Learning Fourier series with parametrized quantum circuits

Dirk Heimann* and Hans Hohenfeld†
Robotics Research Group, University of Bremen, 28359 Bremen, Germany

Gunnar Schönhoff‡ and Elie Mounzer§
German Research Center for Artificial Intelligence - Robotics Innovation Center (DFKI RIC), 28359 Bremen, Germany

Frank Kirchner¶
Robotics Research Group, University of Bremen, 28359 Bremen, Germany and
German Research Center for Artificial Intelligence - Robotics Innovation Center (DFKI RIC), 28359 Bremen, Germany

(Dated: May 21, 2025)

Variational quantum algorithms (VQAs) and their applications in the field of quantum machine learning through parametrized quantum circuits (PQCs) are thought to be one major way of leveraging noisy intermediate-scale quantum computing devices. However, differences in the performance of certain VQA architectures are often unclear since established best practices, as well as detailed studies, are missing. In this paper, we build upon the work by Schuld et al. [Phys. Rev. A 103, 032430 (2021)] and Vidal et al. [Front. Phys. 8, 297 (2020)] by comparing how well popular ansätze for PQCs learn different one-dimensional truncated Fourier series. We also examine dissipative quantum neural networks (dQNN) as introduced by Beer et al. [Nat. Commun. 11, 808 (2020)] and propose a data reupload structure for dQNNs to increase their capability for this regression task. By comparing the results for different PQC architectures, we can provide guidelines for designing efficient PQCs.

I. INTRODUCTION

Quantum machine learning (QML) is one of the areas of quantum computing that has attracted a lot of attention in recent years [1–7], especially because QML algorithms seem to be a promising candidate for making use of noisy intermediate-scale quantum computing (NISQ) [8, 9]. There are already a variety of prototypical practical applications of QML, ranging from the classification of radiological [10] or car images [11] to other industrial use cases [12], reinforcement learning environments [13] and robot pathfinding [14]. Among the methods investigated for QML are variational quantum algorithms (VQAs) which make use of parametrized quantum circuits (PQCs) in a hybrid quantum-classical optimization loop [15]. The term *quantum neural networks* is often used synonymously with this application of PQCs for QML tasks [16, 17].

While improving the performance and scalability of VQAs is an area of ongoing research (see, for example, Refs. [18–20]), and the question of whether QML algorithms can eventually outperform classical algorithms is still open (see, for example, Refs. [15, 21, 22]), significant steps have already been made [23–25]. For classification tasks on classical data, Ref. [26] shows empirically that evidence for hybrid QML algorithms outperforming classical neural networks is still missing. Different circuit architectures or *ansätze* for PQCs have been proposed and used, ranging from hardware-efficient circuits [27] and the quantum alternating operator ansatz (QAOA) [28] to dissipative quantum neural networks (dQNNs) [29], which were successfully implemented on a six-qubit superconducting processor [30]. Additionally, data reupload has been shown to improve VQA performance significantly when encoding classical input data [31–33].

Steps to compare different PQCs were done by several groups, for example in Ref. [33], where a methodology based on Fourier series and coefficients was developed to compare the expressiveness of PQCs; in Ref. [34], where architectures with amplitude-encoding and varying circuit depths were compared on different benchmark classification tasks; in Refs. [35], [36], and [37], where selected architectures were evaluated numerically regarding expressibility and entangling capability; in Ref. [38], where the entanglement and the entangling speed of different circuits were compared; in Ref. [39], where effective capacity and effective dimension were analyzed for different types of PQCs, including dQNNs; and in Ref. [40], where the work from Ref. [33] was investigated in more detail for single-qubit circuits. Furthermore, the authors of Ref. [21] investigate the performance of classical surrogates for layered PQCs. While this fact indicates that it may be difficult to find advantages of PQCs in comparison to classical machine learning on classical data, this question is not settled and, for example, the authors of Ref. [25] show that so-called flipped quantum models exist, which have a provable learning advantage over fully classical learners despite having a classical surrogate model. Also, as laid out in, for example, Refs. [41, 42], mathematically rigorous bounds exist that link model complexity of PQCs with their ability to generalize, pro-

* dirk.heimann@uni-bremen.de
† hans.hohenfeld@uni-bremen.de
‡ gunnar.schoenhoff@dfki.de
§ elie.mounzer@dfki.de
¶ frank.kirchner@dfki.de

viding a starting point for quantum learning theory.

Despite all this work, general established best practices for the use of PQC architectures are still missing and the construction of efficient, trainable PQCs is an ongoing research topic (see, for example, Refs. [43, 44]). Existing methods for performance evaluation of PQC ansätze are not enough to fully describe the ansatz performance since they do not take into account the combination of expressiveness and trainability that is needed in practical use cases. Furthermore, as we show in Sec. IV A only evaluating the Fourier coefficients resulting from the inverse Fourier transform of an ansatz evaluation with sampled parameters also misses differences between certain PQC architectures. In this paper, we tackle this problem by extending and using the numerical methodology presented in Ref. [33] to lay out a framework to both more strictly and practically compare the performance of small circuit elements as well as complete PQC ansätze.

In particular, our contributions are the following:

- We extend the numerical results by Schuld et al. [33], in particular by showing that PQCs with the same sampled Fourier coefficients can have different training results.
- We propose the new measure of *learning capability* to analyze PQC ansatz performance.
- We compare different types of ansätze regarding this measure.
- We define and evaluate a data reupload structure for dQNNs.

In addition to other challenges [45], barren plateaus remain a major issue for the practical use of VQAs [46–52]. For example, in Ref. [52], the authors show that PQCs with full rank dynamical Lie algebras exhibit barren plateaus because, as the number of layers increases, they become ϵ-approximate 2-designs. In this paper, we compare architectures which all tend to exhibit barren plateaus for larger depth or global cost functions as shown in Appendix A, especially Corollary. A.5. The architectures utilize local cost functions and can exhibit barren plateaus if the depth is of polynomial order in the number of qubits [47]. Our analysis gives guidelines to minimize layers, qubits, gates, and parameters which can be used to reduce the depth. Thus, our work helps to reduce the effect of barren plateaus.

These guidelines are drawn from numerically investigating how well architectures can learn Fourier series, which can, in principle, approximate arbitrary functions up to a certain precision. Hence, we conclude that the guidelines for PQCs obtained from these results can be transferred to more general learning tasks. However, this assumption may not hold universally.

The sections of this paper are organized as follows: In Sec. II, we summarize some of the existing work on PQC evaluation. In Sec. III, we present our approach for the measure of learning capability and describe both the evaluated ansätze and the numerical setup. We then show our main results in Sec. IV and discuss the implications in Sec. V.

II. PRIOR WORK

Investigating the model complexity of PQCs is a topic of active research. There are different angles and approaches one can take here. One important and active research field links model complexity to the generalization behavior [17, 18, 53–57], with only some works involving data reupload to obtain generalization bounds depending on the data reupload encoding scheme (see, for example, Ref. [58]). Interestingly, Ref. [59] demonstrates that, depending on the number of training samples, generalization is possible despite overfitting to training data. Furthermore, Ref. [60] recently questioned the applicability of uniform generalization bounds by constructing a randomized supervised learning example, which reveals the memorization capability of PQCs. More work exists which investigates a model's complexity by assessing its capacity to memorize random data. Reference [61] compares the memorization capacity of quantum neural networks with classical neural networks based on the information-theoretic ideas introduced in Ref. [62] for classical perceptrons and generalized to neural networks in Ref. [63]. This notion got extended to storing quantum states in [64].

A. Expressiveness and entanglement

In the following subsection, we present works in more detail, which investigate model complexity of PQCs in terms of their expressiveness and use this to reason about the performance of PQCs on QML tasks. An important approach in this area is the sampling of different parameters for a PQC and the calculation of the Kullback-Leibler divergence [65] of the estimated fidelity distribution in comparison with the Haar-distributed ensemble [35, 66]. This gives a description of the expressiveness in the form that a small deviation from the Haar distribution means that all of the corresponding Hilbert space is equally reachable. However, this analysis does neither take into account the effect that input encoding has on the expressiveness of a circuit—it plays an essential role when classifying classical data as was proven in [33]—nor the trainability, which is crucial for practical use cases.

Other measures in this realm are the expressive power and effective dimension. Reference [67] investigates the expressive power of PQCs as generative models for probability distributions in terms of the entanglement entropy. In Ref. [68], the authors look at the effect of repeating data-first encoding by introducing further qubits on learning physical observables and analyzing the expressiveness of PQCs via the Hilbert space dimension the

quantum circuits act on. Reference [69] analyzes PQCs based on the effective dimension, which depends on the Fisher information matrix, and the parameter dimension, which the authors numerically estimate by evaluating the effective dimension for random initializations.

In addition to the expressiveness, entanglement capability is another measure that is used to describe the performance of PQCs [35–37]. Results from Ref. [37] indicate that entanglement and expressiveness are only weakly correlated, and the authors of Ref. [48] have found that entanglement can hinder trainability, e.g., by inducing barren plateaus. While this could lead one to the conclusion that a PQC should have only a limited amount of entanglement, a certain amount is considered as needed in order for quantum advantage to be possible. The works that investigate PQC performance use a variety of ways to construct entangling layers, ranging from linear and cyclic entangling with zero parameters [33], one parameter [35], or three parameters [70] to the layer-dependent strong entanglement described in Ref. [71].

B. Expressiveness regarding Fourier series

The current understanding of PQC evaluations representing Fourier series as described in Ref. [33] is laid out in the following. The function $f(x)$ is defined as the expectation value of the result of several PQC evaluations:

$$f_\Theta(x) = \langle 0| U^\dagger(x,\Theta) M U(x,\Theta) |0\rangle \tag{1}$$

where Θ is a set of all trainable parameters, x is a classical data input, U is an arbitrary unitary, and M is a measurement operator. For the usual *layered* ansatz, the unitary U takes the form

$$U(x) = W_L S(x) W_{L-1} ... W_1 S(x) W_0 \tag{2}$$

where the circuit is of depth L, S is the data-upload unitary, and $W_l := W_l(\boldsymbol{\theta}_l)$ are trainable unitaries with $\boldsymbol{\theta}_l \subset \Theta$ (see Fig. 1). The result proven in Ref. [33] is that this model can be described by a partial real-valued Fourier series:

$$f(x) = \sum_{\omega\in\Omega} c_\omega e^{i\omega x} \tag{3}$$

where the Fourier coefficients $c_\omega = \overline{c}_{-\omega}$ are determined by the trainable unitaries and the measurement operator while the available frequency spectrum Ω is determined by the data (re-)upload operators. The frequencies ω are given by all possible sums of eigenvalues of the generating Hamiltonian H of the input encoding $S(x) = e^{-ixH}$, such that $\Omega = [-\omega, ..., 0, ..., \omega]$.

As an example, in the case of one layer and a unitary $U(x) = W_1 e^{-i\frac{x}{2}H} W_0$, the resulting model is a simple sinusoidal function if a Pauli-matrix with eigenvalues $\{\lambda_0, \lambda_1\} = \{+1, -1\}$ is used as generator:

$$f(x) = c_0 + 2|c_1| \cos(x - \arg(c_1)) \tag{4}$$

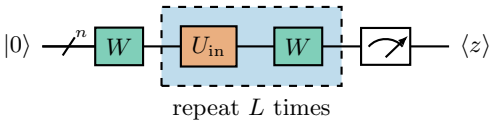


FIG. 1: General circuit layout for layered PQCs ansätze with n qubits, L layers and one W in the zero layer [see Eq. (2)]. The unitary gates W depend on trainable parameters and U_{in} encodes the input x.

with $c_0 = \sum_{i,i',j=0}^{1} \overline{W}_{0j} \overline{W}_{ji'} M_{i'i} W_{ij} W_{j0}$ and $c_1 = \sum_{i,i'=0}^{1} \overline{W}_{01} \overline{W}_{1i'} M_{i'i} W_{i0} W_{00}$ where $\overline{W}_{ji}$ denotes the complex conjugate of W_{ji} and the layer index of the unitaries W is omitted because it follows from the order.

Using the multi-index $\boldsymbol{j_l}$ over n qubits for layer l, the sum over eigenvalues $\Lambda_{\boldsymbol{j_l}} = \sum_{q=1}^{n} \lambda_{ql}$, and the Einstein sum convention for indices, one finds that for n qubits and L layers the expectation value can be written as:

$$\begin{aligned} f(x) =& e^{i\frac{x}{2}(\Lambda_{\boldsymbol{k_1}} + \cdots + \Lambda_{\boldsymbol{k_L}} - \Lambda_{\boldsymbol{j_1}} - \cdots - \Lambda_{\boldsymbol{j_L}})} \\ & \times M^{\boldsymbol{i'}}_{\boldsymbol{i}} \overline{W}^{\boldsymbol{k_L}}_{\boldsymbol{i'}} \cdots \overline{W}^{\boldsymbol{0}}_{\boldsymbol{k_1}} W^{\boldsymbol{i}}_{\boldsymbol{j_L}} \cdots W^{\boldsymbol{j_1}}_{\boldsymbol{0}} \end{aligned} \tag{5}$$

The detailed calculation is documented in Example 6 in Appendix K.

The observation that evaluations of PQCs yield Fourier series can be used to describe the expressiveness of the model via the available frequency range as well as the values of Fourier coefficients that can be reached. The latter one can be accessed via the inverse Fourier transform. For repeated single-qubit data encoding, i.e., the case where $S(x)$ is not changed throughout the circuit—as is often done in the literature for classical input encoding and also in this work—the size K of the available independent nonzero frequencies is limited by

$$K = nL, \tag{6}$$

where L is the number of layers in the PQC and n is the number of qubits; this means that the frequency spectrum is limited by the total number of repetitions of the data-encoding gate. Furthermore, $2K+1$ real-valued parameters are necessary to express K nonzero complex coefficients and one real valued coefficient c_0. Hence, one needs at least $2K+1$ parameters in the gates that form the trainable unitaries W of Eq. (2) to be able to have nonzero coefficients for all accessible frequencies from 0 to ω_K.

Several works take advantage of the fact that PQCs represent Fourier series. For example, Ref. [72] proposes an error mitigation technique based on Fourier coefficients and frequencies. Additionally, Ref. [73] shows that cost functions of training a QML model can be efficiently classically computed if the number of parameters scales polynomial in circuit depth. Reference [74] uses the training of Fourier series to test different PQC architecture adjustments and define PQCs which fit certain Fourier

series better more expressive. Reference [75] define residual connections in quantum neural networks and show that these increase the number of generated frequencies and the flexibility in adjusting the Fourier coefficients. While most works focus on single-dimensional Fourier series, Ref. [76] uses multidimensional truncated Fourier series to analyze the input encoding scheme and determine which PQCs provide enough degrees of freedom for fitting these functions.

