant quantum machine learning, PRX Quantum **5**, 030320 (2024).
[36] K. Zhang, L. Liu, M.-H. Hsieh, and D. Tao, in *Advances in Neural Information Processing Systems* (2022).
[37] C.-Y. Park and N. Killoran, Hamiltonian variational ansatz without barren plateaus, Quantum **8**, 1239 (2024).
[38] Y. Wang, B. Qi, C. Ferrie, and D. Dong, Trainability enhancement of parameterized quantum circuits via reduced-domain parameter initialization, Phys. Rev. Appl. **22**, 054005 (2024).
[39] M. Larocca, F. Sauvage, F. M. Sbahi, G. Verdon, P. J. Coles, and M. Cerezo, Group-invariant quantum machine learning, PRX Quantum **3**, 030341 (2022).
[40] J. J. Meyer, M. Mularski, E. Gil-Fuster, A. A. Mele, F. Arzani, A. Wilms, and J. Eisert, Exploiting symmetry in variational quantum machine learning, PRX Quantum **4**, 010328 (2023).
[41] A. Skolik, M. Cattelan, S. Yarkoni, T. Bäck, and V. Dunjko, Equivariant quantum circuits for learning on weighted graphs, npj Quantum Inf. **9**, 47 (2023).
[42] M. Ragone, Q. T. Nguyen, L. Schatzki, P. Braccia, M. Larocca, F. Sauvage, P. J. Coles, and M. Cerezo, Representation theory for geometric quantum machine learning, arXiv:2210.07980.
[43] Q. T. Nguyen, L. Schatzki, P. Braccia, M. Ragone, P. J. Coles, F. Sauvage, M. Larocca, and M. Cerezo, Theory for equivariant quantum neural networks, PRX Quantum **5**, 020328 (2024).
[44] L. Schatzki, M. Larocca, Q. T. Nguyen, F. Sauvage, and M. Cerezo, Theoretical guarantees for permutation-equivariant quantum neural networks, npj Quantum Inf. **10**, 12 (2024).
[45] M. Kieferova, O. M. Carlos, and N. Wiebe, Quantum generative training using Rényi divergences, arXiv:2106.09567.
[46] M. S. Rudolph, S. Lerch, S. Thanasilp, O. Kiss, O. Shaya, S. Vallecorsa, M. Grossi, and Z. Holmes, Trainability barriers and opportunities in quantum generative modeling, npj Quantum Inf. **10**, 116 (2024).
[47] A. Letcher, S. Woerner, and C. Zoufal, Tight and efficient gradient bounds for parameterized quantum circuits, Quantum **8**, 1484 (2024).

[48] M. S. Rudolph, J. Miller, D. Motlagh, J. Chen, A. Acharya, and A. Perdomo-Ortiz, Synergistic pretraining of parametrized quantum circuits via tensor networks, Nat. Commun. **14**, 8367 (2023).

[49] M. Cerezo, M. Larocca, D. García-Martín, N. L. Diaz, P. Braccia, E. Fontana, M. S. Rudolph, P. Bermejo, A. Ijaz, S. Thanasilp *et al.*, Does provable absence of barren plateaus imply classical simulability? Nat. Commun. **16**, 7907 (2025).

[50] I. Cong, S. Choi, and M. D. Lukin, Quantum convolutional neural networks, Nat. Phys. **15**, 1273 (2019).

[51] T. Hur, L. Kim, and D. K. Park, Quantum convolutional neural network for classical data classification, Quantum Mach. Intell. **4**, 3 (2022).

[52] S. Oh, J. Choi, and J. Kim, in *2020 International Conference on Information and Communication Technology Convergence (ICTC)* (IEEE, 2020), pp. 236–239.

[53] H. Baek, W. J. Yun, and J. Kim, Scalable quantum convolutional neural networks, arXiv:2209.12372.

[54] L.-H. Gong, J.-J. Pei, T.-F. Zhang, and N.-R. Zhou, Quantum convolutional neural network based on variational quantum circuits, Opt. Commun. **550**, 129993 (2024).

[55] J. Kim, J. Huh, and D. K. Park, Classical-to-quantum convolutional neural network transfer learning, Neurocomputing **555**, 126643 (2023).