In the following, we use the Fourier series character of PQC evaluations to test which ansätze provide a richer class of trainable functions. ´

III. METHODS AND SETUP

A. Learning capability

To improve the understanding of PQC performance, we extend the methodology first described in Ref. [33] as laid out in the following. The learned model function f_Θ that results from training a PQC model [see Eq. (1)] to approximate a given Fourier function g is a measure of both the trainability and the expressiveness of the corresponding circuit, given a choice of training procedure. In our studies, we use the corresponding loss value given by the mean-squared error (MSE)

$$\varepsilon_g = \frac{1}{|X|} \sum_{x \in X} \left[f_\Theta(x) - g(x) \right]^2 \tag{7}$$

where $|X|$ is the size of the test set. The mean squared error is used here since it is a common metric in machine learning that is suitable to determine the difference between the results of the PQC model and the Fourier function.

We use a set of functions G_d which contains $|G_d|$ randomly sampled, normalized Fourier functions, i.e., trigonometric polynomials, of degree d and define learning capability as the average over the final individual validation losses:

$$\mu_d = \frac{1}{|G_d|} \sum_{g \in G_d} \varepsilon_g. \tag{8}$$

This measure enables the practical analysis of the performance of different PQC architectures that include data encoding; this is done by using the insight that these architectures can (only) learn Fourier functions to investigate expressiveness and trainability. We take the average over a set of Fourier functions G_d to address statistical variance in the training procedure. This way, an ansatz with a lower numerical value for μ_d is on average better suited to learn Fourier series with a determinable confidence interval than an ansatz with a higher numerical value μ_d.

The MSE in Eq. (7) is a common and natural choice as a loss function for regression problems. It connects the learning task considered in this work to the framework of maximum likelihood estimation, providing an objective function proportional to the negative log-likelihood, under the assumption of independent and normally distributed residuals [77–79]. In principle, ε_g could be replaced with a different loss, if a similar analysis would be considered on a different problem class, e.g., classification with generic data sets such as the ones suggested in Ref. [26] and a cross-entropy loss.

The size of the set of random Fourier functions needs to be large enough to sufficiently characterize the performance of a given circuit architecture on the problem set. We chose $|G_d| = 100$ as this produces reasonably tight confidence intervals to assume an unbiased estimate and allow for qualitative statements. Smaller sizes of G_d may be possible, but were not considered here; for considerably larger sizes, one would expect diminishing returns on the quality of the estimate of μ_d. In Appendix. H we provide a numerical comparison of the mean and confidence interval for different set sizes based on the first experiments presented in our Results section. The numerical comparison underlines that the choice of $|G_d| = 100$ leads to a reasonably accurate estimate of the mean while still being technically feasible.

An interesting question is if G_d can be limited to a potentially small number of specific functions that are a good representative of a real-world problem or class of real-world problems. This question is not straightforward to answer and relates to the field of neural architecture search [80] in classical machine learning, an active area of ongoing research. We discuss some aspects of this in Sec. V.

B. Evaluated architectures

We evaluate the learning capability of different PQC ansätze and change circuit properties systematically to reveal the effects of each part. More precisely, we analyze different ways of how to construct the trainable Ws in Fig. 1 and Eq. (2).

In the literature, the most commonly used entanglement gates without trainable parameters are CNOT (e.g., circuits 2, 11, and 15 in Ref. [35] as well as circuits in Ref. [33]) and CZ (e.g., circuits 9, 10, and 12 in Ref. [35] as well as circuits in Refs. [13, 14, 31, 46, 47, 50, 81]). In addition, controlled-R_X (CRX) (e.g., circuits 4, 6, 8, 14, 17, and 19 in Ref. [35] as well as circuits in Ref. [20]) or R_Z (e.g., circuits 3, 5, 7, 13, 16, and 18 in Ref. [35]) are often used as entanglement gates with one trainable parameter. We also test the CAN gate [82] because it provides the interesting case of three trainable parameters and bridges the gap to dQNN ansatz structures. This gate is composed of three two-qubit rotations, $\mathrm{CAN}(\boldsymbol{\theta}) = R_{XX}(\theta_0) R_{YY}(\theta_1) R_{ZZ}(\theta_2)$.

Besides the entangling gates, every unitary W contains

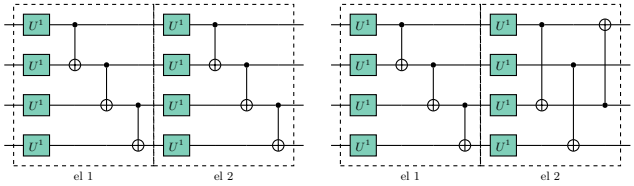


FIG. 2: Simple (left) and strong (right) entanglement structure for entanglement layer one and two depicted for generic single-qubit unitaries U^1 and CNOT as an example for a two-qubit unitary.

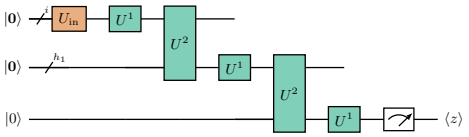
single-qubit gates. Different options range from single operations (e.g., R_Y in Ref. [47]) for more hardware-efficient ansätze, to two operations (e.g., $R_Y R_Z$ in Ref. [13]) and to general operations with three single gates (e.g., $R_Y R_Z R_Y$ in Ref. [33]). Thus, in this work, we use these typical single-qubit unitaries $U^1 \in \{R_Y, R_Y R_Z, R_Y R_Z R_Y\}$ together with two-qubit unitaries $U_{\text{ent}} \in \{\text{CZ}, \text{CNOT}, \text{CRX}, \text{CAN}\}$.

In Ref. [71], strongly entangling circuits were introduced to increase the entanglement in shallow circuits. This technique includes splitting each W in blocks, which we refer to as *entanglement layers*, where rotation gates are applied at the beginning of each block followed by controlled two-qubit gates. In our analysis, we choose the entanglement range for strong entanglement in each block to be equal to the number of the current block, i.e., if we use three entanglement layers, the first block contains ordinary controlled operations with control range 1, then the second block with range 2 and the third block with range 3. We contrast this ansatz of creating strong entanglement with a simpler one, which has several entanglement layers but keeps the control range of the entanglement equal to 1 for each block. The difference is depicted in Fig. 2.

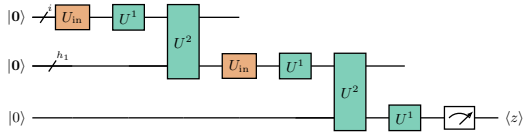
Another property which is interesting to analyze is the amount of entanglement gates per entanglement layer. If this amount is equal to the number of qubits, then we arrange the entanglement gates descending and connect the last with the first qubit. This method is commonly used (e.g., circuits 10, 13, 14, 18, and 19 in Ref. [35] as well as circuits in Refs. [13, 33]). We refer to it as *cyclic*. One can also use one entanglement gate less than the number of qubits, neglecting the last entanglement gate in the cyclic structure (e.g., circuits 2, 3, 4, 9, 16, and 17 in Ref. [35] as well as circuits in Refs. [46, 50]). We call this setup *linear* because it does not connect the last qubit with the first one in an entanglement structure with a range equal to one. This case is shown in Fig. 3.

As an example, if one chooses a simple, cyclic entanglement with one entanglement layer, CZ gates, and two rotation gates as a single-qubit unitary, then one arrives at the circuit ansatz chosen in Ref. [13]. Furthermore, we consider circuits that are similar to the alternating layer ansatz [47, 66] and the hardware-efficient ansatz [27, 66] by using the entanglement gates to first connect qubits

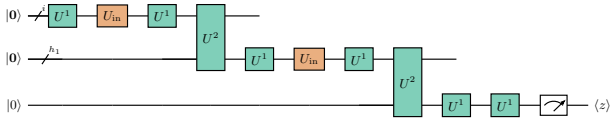
FIG. 3: Linear (left) and cyclic (right) entanglement style, with the CNOT as an example of a two-qubit unitary acting on the first and the last wire it overlaps with.



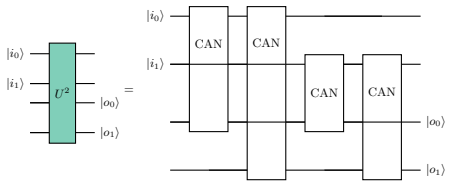
(a) dQNN circuit ansatz $[i, h_1, 1]$ for complex-valued data as described by Beer et. al. [29, 70] with i input qubits, one hidden layer with h_1 qubits, and one output qubit.



(b) dQNN circuit ansatz with included data reupload structure.



(c) dQNN ansatz with data reupload and $U^1 U_{\text{in}} U^1$ structure in each hidden layer.



(d) For the dQNN ansätze, the unitary gate U^2 is composed of CAN gates.

FIG. 4: Circuit representation and our modification of dQNN ansätze by Beer et. al. [29, 70].

with even and then qubits with uneven index within an entanglement layer.

In addition to the layered architectures, we include the proposed circuit structure for dQNNs in Refs. [29, 70] as an ansatz type. The original circuit of Ref. [70] is depicted in Fig. 4a. This architecture uses qubits as neurons in analogy to classical neural networks. Quantum perceptrons U_j^l propagate the information through a unitary U^2 consisting of two-qubit CAN gates which connect

each qubit of the previous layer $l-1$ with the jth neuron of layer l [see Fig. 4d]. In addition, parametrized single-qubit gates U^1 are applied to each qubit. Once the information is propagated, qubits of input and hidden layers are discarded, i.e., their values are not measured. The notation

$$[i, h_1, \cdots, h_H, \text{out}] \tag{9}$$

represents the number of input qubits i, the number of qubits of each hidden layer h_j for all H hidden layers, and the number of output qubits out which is equal to one in this work.

We consider classical data $\{x, g(x)\}_{x\in X}$ which is encoded via $U_{\text{in}}(x)$ creating the input state $\rho^{\text{in}} = \left|\phi_x^{\text{in}}\right\rangle\left\langle\phi_x^{\text{in}}\right|$ with $\left|\phi_x^{\text{in}}\right\rangle = (U_{\text{in}}(x)\left|0\right\rangle)^{\otimes i}$. The, possibly mixed, network's final output state can be expressed by

$$\begin{aligned}\rho_{x,\Theta}^{\text{out}} &= \text{tr}_{\text{in, hid}}\left(\mathcal{U}\rho^{\text{in}\prime}\mathcal{U}^\dagger\right) \\ &= \text{tr}_{\text{in, hid}}(U_{m_{\text{out}}}^{\text{out}}\ldots U_1^1 \\ &\qquad \times \rho^{\text{in}} \otimes \left|0\ldots0\right\rangle_{\text{hid, out}}\left\langle 0\ldots0\right| \\ &\qquad \times U_1^{1\dagger}\ldots U_{m_{\text{out}}}^{\text{out}\dagger}),\end{aligned} \tag{10}$$

where each quantum perceptron $U_j^l = U_j^l(\boldsymbol{\theta}_j^l)$ depends on a set of trainable parameters $\boldsymbol{\theta}_j^l$. In Fig. 10, this setup is called dQNN without zero layer (w/o zl).

Inspired by the layered structure of PQCs explained above, we propose and compare enhancements to the dQNN ansatz structure. First, we introduce a data reupload scheme by allowing data encoding on every qubit but the output one. The network's final output state is obtained as in Eq. (10) by using $\left|\phi_x^{\text{in}}\right\rangle = \left|0\right\rangle^{\otimes i}$ and the modified quantum perceptron $\tilde{U}_j^l(x, \boldsymbol{\theta}_j^l)$ which includes parametrized single-qubit operation U^1 applied after input encoding unitary $U_{\text{in}}(x)$ on each qubit of the previous layer $l-1$. This ansatz is depicted in Fig. 4b and referred to as dQNN reuploading without a zero layer (reup w/o zl). Second, to provide more degrees of freedom, we introduce single-qubit gates before each encoding gate which changes $\left|\phi_x^{\text{in}}\right\rangle$ to $\left|\phi^{\text{in}}\right\rangle = \left|0\right\rangle^{\otimes i}$ and the quantum perceptron to $\tilde{U}_j^l(x, \boldsymbol{\theta}_j^l)$ consisting of CAN gates linking all qubits of layer $l-1$ to the jth qubit of layer l and $U^1(\boldsymbol{\theta}_{j,1}^{l-1})U_{\text{in}}(x)U^1(\boldsymbol{\theta}_{j,0}^{l-1})$ applied on all qubits except the output qubit. This setup is referred to as dQNN reuploading with zero layer (reup w zl). This way, the architecture gets closer to the layered ansatz type. The full modification is depicted in Fig. 4c. If input unitaries are not repeated in between those parametrized single-qubit gates and the only input encoding is on the input layer $\left|\phi^{\text{in}}\right\rangle = u(\boldsymbol{\theta}_{i,1}^0)U_{\text{in}}(x)u(\boldsymbol{\theta}_{i,0}^0)\left|0\right\rangle^{\otimes i}$, then the setup is called dQNN with a zero layer (w zl) and the network's output state is obtained by Eq. (10) with quantum perceptrons $\tilde{U}_j^l(\boldsymbol{\theta}_j^l)$.

The final output of our circuits, Eq. (1), is given by the expectation value of an observable $f_\Theta(x) = \langle M\rangle_{\psi(x,\Theta)} = \langle\psi(x,\Theta)|M|\psi(x,\Theta)\rangle$. In density matrix representation for possibly mixed states $\rho_{x,\Theta}^{\text{out}}$ as in Eq. (10) that equation becomes $f_\Theta(x) = \langle M\rangle_{\rho_{x,\Theta}^{\text{out}}} = \text{tr}\left(\rho_{x,\Theta}^{\text{out}}M\right)$ which allows to rewrite the cost function in Eq. (7) as

$$\epsilon_g = \frac{1}{|X|}\Sigma_{x\in X}\left|\langle M\rangle_{\rho_{x,\Theta}^{\text{out}}} - g(x)\right|^2. \tag{11}$$

We outline that we do not use the parameter matrix multiplication update rule defined by Ref. [29] in our work to train the dQNN ansätze but rather stick to the gradient method which we use for layered ansätze and is explained in more detail in the next subsection. In the follow-up work [70], the authors use a gradient descent method with a first-order approximation of the training cost's gradient. This way, they are able to approximate the gradient by evaluating the circuit on real quantum devices for slightly shifted parameters $(p-\epsilon)$ and $(p+\epsilon)$.

Since the authors of Ref. [21] suggest that a possible quantum advantage for machine learning tasks can only lie in the training on real quantum hardware, the search for optimal update rules is a major task ahead and left for further studies.

C. Numerical setup

As is standard for the application of VQAs, we use a hybrid quantum-classical optimization scheme [15]. Numerical results in this work were obtained using TensorFlow Quantum [83], which utilizes the quantum circuit simulator qsim [84], and Cirq [85]. Gradients are calculated with the Adjoint method [86, 87], which is suited for analytic simulations and the standard differentiator in TensorFlow Quantum. All results shown in the main text of this paper are calculated using analytic expectation values; for a comparison to shot-based values, see Appendix B. Moreover, we use a simulated noise model based on calibration data of the IBM Quantum Falcon r5.11H Processsor to demonstrate that the learning capability can be determined in a noisy setup and that general trends and observations can survive, see Appendix C.

For our trainings, we use the Adam optimizer [88] with a maximum of 360 epochs. The learning rate is decreased from 0.5 for the first 120 epochs to 0.1 for the second 120 epochs and 0.05 for the last 120 epochs. In Appendix D we compare the results for different hyperparameter configurations to justify this choice. The train and validation data sets contain 50, respectively, 100, data points for functions of degree less than 10, respectively, greater or equal 10. The x-values of the test data are calculated by including the endpoint of the interval $[0, 2\pi]$, whereas the endpoint is excluded in the validation data $[0, 2\pi)$ to ensure different sets. During the training, the train data is split into batch sizes of 25, resp. 50, and validated after each epoch using the validation data.

As a cutoff, we choose a loss value of 5×10^{-5} where the training terminates. We consider fits reaching this value as quasiperfect because they correspond to an average absolute error of 0.7% for each data point. While

this cutoff corresponds to overfitting, it reveals ansätze that are well suited to learn arbitrary partial Fourier series of a given degree. As the exact ground truth of the learning task for the learning capability is known, and we are interested in the model's ability to fit the Fourier functions exactly, such small absolute errors are sensible compared to a concrete real-world learning problem, where the generalization over potentially incomplete, corrupted, or noisy data has to be balanced and traded off with model accuracy on training data.

We test on 100 randomly chosen normalized Fourier functions. This gives stable results for mean values and standard deviations of the loss values, which are independent of the random initial parameters and the set of Fourier functions for each training. In Appendix E we provide details on the generation of the Fourier functions, their correlation, and the dependence of the learning capability on the set of Fourier functions.

We provide a marker for an average loss value of 6.25×10^{-4} depicted as a gray line in the plots for the learning capability because this level corresponds to an absolute average error for each function fit of less than 5% of the maximum Fourier function value. Below that value, we regard average loss values as not statistically significant and view the corresponding architectures as equal regarding their learning capability since there is no clear way to distinguish between different loss values in this regime, and no practical reason to do so.

Regarding the architectures, we always use R_X gates to encode the real-valued input data and a σ_z-measurement on the last qubit for layered PQCs as well as for dQNNs.

IV. RESULTS

To get an intuition on how the learning capability of a PQC relates to the Fourier coefficient calculation and numerical methods introduced by Schuld et al. [33], we advise the reader to look into Appendix F where introductory results for simple circuits for Fourier degree 2 are presented that motivate our method. Furthermore, we look into the minimal number of qubits and layers required to learn a specific Fourier function in that Appendix.

In the following subsections, we demonstrate an example where two PQC architectures have the same sampled Fourier coefficients but differ in their ability to fit different Fourier functions, i.e., they show differences in their learning capability in Sec. IV A and analyze various elements of PQC architectures in detail for layered ansätze in Sec. IV B and for different dQNN circuits in Sec. IV C.

A. Comparison to sampling Fourier coefficients

We extend the numerical results by Schuld et al. [33], and show that the learning capability provides insights that cannot be gained, for example, by sampling parameters of PQCs and calculating Fourier coefficients.

As an example, we consider Fourier series of degree 6 and ansätze with three qubits, two layers, single-qubit operation R_Y, and a simple, linear entanglement structure with one entanglement layer. The complexity of the circuits makes analytic calculations already unfeasible and, hence, motivates the introduction of learning capability. The first row of Fig. 5 corresponds to results for such an ansatz with CZ gates as entanglement gates, the second row uses CNOT gates, the third uses CRX, and the fourth uses CRX gates but two entanglement layers instead of one. All circuits are fully depicted in Figs. 33, 34, 35, and 36 in Appendix L. Because the learning capability is an average value, we select and plot in the first column of Fig. 5 the learned model that has a loss value closest to the average. The details of the distribution of the errors are depicted in Fig. 20 in Appendix G. For the first two ansätze, determining the learning capability is not strictly necessary; however, it incorporates the fact that not all Fourier coefficients can be represented because the loss values for both ansätze are relatively high. The last two rows of Fig. 5 show a case in which analyzing Fourier coefficients alone is not enough to describe how well circuits can represent Fourier functions since both ansätze give access to all possible coefficients. However, determining the learning capability reveals that the last ansatz, with two entanglement layers, fits Fourier functions of degree 6 on average much better than the other ansätze.

How well an ansatz can fit Fourier functions depends on the ansatz, as well as the hyperparameters used for the training. However, we consider ansätze well suited in a practical sense, if they are robust to hyperparameter changes or if it is easier to find hyperparameters that lead to successful trainings. We provide a grid comparison of different hyperparameters for the third ansatz (three qubits, two layers, single-qubit operation R_Y, a simple, linear entanglement structure, one entanglement layer and CRX entanglement gates) in Appendix D. None of the 21 different hyperparameters lead to a different learning capability. Hence, none of these parameter sets improves the learning capability of this ansatz. This fact strongly indicates that ansätze which enable the same Fourier coefficients but have different learning capabilities have some profound differences.

B. Layered ansätze

Because we know that $nL = d$ is necessary to enable $d + 1$ Fourier coefficients (including c_0), we start with analyzing the impact of different combinations of qubits and layers. For degree 12 functions, for example, we can use the following ansätze: (12 qubits, 1 layer), $(6, 2)$, $(4, 3)$, $(3, 4)$, $(2, 6)$, and $(1, 12)$. If we choose layered architectures with simple, linear entanglement structure and parametrized single-qubit gates $R_Y R_Z$, then we can vary

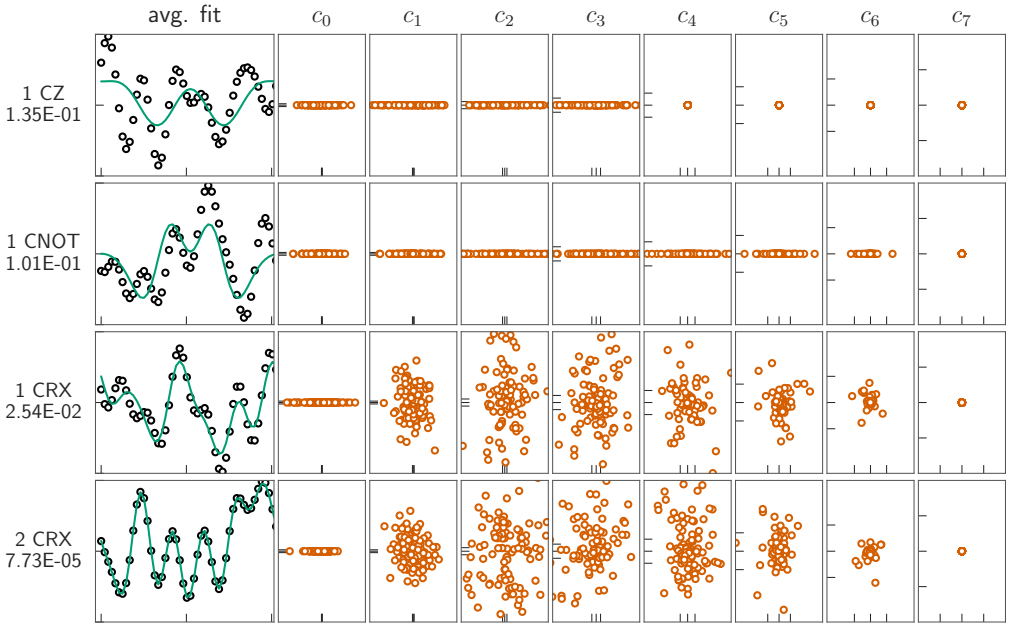


FIG. 5: **Left column**: Validation results closest to the average loss value for four different layered ansätze with $n = 3$, $L = 2$, and R_Y as the single-qubit rotation gate; the number of entanglement layers, the entanglement gate, and the average loss value are listed next to each graph. The results of the quantum circuits are shown as a green line and the Fourier functions as black circles. The x axis in this column is from 0 to 2π, the y axis from -1 to 1. **Right columns**: Fourier coefficients c_0 to c_7 resulting from the inverse Fourier transform of the results of evaluating the four ansätze with sampled parameters Θ. The ticks on the x and y axes are always at -0.01, 0, and 0.01, showing how the Fourier coefficients become smaller with increasing degree due to the function values being limited to the range $[-1, 1]$. The x axis shows the real part of the coefficient, while the y axis shows the imaginary part. **Rows**: first: ansatz with CZ as the entangling gate and one entanglement layer; second: ansatz with CNOT as the entangling gate and one entanglement layer; third: ansatz with CRX as the entangling gate and one entanglement layer; fourth: ansatz with CRX as the entangling gate and two entanglement layers per W.

the amount of entanglement layers as well as the entanglement gates. A summary of this comparison is depicted in Fig. 6. The learning capability is plotted with the 95% confidence interval for each architecture and the 5% absolute error baseline is drawn in gray. First, we see that ansätze with $L \lesssim n$ have the best learning capability μ_{12}, while ansätze with $L = d$ or $n = d$ show a poor learning capability μ_{12} with values of $0.1 \geq \mu_{12} \geq 0.01$.

In principle, barren plateaus could result in worse learning capabilities for ansätze with large n because, as can be seen in Fig. 3 in Ref. [46], varying the number of qubits between 1 and 12 reduces the variance of the gradients from 10^{-1} to 10^{-4}, indicating a relatively strong decline. However, as can be seen in Fig. 4 in Ref. [46], the variance of the gradients declines much less when varying the layers, especially for a small number of qubits (roughly $n < 10$). From that figure, we would not expect to have vanishing gradients in our cases $(1, 12)$ and $(2, 6)$. Hence, barren plateaus fail to explain the fact that ansätze with four qubits perform consistently better than ansätze with fewer qubits. Numerical simulations which are explained in more detail in Appendix I confirm the conclusion that we draw from the figures in Ref. [46] regarding the barren plateaus for the PQCs that we consider.

Second, we find that CAN gates with two entanglement layers perform slightly better than CRX with three entanglement layers and CNOT gates are not able to achieve reasonable results with two entanglement layers. However, due to the structure of the CAN gates, the corresponding architectures need more two-qubit gates despite having less entanglement layers (see Tab. IV in Appendix J).

The confidence interval for the configuration with three qubits, four layer and CAN gates with two entanglement layers and four qubits, three layer and CRX gates with three entanglement layers are visually large. In Appendix G, especially Figs. 22 and 23, we plot the error distribution in comparison to results of different configurations. For those two examples, it becomes evident that the logarithmically scaled y-axis and cutting the plot's y-axis at 10^{-5} are the major reasons for the seemingly skewed and large confidence intervals.

A more detailed comparison is given in Fig. 28 in Appendix J, which includes all results for linear and cyclic entanglement structure and two and three entanglement

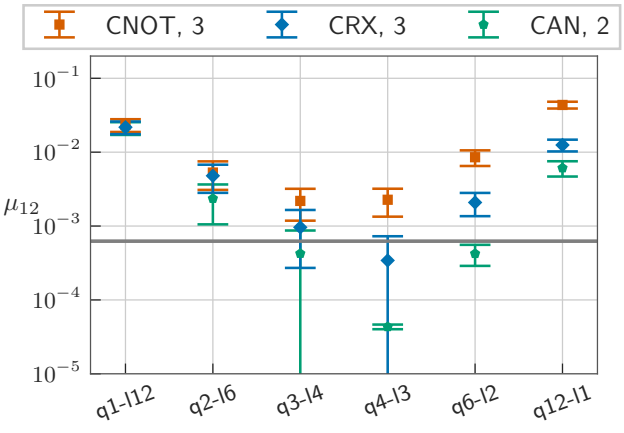


FIG. 6: Learning capability μ_{12} for layered ansätze with single-qubits gates $R_Y R_Z$ and simple, linear entanglement structure but different entanglement gates and layers. The different entanglement gates are $\{\mathrm{CNOT}, \mathrm{CRX}, \mathrm{CAN}\}$ with the number of entanglement layers $\{3, 3, 2\}$. The x axis shows the architectures in terms of number of qubits and number of layers. An extended version of this figure is shown in Fig. 28 in Appendix J.

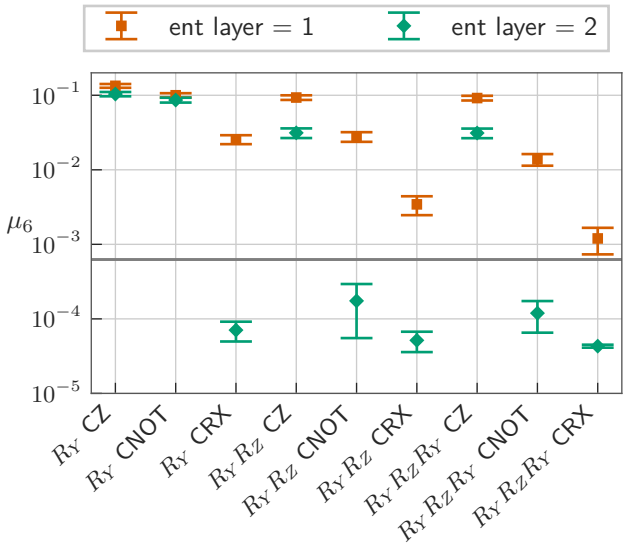


FIG. 7: Comparison of μ_6 for different layered ansätze with three qubits and two layers and simple, linear entanglement structure. The different $U^1 \in \{R_Y, R_Y R_Z, R_Y R_Z R_Y\}$ and entanglement gates $\{\mathrm{CZ}, \mathrm{CNOT}, \mathrm{CRX}\}$ are split for different entanglement layers $\{1, 2\}$. An extended version of this figure is shown in Fig. 29 in Appendix J.

layers. These results show that the cyclic entanglement structure does not lead to a significant improvement in learning capability. This might be due to the fact that the amount of qubits is quite small but is nevertheless remarkable because it shows that for ansätze representing degrees ≤ 12, cyclic entanglement structure is mostly not needed and thus, the gate count can be reduced.