[56] W. Li, P.-C. Chu, G.-Z. Liu, Y.-B. Tian, T.-H. Qiu, and S.-M. Wang, An image classification algorithm based on hybrid quantum classical convolutional neural network, Quantum Eng. **2022**, 5701479 (2022).

[57] D. Bokhan, A. S. Mastiukova, A. S. Boev, D. N. Trubnikov, and A. K. Fedorov, Multiclass classification using quantum convolutional neural networks with hybrid quantum-classical learning, Front. Phys. **10**, 1069985 (2022).

[58] E. Ovalle-Magallanes, D. E. Alvarado-Carrillo, J. G. Avina-Cervantes, I. Cruz-Aceves, and J. Ruiz-Pinales, Quantum angle encoding with learnable rotation applied to quantum–classical convolutional neural networks, Appl. Soft. Comput. **141**, 110307 (2023).

[59] S. Y. Chang, M. Grossi, B. Le Saux, and S. Vallecorsa, in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)* (2023), Vol. 01, pp. 229–235.

[60] F. Fan, Y. Shi, T. Guggemos, and X. X. Zhu, Hybrid quantum-classical convolutional neural network model for image classification, IEEE Trans. Neural Netw. Learn. Syst. **35**, 18145 (2024).

[61] Y. Li, R.-G. Zhou, R. Xu, J. Luo, and W. Hu, A quantum deep convolutional neural network for image recognition, Quantum Sci. Technol. **5**, 044003 (2020).

[62] Y. Zeng, H. Wang, J. He, Q. Huang, and S. Chang, A multi-classification hybrid quantum neural network using an all-qubit multi-observable measurement strategy, Entropy **24**, 394 (2022).

[63] A. Sebastianelli, D. A. Zaidenberg, D. Spiller, B. Le Saux, and S. L. Ullo, On circuit-based hybrid quantum neural networks for remote sensing imagery classification, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **15**, 565 (2022).

[64] A. Chalumuri, R. Kune, S. Kannan, and B. S. Manoj, Quantum–classical image processing for scene classification, IEEE Sens. Lett. **6**, 1 (2022).

[65] X. Zhang, J. Xia, X. Tan, X. Zhou, and T. Wang, Polsar image classification via learned superpixels and QCNN integrating color features, Remote Sens. **11**, 1831 (2019).

[66] Y. Matsumoto, R. Natsuaki, and A. Hirose, Full-learning rotational quaternion convolutional neural networks and confluence of differently represented data for Polsar land classification, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **15**, 2914 (2022).

[67] S. Y. Chang, B. Le Saux, S. Vallecorsa, and M. Grossi, in *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium* (IEEE, 2022), pp. 4907–4910.

[68] Z. N. Aldoski and C. Koren, Impact of traffic sign diversity on autonomous vehicles: A literature review, Period. Polytechn. Transp. Eng. **51**, 338 (2023).

[69] M. A. Khan, H. Park, and J. Chae, A lightweight convolutional neural network (CNN) architecture for traffic sign recognition in urban road networks, Electronics **12**, 1802 (2023).

[70] S. Y.-C. Chen, T.-C. Wei, C. Zhang, H. Yu, and S. Yoo, Quantum convolutional neural networks for high energy physics data analysis, Phys. Rev. Res. **4**, 013231 (2022).

[71] S. Y.-C. Chen, T.-C. Wei, C. Zhang, H. Yu, and S. Yoo, Hybrid quantum-classical graph convolutional network, arXiv:2101.06189.

[72] A. Delgado, K. E. Hamilton, J.-R. Vlimant, D. Magano, Y. Omar, P. Bargassa, A. Francis, A. Gianelle, L. Sestini, D. Lucchesi *et al.*, Quantum computing for data analysis in high energy physics, arXiv:2203.08805.

[73] S. Li, M. S. Salek, Y. Wang, and M. Chowdhury, Quantum-inspired activation functions in the convolutional neural network, arXiv:2404.05901.

[74] S. Monaco, O. Kiss, A. Mandarino, S. Vallecorsa, and M. Grossi, Quantum phase detection generalization from marginal quantum neural network models, Phys. Rev. B **107**, L081105 (2023).