Next, we systematically analyze the impact of single-qubit unitaries and methods to construct the trainable Ws. We choose degree 6 Fourier functions because they are complex enough to reveal meaningful results about parameterized single-qubit operations and complement the previous results. We outline the results of three qubit, two layer ansätze in Fig. 7 as this combination follows the rule of $L \lesssim n$. Within each plot, all ansätze utilize a simple, linear entanglement structure. By varying $U^1 \in \{R_Y, R_Y R_Z, R_Y R_Z R_Y\}$, we find that using $R_Y R_Z$ yields a satisfying learning capability in most cases, similar to $R_Y R_Z R_Y$ and, in most cases, strictly better than using a single R_Y. Thus, the use of arbitrary rotations with three parameters is, in many cases, not needed, but using only one R_Y is not enough to obtain good learning capabilities for layered ansätze. By varying the entanglement gates $\{\mathrm{CZ}, \mathrm{CNOT}, \mathrm{CRX}\}$ we show that, with our setup, CZ gates perform either as good as or worse than CNOT gates. Thus, we find that using CNOT gates as entanglement gates with no trainable parameter is preferable over CZ gates in a setup with R_X encoding and σ_z-measurement on the last qubit. When varying the entanglement layer $\{1, 2\}$ it turns out that, for degree 6 functions, using two entanglement layers is already enough for all considered entanglement gates. Remarkably, the ansatz with two linear entanglement layers, CRX, and R_Y has a high learning capability. However, this configuration had worse results for the learning capability on degree 12 functions, showing that this configuration does not perform as well on higher order problems. CZ gates represent the only cases where a cyclic entanglement structure performs clearly better than a linear structure. However, even in these cases, the values for the learning capability only catch up to the values of the corresponding ansatz with CNOT gates (see Fig. 29 in Appendix J). A comparison of the results from three qubits, twolayers with six qubits, one layer is also given in Fig. 29. It clearly indicates that three qubits, two layers performs much better than the other combination, similar to what we have seen for $d = 12$.

Since entanglement can play a larger role for larger qubit numbers, we increase the Fourier degree to 12 and test ansätze with four qubits and three layers. We choose ansätze with $U^1 = R_Y R_Z$ and entanglement gate CRX because they performed stable in previous experiments. We vary the entanglement structure $\{\mathrm{simple}, \mathrm{strong}, \mathrm{strongc14}\}$, which denotes the styles shown in Fig. 2 and the style used in circuit 14 from Ref. [35]. The first three entries in Fig. 8 show that these ansätze with two entanglement layers do not achieve a reasonable learning capability, no matter which structure we use. Instead, with three entanglement layers, all ansätze achieve an optimal learning capability indepen-

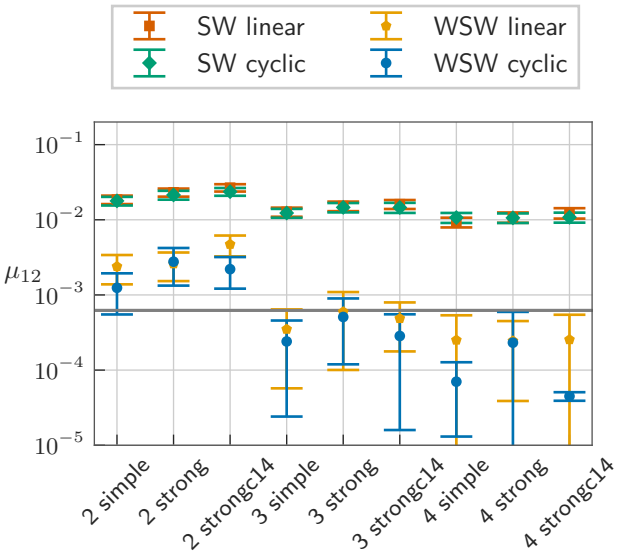


FIG. 8: Learning capability μ_{12} for layered ansätze with four qubits and three layers, $U^1 = R_Y R_Z$ and entanglement gate CRX, either with zero layer (WSW) or without zero layer (SW) and linear or cyclic entanglement. The y axis shows average loss values, while the x axis shows the architectures with different entanglement layer numbers $\{2, 3, 4\}$ and entanglement structures $\{\text{simple}, \text{strong}, \text{strongc14}\}$. The latter corresponds to the entangling structure of circuit 14 in Ref. [35].

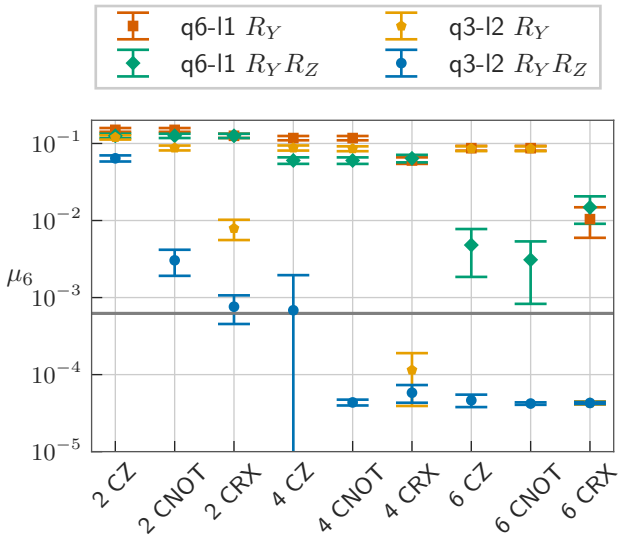


FIG. 9: Learning capability μ_6 of different alternating-layer (alt) ansätze with six qubits, one layer and three qubits, two layers. The x-axis shows different entanglement layer numbers and entanglement gates per W.

dent of their entanglement structure and style. Furthermore, as the examples in Appendix F 1 suggest—even beyond the case of one qubit and one layer—having a zero layer can enhance the learning capability drastically. All presented results demonstrate that increasing the number of entangling layers can change the learning capability quite drastically while not changing the Fourier coefficient picture. The results also indicate that the amount of entanglement layers is much more important than the structure that is implemented within each entanglement layer.

Using only half of the controlled operations in each entanglement layer, as done in the hardware-efficient or alternating layer ansatz reduces the learning capability; similar learning capabilities with these ansätze can be achieved by doubling the number of entanglement layers. This can be seen in Fig. 9 where entanglement layers and gates vary as well as single-qubit operations. Furthermore, these results reveal the following three aspects of a circuit's depth: (1) architectures with four alternating entanglement layers with CRX and R_Y or $R_Y R_Z$ have a very low learning capability value, indicating that if one is interested in highly expressible PQCs the R_Y has lower depth compared to $R_Y R_Z$; (2) increasing the depth to six alternating entanglement layers does not improve the learning capability value, indicating that, in this setup, four alternating entanglement layers could be used for an architecture with lower depth; and (3) if the circuit must be short, i.e., contain two alternating entanglement layers, then utilizing the CRX gate would lead to PQCs with a lower learning capability value than CZ or CNOT.

The larger confidence interval for the configuration with CZ gates and two entanglement layers is partially due to the logarithmic scaling, but the confidence interval is indeed larger than, for example, the configuration with CRX gate and one entanglement layer because the result contains one outlier. The error distribution is plotted and compared to other results in Fig. 21 in Appendix G.

To summarize our results for layered ansätze, we state that the best learning capabilities can already be achieved by using only a CNOT gate or the one-parameter CRX gate. The three-parameter CAN gate has a slightly better learning capability but also more parameters even though only two entanglement layers are necessary when using this gate for $d = 12$ (see Figs. 6 and 28). The entanglement style (linear or cyclic) as well as the entanglement structure (simple or strong) has almost no impact on μ_d for $d \leq 12$ (see Fig. 8). This shows that some necessary amount of entanglement has to be created by the ansatz. We establish that, for degree 6, two entanglement layers are enough for most architectures to gain sufficient learning capability (see Fig. 5) and three entanglement layers are enough for degree 12 functions (see Figs. 28 and 8). Increasing the entanglement layer count further does not improve the learning capability and, thus, can be considered inefficient for functions with equal or less degrees (see Fig. 8).

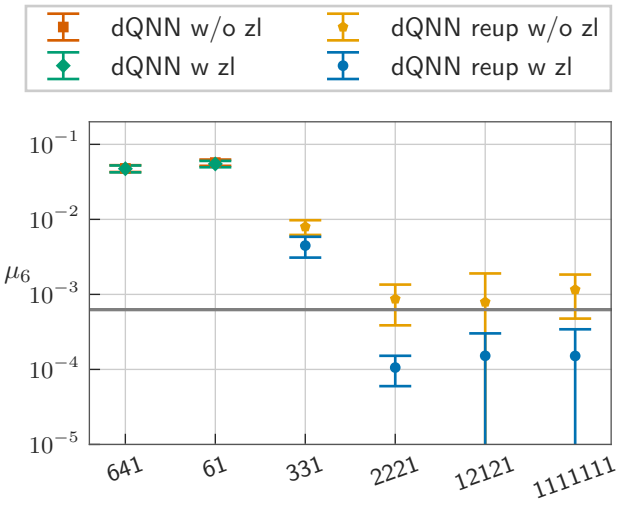


FIG. 10: Comparison of learning capability μ_6 of dQNN ansätze with $U^1 = R_Y R_Z$ and without [see Fig. 4b] or with [see Fig. 4c] zero layer (zl). The x axis shows the configuration of the dQNN ansätze, where the first digit is the number of qubits in the input layer, the last digit is the number of qubits in the output layer, and the digits in between describe the number of qubits in the hidden layers. The first two architectures have no data reupload on their hidden qubits.

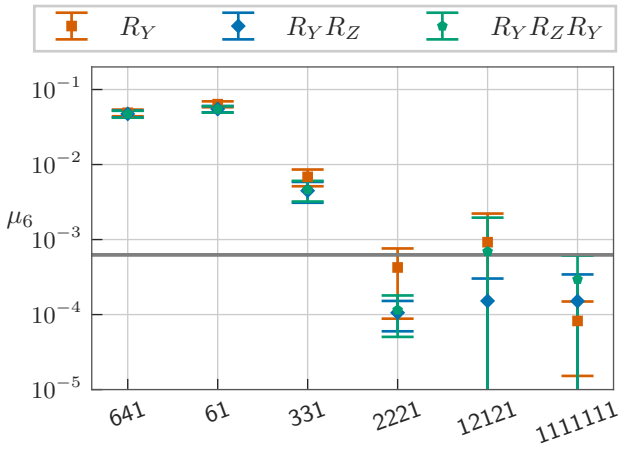


FIG. 11: Comparison of learning capability of dQNN ansätze with zero layers according to Fig. 4c with different single-qubit unitary gates. The y axis shows the learning capability as measured via the average mean squared error. The x axis shows the configuration of the dQNN ansätze, where the first digit is the number of qubits in the input layer, the last digit is the number of qubits in the output layer, and the digits in between describe the number of qubits in the hidden layers. Ansätze with different single-qubit unitaries are represented in different colors.

C. dQNN ansätze

The dQNN ansatz type is shown in Fig. 4 for the original circuit architecture from Refs. [29, 70] and our enhancements, including the data reupload structure. Similar to the layered ansätze, one can see from Fig. 19 in Appendix F 2 that a certain number of hidden qubits and layers is needed to achieve meaningful learning capability for a certain Fourier degree when using the dQNN ansatz type. In the case of $d = 6$, a structure of at least $[2, 2, 2, 1]$ was needed where the notation is according to Eq. (9). This ansatz has two hidden layers and a maximum of four neighboring qubits, meaning qubits that are directly connected to each other.

Evaluating μ_6 for the dQNN ansatz type with $U^1 = R_Y R_Z$, we find as a first result that the learning capability is close to $\mu_6 \approx 0.1$, and we are, thus, not able to learn Fourier series when we encode classical data only on the input qubits as depicted in Fig. 4a. This is shown in Fig. 10 where the first two ansätze $[6, 1]$ and $[6, 4, 1]$ have enough qubits but not the right structure to achieve a meaningful learning capability. Note that the ansatz $[6, 4, 1]$ has a hidden layer of four qubits without data reupload. Defining a data reupload strategy for the hidden qubits increases the learning capability drastically. This is shown in Fig. 10 for both reupload ansätze from Fig. 4b and Fig. 4c. Furthermore, we find that a medium-sized amount of neighboring qubits (three or four in the case of a degree six function) is preferable over having many neighboring qubits (seven in the case of degree six) or only a few (two for degree six). If the number of neighboring qubits is higher (four or six instead of three for μ_6), then the learning capability is better when more hidden than input qubits are used [see Fig. 10].

Next, we investigate the effect of different single-qubit gates on the learning capability of dQNN ansätze. First, we look at the effect of introducing a zero layer of parametrized rotations on each qubit, i.e., changing the architecture from the one in Fig. 4b to the one in 4c. Starting with single-qubit parameterized rotations in a $U^1 U_{\text{in}} U^1$ structure instead of starting with the data-encoding in a $U_{\text{in}} U^1$ structure and including the parametrized rotations on each hidden qubit as well increases the learning capability consistently (see Fig. 10). However, compared to the layered ansätze, the effect of this change is not as significant, and basic learning capability can already be achieved without a zero layer.

As a second step, we compare the learning capabilities for dQNN ansätze with different types of parametrized rotations in Fig. 11. We find that using single-qubit gates $R_Y R_Z$ yields a satisfying learning capability, which is similar to an ansatz using $R_Y R_Z R_Y$. However, a single R_Y gate already yields satisfying results in many cases. Thus, the use of rotations with three or even two parameters is not needed in our test cases for dQNN ansätze, which could be due to the rich entanglement structure of the dQNN ansätze resulting from the CAN gates.

V. DISCUSSION AND OUTLOOK

In this paper, we have defined the new measure of learning capability, which quantifies how well a PQC ansatz is able to learn Fourier functions of a specific degree on average. This quantity turns out to be necessary since existing measures for PQC performance, like the expressiveness in terms of the Haar measure [35], in terms of Fourier series coefficients [33], or in terms of dynamical Lie algebras (see Appendix. A), miss differences between certain ansatz types or lack a quantification of the trainability. The learning capability experiments reveal that PQCs exist, which provide all Fourier coefficients for the respective degree, but differ in their ability to fit different Fourier functions, i.e., they show differences in their learning capability.

The results in Sec. IV A show that even though the inverse Fourier transform of a circuit sampling provides all the Fourier coefficients for the respective degree, the learning capability may still be small, which extends the numerical results provided in Ref. [33]. One reason for different learning capabilities in these cases could be due to interdependencies between different Fourier coefficients that hinder access to arbitrary Fourier functions.

We have used the learning capability of PQCs to compare several layered and dQNN type ansätze. Thereby, we hope to build a bridge between theoretical analysis and concrete machine learning tasks that takes into account the input encoding, expressiveness, entanglement, and trainability. This is especially important considering the gate count reduction that is needed in the NISQ era and the different native gate sets that are available on different quantum computing hardware. When proposing new PQC architectures, the learning capability can be used to determine their performance compared to existing architectures.

In general, we find that both layered and dQNN ansätze have a similar performance for a similar parameter or gate count (see Table V). For layered ansätze, the results suggest that an efficient architecture with high learning capability μ_d for Fourier functions with degree $d \in \{6, 12\}$ can be obtained by using a similar number of qubits and layers or slightly more qubits than layers which can be stated by $n \gtrsim \sqrt{d}$ and $L \lesssim \sqrt{d}$. Furthermore, two rotation gates $R_Y R_Z$ and the CNOT or CRX gate with a simple, linear entangling structure yield good learning capability. Within this setup, using more entanglement layers within one trainable unitary W increases the learning capability until a saturation limit is reached which depends on the degree that is possible due to the data reupload structure. Our results show that two (respectively, three) entanglement layers in each W yield good learning capabilities for degree six (respectively, 12) functions.

For dQNN ansätze, a data reupload technique, inspired by the reupload technique for layered PQCs, turned out to be necessary. While these ansätze allow for many different amounts of input and hidden qubits, it becomes apparent that having more hidden than input qubits leads to better learning capabilities μ_6 and that the hidden qubits should be structured in several hidden layers. In contrast to layered ansätze, the use of R_Y as a single-qubit gate is sufficient to achieve good learning capability μ_6 for dQNN ansätze.

Overall, the learning capability of a chosen circuit ansatz is inherently limited by the structure and gate usage of the ansatz. Thus, a careful selection of the ansatz type and the number of qubits and layers is needed to build efficient circuits. When designing circuits for practical use cases, those numbers determine the maximum Fourier degree that can be learned and one should select the PQC structure and gates such that the learning capability is suitable. For example, the learning capability can be used to compare different ansätze with a similar value and identify the architectures that need fewer single-qubit operations, fewer entanglement layers, or a different qubit-to-layer ratio. This way, the depth can be reduced, leading to reduced barren plateau effects without lowering the capability to learn Fourier series.

Because discrete Fourier series can approximate arbitrary functions up to a certain precision, we conclude that the performance of PQCs in learning Fourier functions can give some guidelines on the performance in general learning tasks. For example, we expect the architectures that learn Fourier series with low degrees inaccurately, i.e., with high learning capability values, to be unsuited for supervised learning tasks with a complex dependency between input and output data. We also expect architectures that learn a variety of Fourier series with a given degree accurately to be more likely to capture complex dependencies between input and output data. However, we emphasize that this assumption about the correlation between performance on Fourier series and general QML tasks, of course, may not hold universally.

Under this assumption, our results can be used as guidance regarding this selection, meaning that a well-informed initial guess for the architecture of a PQC can be chosen by involving the learning capability. We concretize this by considering two different problems.

First, we assume that very little is known about the solution of a supervised learning problem. Similar to the arguments presented in Ref. [63], we advise starting the training of a QML model with an ansatz with a good learning capability, i.e., choosing layered PQCs with $L \lesssim n$, $R_Y R_Z$, and CRX gates, and to seek for minimizing the training error. By choosing the number of qubits and layers at the beginning of the procedure, the number of input gates is determined and hence, the maximal possible Fourier degree is known. The entanglement layers can be selected accordingly in the sense that two (respectively, three) are sufficient for six (respectively, 12) input encoding gates. More entanglement layers should be considered if more Fourier degrees are targeted. The number of qubits and layers (and entanglement layers) can successively be increased if the ansatz is not able to achieve low errors on the training data. Once an architec-

ture has a low error on the training data, the evaluation on the test data determines if the architecture is suitable for the given task. In case of a large error on the test data, overfitting might be occurring and standard machine learning techniques like regularization could be applied as well as choosing PQC ansätze that reduce the depth, i.e., contain fewer entanglement layers, or those that reduce the gate complexity for compiling them on quantum hardware, i.e., use CNOT gates instead of CRX, to reduce the overfitting. More practically speaking, finding a model that is suited for the training and test data is done simultaneously and by (automatized) hyperparameter search. Hyperparameter search grows exponentially in the number of considered parameters. Therefore, being able to reduce these parameters—e.g., by excluding the parametrized single-qubit operation by restricting it to $R_Y R_Z$ and reducing the considered combination of n and L by excluding the extreme cases—provides benefits to hyperparameter search.

Second, we present nonlearning problems in quantum computational fluid dynamics (qCFD); a growing field of interest (see, e.g., Refs. [89, 90]) that investigates the simulation of nonlinear classical systems on quantum computers. In CFD, Fourier series are known to describe initial conditions that need to be accurately represented by quantum states or circuits. Moreover, spectral methods in fluid dynamics allow to express flow solutions as coefficients for ansatz functions such as Fourier series [91]. Our analysis and the learning capability provide concrete results on the question which architectures are more likely to learn Fourier series accurately. Hence, studying PQCs for state preparation of CFD-relevant initial states is a possible area of application. In addition, in Ref. [92], the authors propose a method that includes quantum variational computing to solve nonlinear problems. The authors propose matrix product states to represent energy potentials in quantum circuits and use them for the case where the energy potential is described by a Fourier series. Therefore, investigating PQCs for representing these energy potentials is a similar possible area of application where our work provides guidelines on the choice of architecture.

In this work, we do not change the input encoding gate and use R_X for one-dimensional classical data in all experiments. However, evaluating the learning capability of PQC ansätze on multi-variate functions is an important next step to move towards common machine learning use cases. This especially means designing and analyzing those PQCs which provide enough degrees of freedom required for fitting multi-variate Fourier series [76]. Analyzing the effect of different input encodings per layer could be interesting for further studies. Additionally, one could also perform an in-depth investigation of the similarities between layered and dQNN ansätze.

The analysis of different gradient methods, like the quantum natural gradient [93] and the behavior of both training and evaluation on real quantum hardware are important topics that need further research. Finally, the learning capability could be used to analyze techniques like entanglement dropout [94] and determine circuit ansätze which are suited to reduce overfitting on real-world, noisy input data. For finding a possible quantum advantage in learning [21] or generalization [18], both research directions could be helpful.

VI. CODE

Code accompanying this paper is given in Ref. [95]. In this resource, we provide code to define an ansatz and determine its learning capability. The code can be used to perform the calculations presented in this paper and reproduce each reported value individually. We also provide code to create new sets of Fourier series and calculate their cross-correlations.

VII. ACKNOWLEDGMENTS

The authors thank Lukas Groß, Felix Wiebe, and Patrick Draheim for helpful discussions and the anonymous reviewers for their constructive remarks and recommendations to improve this manuscript.

We acknowledge support from the Bundesministerium für Bildung und Forschung (BMBF) through the project Q^3-UP! under grant numbers 40 301 121 and 13 N 15 779 administered by the VDI/VDE Innovation + Technik GmbH (VDI) and by the Bundesministerium für Wirtschaft und Klimaschutz (BMWK) through the project QuDA-KI under the grant numbers 50RA2206A and 50RA2206B administered by the Deutsches Zentrum für Luft- und Raumfahrt e.V (DLR). We acknowledge the use of publicly available IBM Quantum services for this work. The views expressed are those of the authors.

Appendix A: Dynamical Lie Algebras of PQCs

In what follows, we will show that the parameterized quantum circuits used in our study have isomorphic dynamical Lie algebras (DLA). We will focus on parameterized quantum circuits with data reupload. A single data reupload layer, denoted by $U_l(\theta_\mathbf{l}, \mathbf{x_l})$, is then composed of an encoding layer, denoted by $U_l^e(\mathbf{x_l})$ for some input $\mathbf{x_l}$, a parameterized layer, denoted $U_l^p(\theta_\mathbf{l})$ for some parameters $\theta_\mathbf{l}$, and an entangling layer, denoted U_l^{ent}. In this case, we have

$$U_l(\theta_\mathbf{l}, \mathbf{x_l}) = U_l^{\text{ent}} U_l^p(\theta_\mathbf{l}) U_l^e(\mathbf{x_l}) \tag{A1}$$

$$U_l^e(\mathbf{x_l}) = \prod_{k=1}^{K} e^{iH_k^e x_{lk}} \tag{A2}$$

$$U_l^p(\theta_\mathbf{l}) = \prod_{m=1}^{M} e^{iH_m^p \theta_{lm}} \tag{A3}$$

$$U_l^{ent} = \prod_{r=1}^{R} e^{iH_r^{ent}} \tag{A4}$$

$$U(\theta, \mathbf{x}) = \prod_{l=1}^{L} U_l(\theta_\mathbf{l}, \mathbf{x_l}) U_0^{ent} U_0^p(\theta_\mathbf{0}) \tag{A5}$$

where $U(\theta, \mathbf{x})$ represents the unitary matrix of the whole circuit, and L is the total number of layers.

Now that we have our notation set for the circuit itself, we will remind you of the necessary definitions for computing the dynamical Lie algebra.

Definition A.1. Given a parameterized quantum circuit $U(\theta, \mathbf{x})$, let $\mathcal{G}_e = \{H_k^e\}_{k=1}^K$ be the set of generators of an encoding layer, $\mathcal{G}_p = \{H_m^p\}_{m=1}^M$ be the set of generators of a parameterized layer, and $\mathcal{G}_{ent} = \{H_n^{ent}\}_{r=1}^R$ be the set of generators for an entangling layer. Then, we define $\mathcal{G} = \mathcal{G}_l = \mathcal{G}_e \bigcup \mathcal{G}_p \bigcup \mathcal{G}_{ent}$ to be the set of traceless Hermitian matrices that generate the unitaries of a single layer.

Definition A.2. Given the set of generators $\mathcal{G}$ of some parameterized quantum circuit, the dynamical Lie algebra $\mathfrak{g}$ is the Lie algebra spanned by the repeated nested commutators of the elements in $\mathcal{G}$:

$$\mathfrak{g} = \text{span}\langle iH_0, ...iH_t\rangle_{\text{Lie}} \tag{A6}$$

Lemma A.1. *Let $\mathcal{G}_p = \{O_i\}_{i=1}^n$, $\mathcal{G}_e = \{P_i\}_{i=1}^n$, where n is the number of qubits, $P_i \neq O_i \in \{X_i, Y_i, Z_i\}$ for all $i \in \{1, 2, ..., n\}$, and $\{X_i, Y_i, Z_i\}$ are the Pauli matrices on qubit i. The Lie algebra generated by the $\mathcal{G}_p \bigcup \mathcal{G}_e$ is*

$$\mathfrak{g}' = \underbrace{\mathfrak{su}(2) \oplus \mathfrak{su}(2) \oplus ... \oplus \mathfrak{su}(2)}_{n \text{ times}}. \tag{A7}$$

Proof. The proof follows immediately from the defining relations of $\mathfrak{su}(2)$ and the definition of a dynamical Lie algebra. □

In the case of a linear entangling layer, the generators of the entangling gates take the following form:

$$\mathcal{G}_{ent} = \{U_{i+1} - Z_i U_{i+1}\}_{i=1}^{n-1} \tag{A8}$$

This follows from the fact that we can write any controlled gate CU as

$$CU = e^{i(I - Z_1)U_2}, \tag{A9}$$

for some Hermitian matrix U_2.

Lemma A.2. *Let $\mathcal{G}_p$, $\mathcal{G}_e$ be as in the previous Lemma A.1, and let $\mathcal{G}_{ent} = \{U_{i+1} - Z_i U_{i+1}\}_{i=1}^{n-1}$, then the Lie algebra generated by $\mathcal{G}_p \bigcup \mathcal{G}_e \bigcup G_{ent}$ contains all nearest neighbor two-body interactions.*

Proof. From the previous Lemma A.1, we know that we have all single-qubit Pauli matrices. Then, we can generate all two-body interactions by computing $[U_{i+1} - Z_i U_{i+1}, X_i]$, followed by all possible commutators of the result with all single-qubit Pauli matrices. □

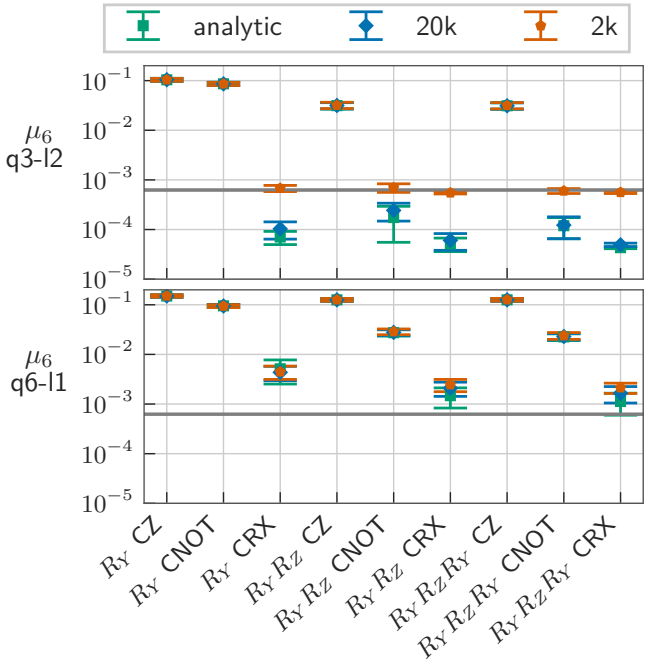


FIG. 12: Comparison of μ_6 for different layered ansätze with two simple, linear entanglement layers. The top plot shows results for three qubits and two layers, while the bottom plot shows results for 6 qubits and one layer. The green data points show analytic simulation results as in Fig. 29. The blue data points depict results of shot-based expectation values using 20000 shots and the parameter shift rule, while the orange data points show this for 2000 shots.

Lemma A.3. *The following holds:*

$$[M_i N_{i+1}, O_{i+1} P_{i+2}] = \alpha M_i Q_{i+1} P_{i+2} \tag{A10}$$
$$[[M_i N_{i+1}, O_{i+1} P_{i+2}], Q_{i+1} R_{i+2}] = \beta M_i S_{i+2} \tag{A11}$$

for all $M, N, O, P, Q, R, S \in \{X, Y, Z\}$, $\alpha, \beta \in \mathbb{C}$, *and* $N \neq O$.

Proof. The first equality follows immediately from the defining relations of $\mathfrak{su}(2)$. The second equality follows from the first equality as well the defining relations of $\mathfrak{su}(2)$. □

Proposition A.4. *Let* $\mathcal{G}_p$, $\mathcal{G}_e$ *be as in the previous Lemma A.1, and let* $\mathcal{G}_{ent} = \{U_{i+1} - Z_i U_{i+1}\}_{i=1}^{n-1}$, *then the Lie algebra generated by* $\mathcal{G}_p \bigcup \mathcal{G}_e \bigcup G_{ent}$ *contains all possible Pauli strings of length at most equal to the number of qubits* n *and the dynamical Lie algebra is* $\mathfrak{su}(2^n)$.

Proof. From the previous Lemmas, we can see that by repeating this argument for next-to-nearest neighbor interactions then next-to-next-to nearest neighbor interactions until we have all 2-body interactions as well as nearest neighbor 3-body interactions. Clearly, a similar argument can be made to generate all 3-body interactions and then n-body interactions. □

Corollary A.5. *The dynamical Lie algebra of any circuit presented in this paper is always given by* $\mathfrak{su}(2^n)$, *where* n *is the number of qubits.*

Proof. The proof follows by applying Proposition A.4 to any circuit. □

Corollary A.5 shows that all our considered circuits tend to exhibit barren plateaus for a large number of layers.

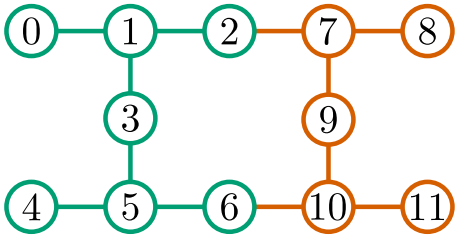


FIG. 13: Topology of the simulated device. The qubits with index $0 \dots 6$ are based on the actual qubits of the *ibmq_perth* system, whereas the added qubits with index $7 \dots 11$ and their couplings are copies to increase the system size.