[75] M. Cea, M. Grossi, S. Monaco, E. Rico, L. Tagliacozzo, and S. Vallecorsa, Exploring the phase diagram of the quantum one-dimensional ANNNI model, arXiv:2402.11022.

[76] A. J. Ferreira-Martins, L. Silva, A. Palhares, R. Pereira, D. O. Soares-Pinto, R. Chaves, and A. Canabarro, Detecting quantum phase transitions in a frustrated spin chain via transfer learning of a quantum classifier algorithm, Phys. Rev. A **109**, 052623 (2024).

[77] M. C. Caro, H.-Y. Huang, M. Cerezo, K. Sharma, A. Sornborger, L. Cincio, and P. J. Coles, Generalization in quantum machine learning from few training data, Nat. Commun. **13**, eid 4919 (2022).

[78] E. Gil-Fuster, J. Eisert, and C. Bravo-Prieto, Understanding quantum machine learning also requires rethinking generalization, Nat. Commun. **15**, 2277 (2024).

[79] I. MacCormack, C. Delaney, A. Galda, N. Aggarwal, and P. Narang, Branching quantum convolutional neural networks, Phys. Rev. Res. **4**, 013117 (2022).

[80] J. Herrmann, S. M. Llima, A. Remm, P. Zapletal, N. A. McMahon, C. Scarato, F. Swiadek, C. K. Andersen, C. Hellings, S. Krinner *et al.*, Realizing quantum convolutional neural networks on a superconducting quantum processor to recognize quantum phases, Nat. Commun. **13**, 1 (2022).