Appendix B: Comparison with shot-based expectation values

We consider the case of shot-based determination of expectation values and the parameter shift value to determine gradients. The analytic expectation values are computed using the noiseless backend of TensorFlow Quantum with no repetitions, while the shot-based expectation values are computed with a given number of samplings from the state. Having the training and execution shot-based brings our setup closer to real world execution while remaining hardware independent.

Figure 12 shows, for three qubits and two layers, that the analytic values can be reached with 20000 shots per expectation value. If we decrease the number of shots to a value lower than $20000 = \frac{1}{5 \cdot 10^{-5}}$, then the maximum reachable precision is lower then the one chosen for our analytic experiments. This is evident in the results for 2000 shots where we get very similar results for architectures with high loss values, but are limited to a lower bound for the precision at $\sim 6.25 \times 10^{-4}$ (see values around the gray line in the upper row of Fig. 12). Comparing to possible experiments, we see that the shot-based expectation value converges to the analytical one, and that using a small number of shots would limit the maximum precision and would make circuit architectures with near-optimal learning capabilities indistinguishable. For six qubits and one layer, Fig. 12 shows that the values are very similar which is possible because the analytic values lay above the precision bound for 2000 shots $\sim 6.25 \times 10^{-4}$.

Appendix C: Evaluation under noise

To explore the usefulness of the proposed learning capability metric in an actual NISQ setting, we conduct exploratory experiments under simulated noise conditions that approximate the behavior of a current superconducting quantum processor. We opted for simulated noise over training on actual quantum hardware for two reasons. First, access to quantum hardware is limited and since the training and evaluation steps outlined below require millions of circuit evaluations, such a setting would be impractical as well as time and cost prohibitive. Second, execution on quantum hardware would necessitate transpilation to the native gate set of the chosen quantum processor. Depending how well certain gates can be expressed in that native gate set, results may be biased by an arbitrary choice of a quantum processor. Using a simulated device enables us to evaluate the exact same circuits and gates as in the noise-free setting.

1. Noise model implementation

We implemented a noise model in Cirq [85] based on a calibration data snapshot of the *ibmq_perth* system, which is one of the IBM Quantum Falcon r5.11H Processors. This system has seven qubits, of which we copied five to extend the simulated device to 12 qubits. Figure 13 shows the topology of the simulated device. The qubits with indices $0 \dots 6$ correspond to the original hardware, whereas qubits $7 \dots 11$ are mirrored from the five rightmost qubits of the original device, including their coupling properties.The relevant calibration metrics obtained from the quantum processor, that we use in our noise model, are summarized in Table I.

To simulate the quantum device noise, we assume decoherence is local and Markovian, which we model by thermal relaxation and depolarizing error channels [96] applied after each gate of the circuit. We compose the thermal relaxation noise $\mathcal{E}_{TR}$ from an amplitude damping $\mathcal{E}_{AD}$ and a phase damping channel $\mathcal{E}_{PD}$ [97, 98], such that for a quantum state ρ:

$$\mathcal{E}_{TR}(\rho) := \mathcal{E}_{PD}(\mathcal{E}_{AD}(\rho)). \tag{C1}$$

For simplicity and due to technical limitations of the software stack used for training the quantum models [83], we omit the small contribution of the excited state population [99] to the thermal relaxation error and assume a qubit temperature of $\Theta = 0K$. Hence, the amplitude damping channel is given by [98]:

$$\mathcal{E}_{AD}(\rho) := K_0 \rho K_0^\dagger + K_1 \rho K_1^\dagger, \tag{C2}$$

$$K_0 := \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma_{AD}} \end{bmatrix}, \tag{C3}$$

$$K_1 := \begin{bmatrix} 0 & \sqrt{\gamma_{AD}} \\ 0 & 0 \end{bmatrix}, \tag{C4}$$

with $\gamma_{AD} = 1 - e^{-\frac{t}{T_1}}$. The phase damping channel is defined accordingly as [98]:

$$\mathcal{E}_{PD}(\rho) := K_0 \rho K_0^\dagger + K_1 \rho K_1^\dagger, \tag{C5}$$

$$K_0 := \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma_{PD}} \end{bmatrix}, \tag{C6}$$

$$K_1 := \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{\gamma_{PD}} \end{bmatrix}, \tag{C7}$$

with $\gamma_{PD} = e^{-\frac{t}{T_1}} - e^{-\frac{2t}{T_2}}$. Here t is the gate time, and T_1 and T_2 the traverse and longitudinal relaxation times.

For the resulting error channel $\mathcal{E}_{TR}$, we compute the average fidelity [98, 100, 101]

$$\bar{\mathcal{F}}(\mathcal{E}_{TR}) = \int \langle\psi| \mathcal{E}_{TR}(|\psi\rangle \langle\psi|) |\psi\rangle \, d\psi = \frac{1}{2} + \frac{1}{6} e^{-\frac{t}{T_1}} + \frac{1}{3} e^{-\frac{t}{T_2}}. \tag{C8}$$

If the infidelity $1 - \hat{\mathcal{F}}(\mathcal{E}_{TR})$ is smaller than the total gate error reported in the calibration data, we model the remaining infidelity with an additional depolarizing channel $\mathcal{E}_{DP}$ [98]

$$\mathcal{E}_{DP}(\rho) := (1 - p_{DP})\rho + \frac{p_{PD}}{3} \sum_{\hat{\sigma} \in \{\hat{X}, \hat{Y}, \hat{Z}\}} \hat{\sigma} \rho \hat{\sigma}. \tag{C9}$$

The depolarization probability p_{DP} is derived from the gate error such that the average fidelity of the entire error channel $\mathcal{E}$ is

$$\begin{aligned} \bar{\mathcal{F}}(\mathcal{E}) &= \bar{\mathcal{F}}(\mathcal{E}_{DP} \cdot \mathcal{E}_{TR}) \\ &= (1 - p_{DP}) \bar{\mathcal{F}}(\mathcal{E}_{TR}) + p_{DP} \bar{\mathcal{F}}(\mathcal{E}_D), \end{aligned} \tag{C10}$$

where $\mathcal{E}_D = \frac{\hat{I}}{2^n}$ is the completely depolarizing channel [98]. Since TensorFlow Quantum [83], the quantum machine learning toolkit used in this work, does not simulate measurements on the circuit but determines expectation values

TABLE I: Relevant metrics from the calibration snapshot of the *imbq_perth* system for our noise model. The table shows traverse (T1) and longitudinal (T2) relaxation times as well as singe-qubit gate times (t), single-qubit gate errors and read-out assignment errors for individual qubits of the system as well as two-qubit gate times and errors for each coupling. Calibration data was obtained on June 1st, 2023.

						Couplings		
Qubit	**T1 (µs)**	**T2 (µs)**	**t (ns)**	**Gate Err.**	**RO Err.**	**Cp.**	**t (ns)**	**Gate Err.**
0	9.6	16.47	35.56	0.0007	0.0220	0→1:	391.11	0.0129
1	150.65	55.92	35.56	0.0003	0.0232	1→0:	426.67	0.0129
						1→2:	355.56	0.0052
						1→3:	405.33	0.0145
2	120.61	103.28	35.56	0.0002	0.0205	2→1:	320.00	0.0052
3	169.23	151.85	35.56	0.0003	0.0176	3→1:	369.78	0.0145
						3→5:	284.44	0.0086
4	159.29	117.10	35.56	0.0005	0.0178	4→5:	590.22	0.0103
5	187.23	140.95	35.56	0.0003	0.0240	5→3:	320.00	0.0086
						5→4:	625.78	0.0103
						5→6:	640.00	0.0102
6	163.37	180.04	35.56	0.0002	0.0060	6→5:	604.44	0.0102

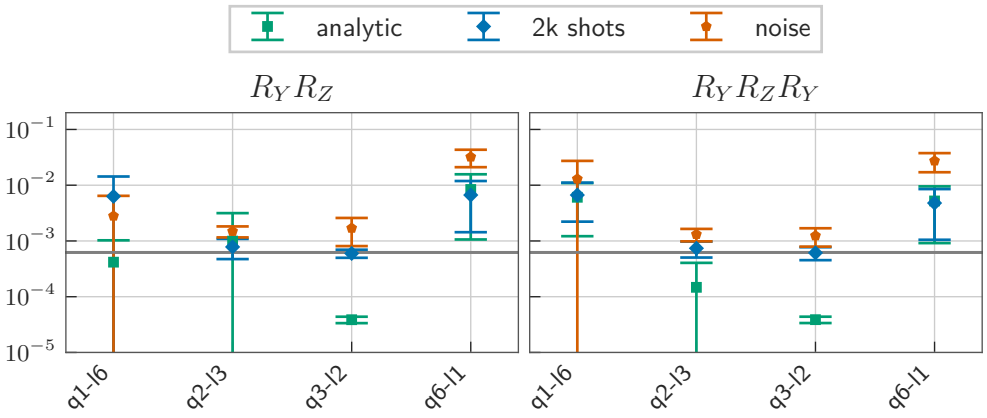


FIG. 14: Comparison of μ_6 for different layered ansätze with two simple, linear CNOT entanglement layers. The left plot shows results for $R_Y R_Z$, while the right plot shows results for $R_Y R_Z R_Y$ single-qubit operations. The green data points show analytic simulation results as in Fig. 29. The blue data points depict results of shot-based expectation values using 2000 shots and the parameter shift rule. The orange data points are based on shot-based expectation values for 2000 shots and the noise model discussed in the previous Appendix C 1.

for an observable directly from the computed quantum state, we approximate the readout error of the simulated device by appending a bit-flip channel $\mathcal{E}_{BF}$ [98] to the end of the circuit. The channel is defined by

$$\mathcal{E}_{BF}(\rho) := (1 - p_{BF})\rho + p_{BF}\hat{X}\rho\hat{X}. \tag{C11}$$

The probability for the bit-flip p_{BF} is set to the readout assignment error probability from the calibration data for each qubit.

2. Evaluation and results

We provide an outline on how to use the learning capability in the presence of noise. Although a more detailed investigation is out of scope of this manuscript, this section shows that the learning capability can, in principle, be evaluated with a quantum circuit simulator that incorporates noise as described in the previous Appendix C 1. At first, normalizing the Fourier function values to values less than one allows the PQCs to compensate for the noise effects. We do so by dividing each function value by $\frac{3}{4}$. Furthermore, to reduce the computational effort, we restrict the set of Fourier series to 10 functions and select them by choosing the ones that are included in the function pairs with the lowest cross-correlation values as calculated in Appendix E.

We test the experiments for degree 6 functions and ansätze with two simple, linear CNOT entanglement layers. Figure 14 shows the results for these experiments for analytical simulation, shot-based, and noisy shot-based expectation values with 2000 shots. We select qubit $\{0\}$ for 1 qubit, $\{0, 1\}$ for 2 qubits, $\{0, 1, 2\}$ for 3 qubits and $\{0, 1, 2, 7, 9, 10\}$ for 6 qubits from the approximated processor in Fig. 13.

The analytical case shows that the changes to reduce the computational effort still enable the reproduction of the characteristic curve except for the one qubit, six layers case with $R_Y R_Z$ single-qubit gates which performs better than expected. The shot-based experiments without noise, however, follow the exact characteristic curve with a higher minimal loss for all runs as expected from Appendix B. That is, the two and three qubits architectures perform the best and the one and six qubits architectures lead to worse results. Including noise weakens the learning capability further. However, the characteristic curve still remains and shows differences in the choice of qubits and layers. Again, the one qubit case performs slightly better than expected which could be due to statistical variances or because of the absence of two-qubit errors. Further noise experiments would be necessary to investigate this finding in more detail.

Appendix D: Hyperparameters

Our definition of learning capability depends on hyperparameters of supervised learning, namely the optimizer, the initialization, epochs, and learning rates. On the one hand, this results from the fact that we provide a measure close

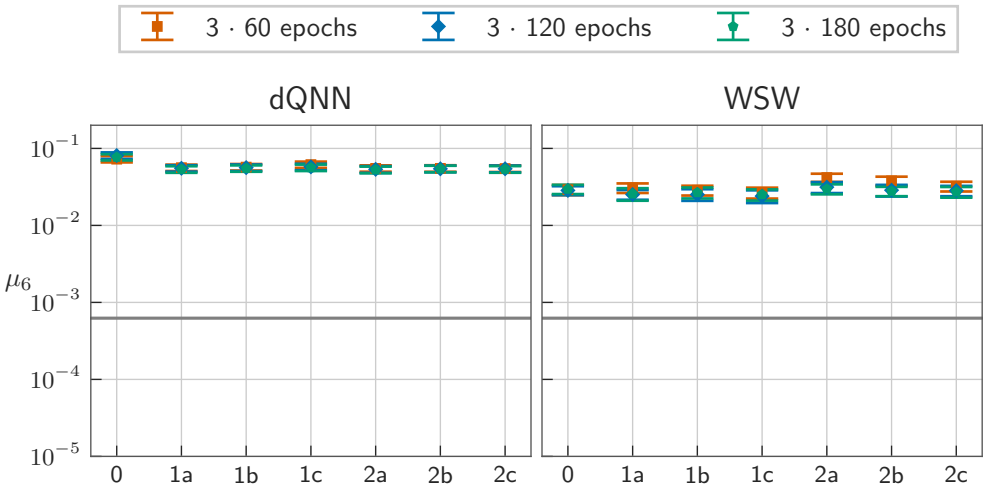


FIG. 15: Overview of values for μ_6 for different hyperparameter sets. The results are depicted for a layered ansatz with $q = 3$, $l = 2$, el $= 1$, $U^1 = R_y$ and CRX and a dQNN ansatz with [61] and $U^1 = R_Y R_Z R_Y$. The values on the x axis are explained in Table II. The legend shows the different epoch sizes per learning rate. For example, the blue marker for setting 2a corresponds to learning 120 epochs with learning rate 0.1, 120 epochs with learning rate 0.05, and finally 120 epochs with learning rate 0.01.

to practical work. On the other hand, this could lead to different results depending on the hyperparameters and not solely on the chosen ansatz. Therefore, we provide a comparison of results obtained by different hyperparameters in Fig. 15. The data in the figure shows the following:

1. Losses are very stable in the range of the tested hyperparameters for the tested ansätze.
2. The gap between one and two entanglement layers in Fig. 5 is not resolved by any of these hyperparameter sets.
3. The gap between dQNNs with and without reupload on hidden neurons in Fig. 10 cannot be resolved.

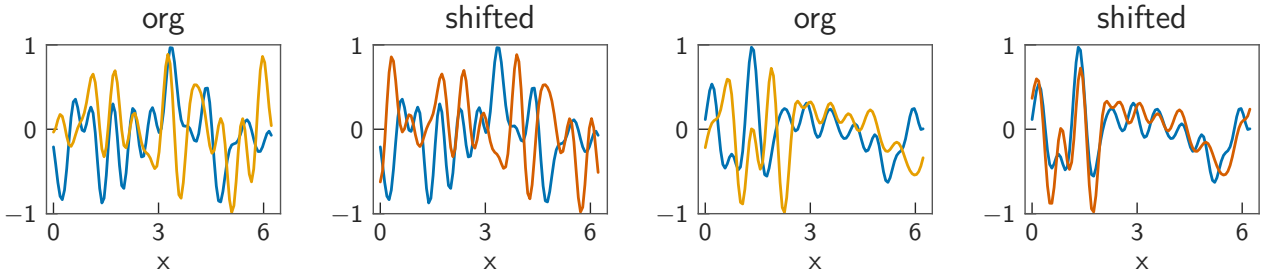
This grid test of hyperparameters gives strong evidence that differences in learning capabilities can be explained by differences in the architecture. However, since our grid test could still miss an important hyperparameter set, we conclude that these sets would be hard to find and thus the ansatz were more impractical then others, because extensive hyperparameter optimization would be needed to obtain similar results.

Appendix E: Details on the sets of Fourier functions

To generate truncated random Fourier series of degree d, we sample for each frequency ω a complex-valued coefficient $c_\omega = a_\omega + ib_\omega$ as two random numbers between -0.5 and 0.5 using a uniform distribution. Due to the constraints

TABLE II: Learning rates for results in Fig. 15.

	Learning rates		
Set	l_0	l_1	l_2
0	0.3	0.3	0.3
1a	0.5	0.1	0.05
1b	0.5	0.1	0.1
1c	0.5	0.5	0.1
2a	0.1	0.05	0.01
2b	0.1	0.05	0.05
2c	0.1	0.1	0.05



(a) Best cross-correlation function pairs.

(b) Worst cross-correlation function pairs.

FIG. 16: Cross-correlation for the original dataset of degree 12 functions. The function pair with the lowest and the highest correlation is depicted in blue and red. Because we consider shifted curves to determine the maximal cross-correlation value for a function pair, the original curve is depicted in yellow.

of PQCs, the coefficients corresponding to $-\omega$ are given by the complex conjugate $c_{-\omega} = \overline{c_\omega}$. Subsequently, all coefficients are normalized such that the highest absolute value of the truncated Fourier series is equal to 1. The code to generate the set of random truncated Fourier series is given in Notebook.

As it was explained in Sec. III A, the mean error μ_d is given by

$$\mu_d = \frac{1}{|G_d|} \sum_{g \in G_d} \epsilon_g, \tag{E1}$$

where ϵ_g is the final validation loss for the Fourier series g. Each parameterized quantum circuit ansatz is reinitialized and trained on a sample for several truncated Fourier series which form the set G_d. For very similar functions in the set one might get the illusion that a model performed well on n Fourier series, when in fact several of them are the same. Therefore, a key point is to ensure that the sampled Fourier series are different to guarantee that we minimize the effect of the sample on the results. This is done by computing the cross-correlation matrix of all the Fourier series in the set G_d.

The cross-correlation is a similarity measure between two series which can be used to objectively quantify how similar two (or more) series are. For two continuous periodic functions f and g with period T, the cross correlation is

$$(f * g)(\tau) = \int_{t_0}^{t_0+T} \overline{f(t)} g(t+\tau) dt \tag{E2}$$

where $\overline{f(t)}$ is the complex conjugate of $f(t)$ and τ is the lag. For the τ values, we divide the interval $[0, 2\pi]$ into 100 equal parts. We calculate the cross-correlation for $\tau = \frac{2\pi \cdot k}{100}$ with $k \in \{0, \ldots, 99\}$. Clearly, for each value of the lag we get a value of the cross-correlation of f and g.

In our case, we are interested in the largest value this cross-correlation could take between the two Fourier series, meaning the lag for which the two Fourier series are most similar. This way, we know that the two Fourier series are not the same up to a shift in the x axis.

The analysis of the cross correlation of our Fourier series data set of degree 12 is given in Table III. For two data sets, G_{12} and G'_{12}, the absolute values of the upper triangle matrix without diagonal elements are listed by counting the number values lying in the specified interval. The values in the table show that the Fourier functions are in general uncorrelated. The function pairs with the lowest and highest cross-correlation are depicted in Fig. 16 which also visualizes the differences between the original and shifted functions.

TABLE III: Upper triangle of the cross-correlation matrix.

set	[0.0, 0.1)	[0.1, 0.2)	[0.2, 0.3)	[0.3, 0.4)	[0.4, 0.5)	[0.5, 0.6)	[0.6, 0.7)	[0.7, 0.8)	[0.8, 0.9)	[0.9, 1.0]
G_{12}	0	0	0	254	2350	1925	383	38	0	0
G'_{12}	0	0	0	268	2404	1823	412	43	0	0

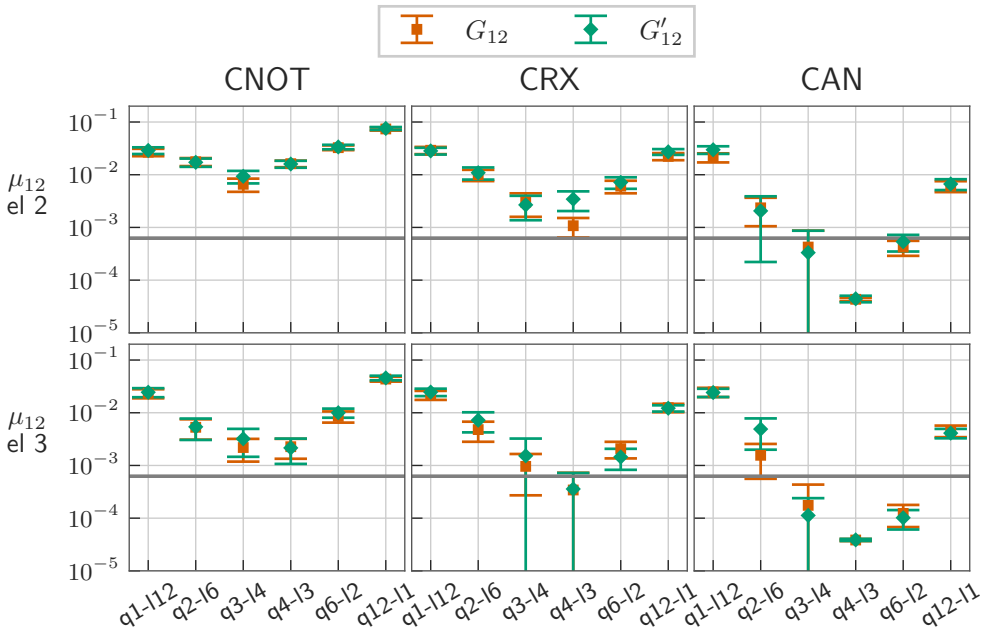


FIG. 17: Comparison of learning capabilities for two different sets of randomly sampled truncated Fourier series. Orange data points show results for the original data as in Fig. 28. Green data points show the result of running the experiments on a different set of truncated Fourier series. The learning capability μ_{12} is for different WSW ansätze with $R_Y R_Z$ single-qubit gates, simple linear entangling structure and $\{\text{CNOT}, \text{CRX}, \text{CAN}\}$ entangling gates. The top row shows results for two entanglement layers, while the bottom row shows results for three entanglement layers.

To test the impact of different sets G_{12}, G'_{12} We compare the results for the learning capability for two different sets of randomly sampled truncated Fourier series of degree 12 in Fig. 17. This shows that there is a difference in the results for the two sets, but that this difference is rather marginal and not statistically significant.

Appendix F: Preliminaries

We present introductory results for simple circuits which underline that data reupload can in principle provide a frequency spectrum of size $K = nL$ but that the structure of the trainable unitaries W determines whether all Fourier coefficients can indeed be represented. The circuits for this section are shown in Appendix L. This Appendix reproduces the results from Ref. [33] for simple examples and demonstrates that the learning capability naturally captures the numerical results.

1. One qubit and one layer

Some frequently used circuits start with the input encoding directly on the input qubits without an initial block of parametrised unitary gates (see for example Refs. [10, 12, 102]). For a systematic comparison we refer to this idea as SW instead of WSW. If we consider the elementary case of one qubit and one layer, then a calculation shows that SW has not enough expressiveness to fit all Fourier functions with degree 1 because the coefficient c_0 of Eq. (4) is always 0. The calculation is given in Example 2 in Appendix K. It uses a σ_z measurement but is independent of the exact implementation of the unitary U^1. This fact can be reproduced numerically in two different ways by choosing three different ansätze: The first ansatz is based on a SW structure with $U^1 = R_Y R_Z R_Y$, i.e. $U_{SW,3}(x, \Theta) = R_Y(\theta_{13}) R_Z(\theta_{12}) R_Y(\theta_{11}) R_X(x)$; the second one has a zero layer but only one gate as the single-qubit operation: $U_{WSW,1}(x, \Theta) = R_Y(\theta_{11}) R_X(x) R_Y(\theta_{01})$; and the third ansatz consists of a zero layer and two gates per single-qubit operation: $U_{WSW,2}(x, \Theta) = R_Z(\theta_{12}) R_Y(\theta_{11}) R_X(x) R_Z(\theta_{02}) R_Y(\theta_{01})$. Each ansatz corresponds to a row in Fig. 18 and the circuits are depicted in Appendix L.

Two tools, introduced in Ref. [33], can be used to analyze the expressive power numerically: The first tool is to

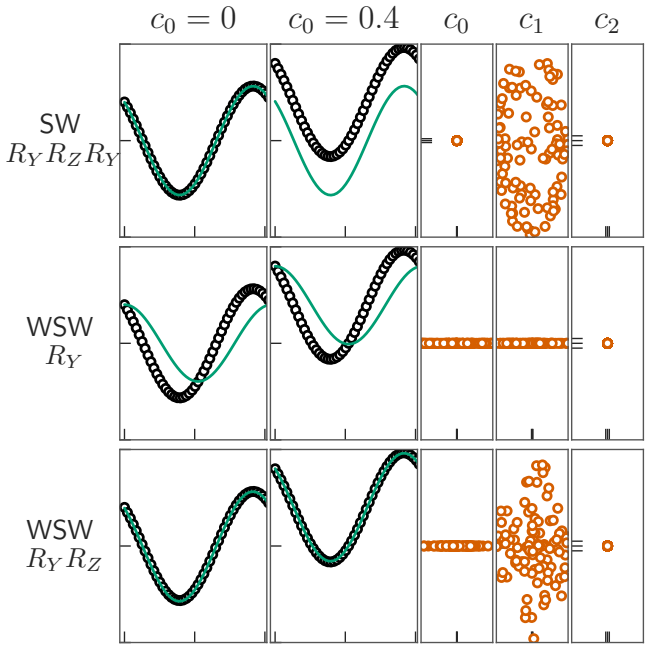


FIG. 18: **Left columns**: Validation results (green lines) of two different trained functions (black circles) with different $c_0 \in \{0, 0.4\}$. The x axis is from 0 to 2π, the y axis from -1 to 1. **Right columns**: Fourier coefficients c_0 to c_2 resulting from the inverse Fourier transform of the results of evaluating the three ansätze with sampled parameters Θ. The x axis shows the real part of the coefficient, while the y axis shows the imaginary part. Both axis are from -1 to 1.
First row: For circuits with depth one - one qubit and only one building block SW-, the Fourier coefficient c_0 is always zero and corresponding functions cannot be fully approximated (see also the calculation in Example 2).
Second row: A WSW structure with depth one and one qubit with R_Y as the rotation gate leads to real Fourier coefficients (see also Example 3).
Third row: Once we use a WSW structure with depth one on one qubit and $R_Y R_Z$ rotation gates, $c_0 \neq 0$ is reachable and $|\mathrm{Im}\{c_1\}| > 0$, so that corresponding Fourier functions can be fitted.

train the ansätze to minimize the loss for a chosen Fourier functions of degree 1. In Fig. 18 the results are depicted in the first two columns for two different functions. The first function is determined by $c_0 = 0$ and $c_1 = 0.2 + i0.2$ while the second function is given by $c_0 = 0.4$ and $c_1 = 0.2 + i0.2$. This way, both functions are normalized such that no post-processing is necessary. These numerical experiments confirm the calculation because the first row shows that $U_{SW,3}(\boldsymbol{x}, \boldsymbol{\theta})$ does only provide $c_0 = 0$. The ansatz $U_{WSW,1}(x, \boldsymbol{\theta})$ (second row) is not able to exactly fit the function while the ansatz $U_{WSW,2}(x, \boldsymbol{\theta})$ (last row) results in perfect fits. The second tool to analyze the expressive power involves randomly initializing the circuits multiple times; by performing an inverse Fourier transformation, the imaginary and real values for each Fourier coefficient can be obtained. Each initialization is represented by an orange circle in the three last columns of Fig. 18 where the third last column shows c_0, the second last c_1, and the last column c_2. Imaginary values are plotted on the y axis, whereas real values are plotted on the x axis. Note that c_0 can only be real because of Eq. (1). Again, $U_{SW,3}(x, \boldsymbol{\theta})$ misses $c_0 \neq 0$, $U_{WSW,1}(x, \boldsymbol{\theta})$ only provides real values for both reachable Fourier coefficients, and $U_{WSW,2}(x, \boldsymbol{\theta})$ enables both coefficients.