[81] N. Wrobel, A. Baul, K.-M. Tam, and J. Moreno, Detecting quantum critical points of correlated systems by quantum convolutional neural network using data from variational quantum eigensolver, Quantum Rep. **4**, 574 (2022).
[82] Y.-J. Liu, A. Smith, M. Knap, and F. Pollmann, Model-independent learning of quantum phases of matter with quantum convolutional neural networks, Phys. Rev. Lett. **130**, 220603 (2023).
[83] P. Zapletal, N. A. McMahon, and M. J. Hartmann, Error-tolerant quantum convolutional neural networks for symmetry-protected topological phases, Phys. Rev. Res. **6**, 033111 (2024).
[84] G. Acampora, A. Ambainis, N. Ares, L. Banchi, P. Bhardwaj, D. Binosi, G. A. D. Briggs, T. Calarco, V. Dunjko, J. Eisert *et al.*, Quantum computing and artificial intelligence: Status and perspectives, arXiv:2505.23860.
[85] M. Doosti, P. Wallden, C. B. Hamill, R. Hankache, O. T. Brown, and C. Heunen, A brief review of quantum machine learning for financial services, arXiv:2407.12618.
[86] W. Guan, G. Perdue, A. Pesah, M. Schuld, K. Terashi, S. Vallecorsa, and J.-R. Vlimant, Quantum machine learning in high energy physics, Mach. Learn.: Sci. Technol. **2**, 011003 (2021).
[87] R. S. Gupta, C. E. Wood, T. Engstrom, J. D. Pole, and S. Shrapnel, A systematic review of quantum machine learning for digital health, npj Digital Med. **8**, 237 (2025).
[88] A. Angrisani, A. Schmidhuber, M. S. Rudolph, M. Cerezo, Z. Holmes, and H.-Y. Huang, Classically estimating observables of noiseless quantum circuits, Phys. Rev. Lett. **135**, 170602 (2025).
[89] That is, the results of Ref. [88] imply that randomly initialized QCNNs are classically simulable, but this does not imply that the simulation will either be efficient or faithful throughout the whole training process.
[90] E. Fontana, M. S. Rudolph, R. Duncan, I. Rungger, and C. Cîrstoiu, Classical simulations of noisy variational quantum circuits, npj Quantum Inf. **11**, 1 (2025).
[91] M. S. Rudolph, E. Fontana, Z. Holmes, and L. Cincio, Classical surrogate simulation of quantum systems with LOWESA, arXiv:2308.09109.
[92] R. Orús, A practical introduction to tensor networks: Matrix product states and projected entangled pair states, Ann. Phys. **349**, 117 (2014).
[93] H.-Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, Nat. Phys. **16**, 1050 (2020).
[94] A. Elben, S. T. Flammia, H.-Y. Huang, R. Kueng, J. Preskill, B. Vermersch, and P. Zoller, The randomized measurement toolbox, Nat. Rev. Phys. **5**, 9 (2023).
[95] F. Sauvage and M. Larocca, Classical shadows with symmetries, arXiv:2408.05279.
[96] H.-Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, and J. R. McClean, Power of data in quantum machine learning, Nat. Commun. **12**, 1 (2021).
[97] H.-Y. Huang, Y. Liu, M. Broughton, I. Kim, A. Anshu, Z. Landau, and J. R. McClean, in *Proceedings of the 56th Annual ACM Symposium on Theory of Computing* (2024), pp. 1343–1351.
[98] S. Jerbi, C. Gyurik, S. C. Marshall, R. Molteni, and V. Dunjko, Shadows of quantum machine learning, Nat. Commun. **15**, 5676 (2024).
[99] It is clear that QCNNs with trivial input states such as the all zero-state are classically simulable via tensor networks. Here we instead wonder if their action is simulable even if the initial state does not admit a simple classical representation.
[100] S. Aaronson and D. Gottesman, Improved simulation of stabilizer circuits, Phys. Rev. A **70**, 052328 (2004).
[101] R. Jozsa and A. Miyake, Matchgates and classical simulation of quantum circuits, Proc. R. Soc. A: Math. Phys. Eng. Sci. **464**, 3089 (2008).
[102] H.-Y. Huang, S. Chen, and J. Preskill, Learning to predict arbitrary quantum processes, PRX Quantum **4**, 040337 (2023).
[103] H.-Y. Huang, Y. Tong, D. Fang, and Y. Su, Learning many-body Hamiltonians with Heisenberg-limited scaling, Phys. Rev. Lett. **130**, 200403 (2023).
[104] M. L. Goh, M. Larocca, L. Cincio, M. Cerezo, and F. Sauvage, Lie-algebraic classical simulations for quantum computing, Phys. Rev. Res. **7**, 033266 (2025).
[105] C. Dankert, R. Cleve, J. Emerson, and E. Livine, Exact and approximate unitary 2-designs and their application to fidelity estimation, Phys. Rev. A **80**, 012304 (2009).
[106] A. W. Harrow and R. A. Low, Random quantum circuits are approximate 2-designs, Commun. Math. Phys. **291**, 257 (2009).
[107] F. G. S. L. Brandao, A. W. Harrow, and M. Horodecki, Local random quantum circuits are approximate polynomial-designs, Commun. Math. Phys. **346**, 397 (2016).
[108] N. Hunter-Jones, Unitary designs from statistical mechanics in random quantum circuits, arXiv:1905.12053.
[109] J. Haferkamp, Random quantum circuits are approximate unitary t-designs in depth $O\left(nt^{5+o(1)}\right)$, Quantum **6**, 795 (2022).
[110] T. Schuster, J. Haferkamp, and H.-Y. Huang, Random unitaries in extremely low depth, Science **389**, 92 (2025).
[111] B. Collins and P. Śniady, Integration with respect to the Haar measure on unitary, orthogonal and symplectic group, Commun. Math. Phys. **264**, 773 (2006).
[112] Z. Puchała and J. A. Miszczak, Symbolic integration with respect to the haar measure on the unitary groups, Bull. Pol. Acad. Sci. Tech. Sci. **65**, 21 (2017).
[113] A. A. Mele, Introduction to Haar measure tools in quantum information: A beginner's tutorial, Quantum **8**, 1340 (2024).
[114] L. Leone, S. F. E. Oliviero, L. Cincio, and M. Cerezo, On the practical usefulness of the hardware efficient ansatz, Quantum **8**, 1395 (2024).
[115] S. Thanaslip, S. Wang, N. A. Nghiem, P. J. Coles, and M. Cerezo, Subtleties in the trainability of quantum machine learning models, Quantum Mach. Intell. **5**, 21 (2023).
[116] P. Braccia, P. Bermejo, L. Cincio, and M. Cerezo, Computing exact moments of local random quantum circuits via tensor networks, Quantum Mach. Intell. **6**, 54 (2024).
[117] S. Sim, P. D. Johnson, and A. Aspuru-Guzik, Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms, Adv. Quantum Technol. **2**, 1900070 (2019).

[118] A. Kitazawa, K. Nomura, and K. Okamoto, Phase diagram of $S = 1$ bond-alternating XXZ chains, Phys. Rev. Lett. **76**, 4038 (1996).
[119] F. D. M. Haldane, Nonlinear field theory of large-spin Heisenberg antiferromagnets: Semiclassically quantized solitons of the one-dimensional easy-axis Néel state, Phys. Rev. Lett. **50**, 1153 (1983).
[120] R. J. Elliott, Phenomenological discussion of magnetic ordering in the heavy rare-earth metals, Phys. Rev. **124**, 346 (1961).
[121] M. Suzuki, Relationship among exactly soluble models of critical phenomena. I: 2D Ising model, dimer problem and the generalized XY-model, Prog. Theor. Phys. **46**, 1337 (1971).
[122] M. Fishman, S. White, and E. Stoudenmire, The ITensor software library for tensor network calculations, SciPost Phys. Codebases, 004 (2022).
[123] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, Julia: A fresh approach to numerical computing, SIAM Rev. **59**, 65 (2017).
[124] D. C. Liu and J. Nocedal, On the limited memory BFGS method for large scale optimization, Math. Program. **45**, 503 (1989).
[125] S. Thanasilp, S. Wang, N. A. Nghiem, P. Coles, and M. Cerezo, Subtleties in the trainability of quantum machine learning models, Quantum Mach. Intell. **5**, 21 (2023).
[126] J. C. Napp, R. L. La Placa, A. M. Dalzell, F. G. S. L. Brandao, and A. W. Harrow, Efficient classical simulation of random shallow 2D quantum circuits, Phys. Rev. X **12**, 021021 (2022).
[127] D. García-Martín, M. Larocca, and M. Cerezo, Quantum neural networks form Gaussian processes, Nat. Phys. **21**, 1153 (2025).
[128] A. M. Dalzell, N. Hunter-Jones, and F. G. Brandão, Random quantum circuits transform local noise into global white noise, Commun. Math. Phys. **405**, 78 (2024).
[129] J. Napp, Quantifying the barren plateau phenomenon for a model of unstructured variational ansätze, arXiv:2203.06174.
[130] A. M. Dalzell, N. Hunter-Jones, and F. G. S. L. Brandão, Random quantum circuits anticoncentrate in log depth, PRX Quantum **3**, 010333 (2022).
[131] D. Aharonov, X. Gao, Z. Landau, Y. Liu, and U. Vazirani, in *Proceedings of the 55th Annual ACM Symposium on Theory of Computing* (2023), p. 945.
[132] J. M. Chow, J. M. Gambetta, A. D. Córcoles, S. T. Merkel, J. A. Smolin, C. Rigetti, S. Poletto, G. A. Keefe, M. B. Rothwell, J. R. Rozen *et al.*, Universal quantum gate set approaching fault-tolerant thresholds with superconducting qubits, Phys. Rev. Lett. **109**, 060501 (2012).
[133] F. J. Schreiber, J. Eisert, and J. J. Meyer, Classical surrogates for quantum learning models, Phys. Rev. Lett. **131**, 100803 (2023).
[134] H.-Y. Huang, R. Kueng, G. Torlai, V. V. Albert, and J. Preskill, Provably efficient machine learning for quantum many-body problems, Science **377**, eabk3333 (2022).
[135] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Ann. Phys. **326**, 96 (2011).
[136] S. R. White, Density matrix formulation for quantum renormalization groups, Phys. Rev. Lett. **69**, 2863 (1992).
[137] S. Holtz, T. Rohwedder, and R. Schneider, The alternating linear scheme for tensor optimization in the tensor train format, SIAM J. Sci. Comput. **34**, A683 (2012).
[138] D. P. Kingma and J. Ba, in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)* (2015).
[139] M. Duschenes, D. García-Martín, Z. Holmes, and M. Cerezo, Moments of quantum channel ensembles, arXiv:2511.12700.
[140] R. Horodecki, P. Horodecki, M. Horodecki, and K. Horodecki, Quantum entanglement, Rev. Mod. Phys. **81**, 865 (2009).