2. Minimum number of qubits and layers

According to the results in Ref. [33], we are also able to capture the fact that every ansatz needs a certain number of layers L and qubits n to be able to, in principle, reach a sufficient learning capability for a certain Fourier degree $d_{\max}$, with $d_{\max} = Ln$. The corresponding numerical results are shown in Fig. 19, where we fit 100 random normalized Fourier functions of degree 6 with different ansätze to determine μ_6 for each ansatz. The left side of Fig. 19 shows two layered ansätze: Both utilize $U^1 = R_Y R_Z$ and a simple, linear entanglement structure with two entanglement layers but different entanglement gates $\{\mathrm{CNOT}, \mathrm{CRX}\}$. The number of qubits in these architectures is increased from 1 to 4

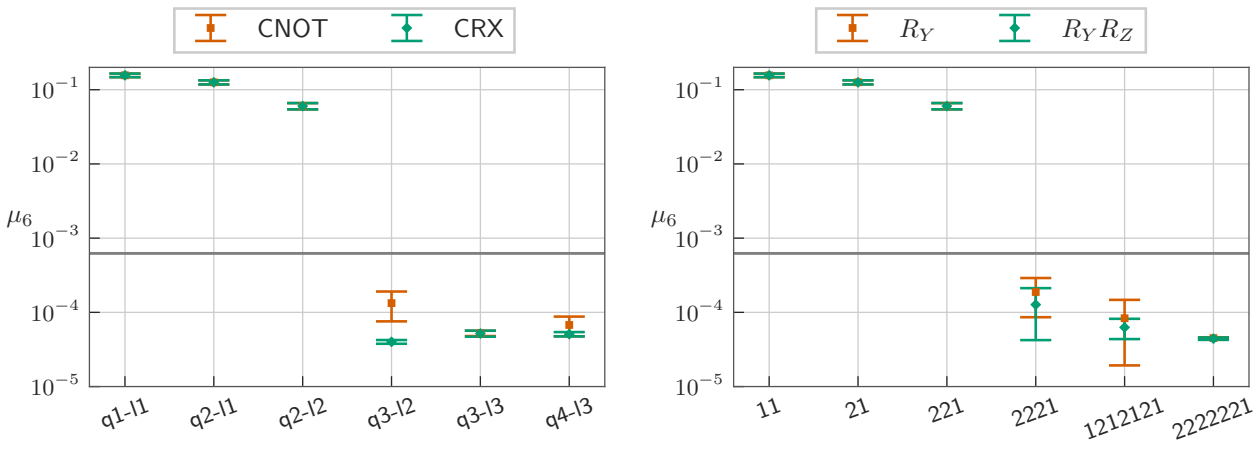


FIG. 19: Comparison of μ_6 for layered (left) and dQNN (right) architectures of different layer and qubit numbers. **Left**: results for layered ansätze with $R_Y R_Z$ single-qubit gates and $\{\text{CNOT}, \text{CRX}\}$ entangling gates for qubit numbers 1 to 4, layer numbers 1 to 3, and two entanglement layers per W; **Right**: results for dQNN ansätze with R_Y and $R_Y R_Z$ single-qubit gates and different numbers of qubits in (hidden) layers on the x axis. The plots show that for these combinations of single-and two-qubit gates, which are known to perform well from the results shown in the main text, the optimal learning capability is reached once the number of qubits and layers matches the degree. Note that the results for μ_6 for the first three layer and qubit combinations on the left and right are nearly indistinguishable on the logarithmic scale.

and the number of layers from 1 to 3. The learning capability together with its 95% confidence interval is plotted in each figure as circles with error bars. Once $nL = d = 6$, the learning capability is enhanced by more than two orders of magnitude. However, it is not improved significantly for ansätze with larger L or n.

The same holds for modified dQNN ansätze as depicted in Fig. 19 on the right side where both ansätze use circuits of the form in Fig. 4c but vary in the single-qubit operation $\{R_Y, R_Y R_Z\}$. The architectures increase the number of input qubits from one to two and the hidden qubits from 0 to 10 (which are split in different layers for some architectures). For example, the notation 221 represents two input and one output qubit, with a hidden layer with two qubits in between.

Appendix G: Analysis of error distribution

We calculate the mean μ and its confidence interval $[\mu - I, \mu + I]$ with $I = c \cdot \widehat{\sigma}_N$ by assuming a student's t distribution with $N = 100$ samples, $c = 1.98$ and use the standard error of the mean (SEM) $\widehat{\sigma}_N$ given by $\widehat{\sigma}_N = \frac{\sigma_N}{\sqrt{N}}$ where $\sigma_N = \sqrt{\frac{\sum_{i=1}^{N}(X_i - \mu)^2}{N-1}}$ is the corrected standard deviation of the sample with μ being the sample's mean over all final MSEs $\{X_i\}_{i=0}^{N}$ obtained for each function in the set.

To further motivate the mean in Eq. (8) from an empirical perspective, we analyze the errors of the runs depicted in Fig. 5 of the result Sec. IV A. Figure 20 shows the error distribution together with its mean and 95% confidence interval.

In Fig. 9, the configuration $R_Y R_Z$ 2 CZ seems to lead to a large confidence interval compared to the other configurations. For this configuration, the mean is 6.8×10^{-4} and the SEM is 6.4×10^{-4}, which is larger than that for $R_Y R_Z$ 1 CRX (1.5×10^{-4}) because an outlier result at $> 6.1 \times 10^{-2}$ increases the variance. The error distribution is depicted in Fig. 21. Furthermore, the logarithmic scale in the result plots and cutting the plots at 10^{-5} lead to seemingly skewed confidence intervals. In contrast, for example, the R_Y 1 CRX configuration has a higher SEM with 1.2×10^{-3}.

We provide two more comparisons of seemingly large confidence intervals from Fig. 6 for degree 12 Fourier functions. In Fig. 22, respectively, Fig. 23, the results for configurations containing three qubits and four layers, respectively, four qubits and three layers are depicted. In both cases, the configurations with errors in lower baskets have lower means and smaller confidence intervals because of less outlier which underlines that logarithmic scaling is one major reason for the seemingly large confidence intervals for those cases.

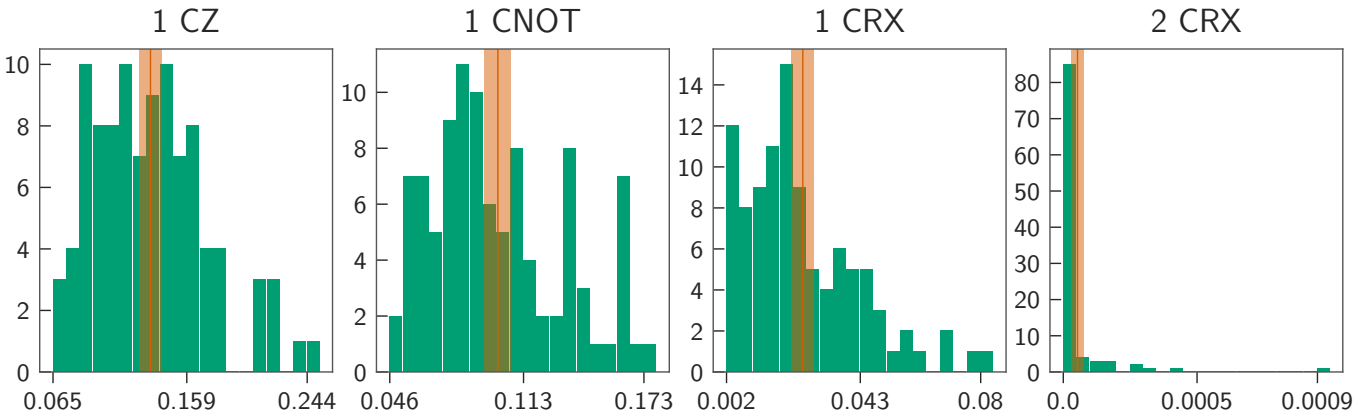


FIG. 20: Detailed error distribution of validation error results in Fig. 5. For each ansatz, the range of all 100 final validation errors is divided into 20 equal baskets. The basket is determined by the minimum and maximum error for each configuration individually resulting in different x axis. The y axis counts the number of errors included in each basket. The mean with its 95% confidence interval is depicted as a red vertical line.

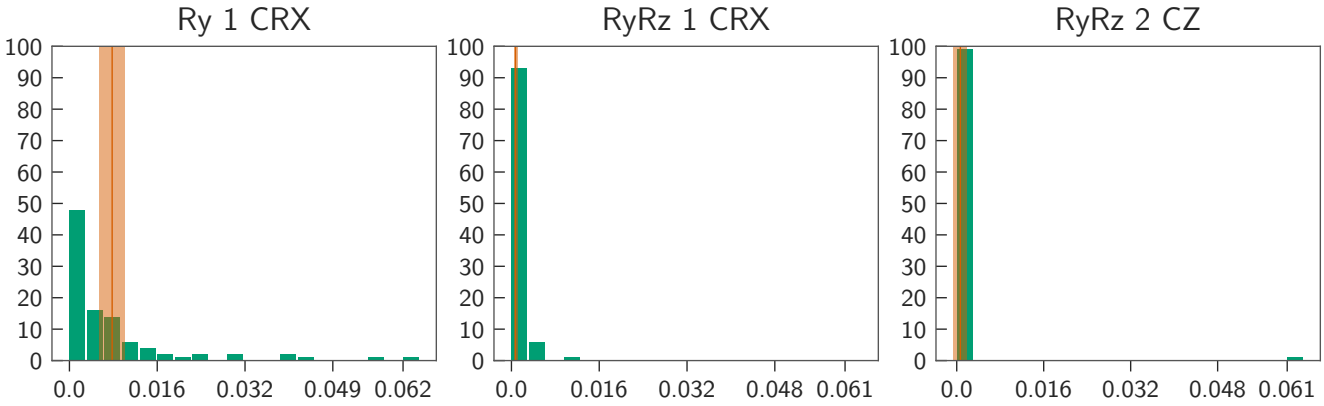


FIG. 21: Detailed error distribution of validation MSE results in Fig. 9 for selected configurations. For each ansatz, the range of all 100 final validation errors is divided into 20 equal baskets. The baskets are the same for all three configurations and are determined by their total minimum and maximal error. The y axis counts the number of errors included in each basket. The mean with its 95% confidence interval is depicted as an orange vertical line.

Appendix H: Analysis of the size of the underlying set of Fourier functions

We investigate the impact of the data set's size on the mean and confidence interval of our result plots. For the same configuration examples as in the previous Appendix, we create subsets by uniformly randomly removing five elements from the results to obtain subsets of sizes from 5 to 100 in steps of five. Calculating the mean and confidence interval for each subset enables us to analyze the dependence of the number of functions in G_d on the learning capability value.

The data leading to the results of Fig. 5 is analyzed in Fig 24 showing a smooth convergence towards the reported mean for set sizes $\gtrsim 25$. In addition, the data for three qubits and four layers configurations, respectively, four qubits and three qubits, of Fig. 6 are analyzed in Fig. 25, respectively, Fig. 26. The convergence for the CRX gate with three entanglement layers is less smooth than for the others which suggests that the size of $|G_d| = 100$ should not be decreased. The other plots show a smooth convergence to the mean.

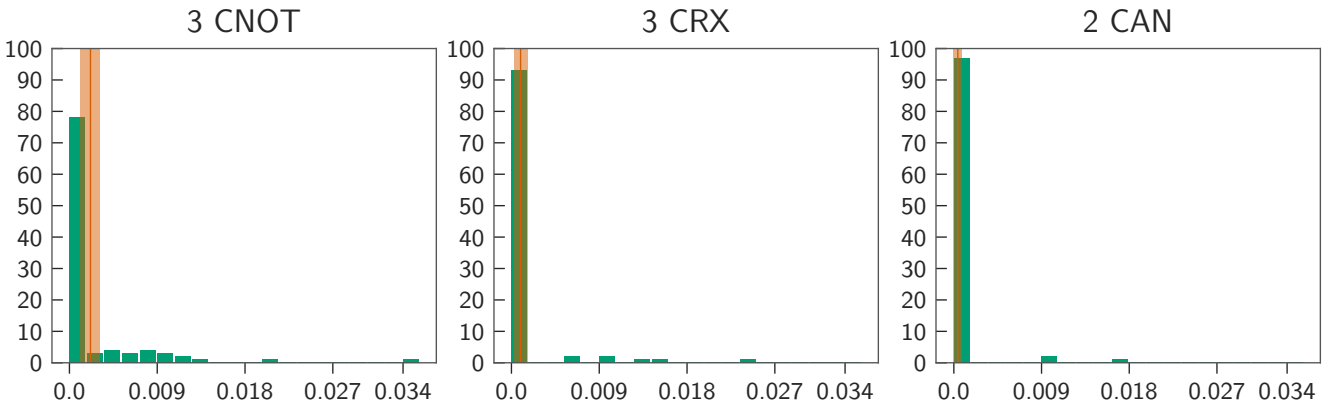


FIG. 22: Detailed error distribution of validation MSE results in Fig. 6 for three qubits and four layers configurations. For each ansatz, the range of all 100 final validation errors is divided into 20 equal baskets. The baskets are the same for all three configurations and are determined by their total minimum and maximal error. The y axis counts the number of errors included in each basket. The mean with its 95% confidence interval is depicted as an orange vertical line.

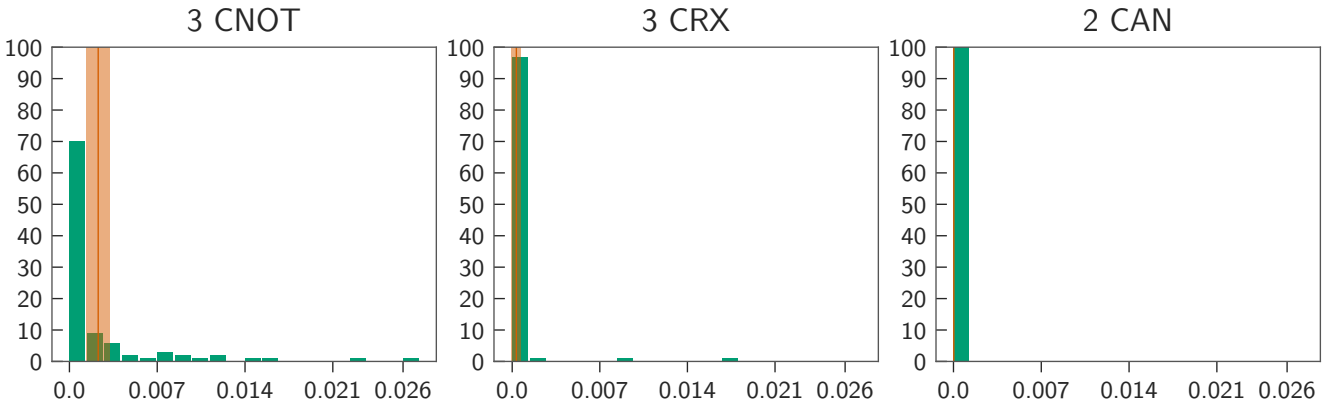


FIG. 23: Detailed error distribution of validation MSE results in Fig. 6 for four qubits and three layers configurations. For each ansatz, the range of all 100 final validation errors is divided into 20 equal baskets. The baskets are the same for all three configurations and are determined by their total minimum and maximal error. The y axis counts the number of errors included in each basket. The mean with its 95% confidence interval is depicted as an orange vertical line.

Appendix I: Barren plateaus

We numerically check if barren plateaus can explain the results of the layered PQCs depicted as blue points in Fig. 6 and the blue points for dQNNs in Fig. 10 (or Fig. 11). The layered WSW ansätze with $n \cdot L = 12$ vary in their number of qubits n and layers L but all consist of single-qubit rotations $R_Y R_Z$ and 3 entanglement layers in a simple, linear structure utilizing CRX entanglement gates. The dQNNs ansätze enabling degree 6 are the following (in the notation of Eq. (9)):

- $[6, 1]$ (max. 7 neighboring qubits),
- $[3, 3, 1]$ (max. 6 neighboring qubits),
- $[2, 2, 2, 1]$ (max. 4 neighboring qubits),
- $[1, 2, 1, 2, 1]$ (max. 3 neighboring qubits),
- $[1, 1, 1, 1, 1, 1, 1]$ (max. 2 neighboring qubits).

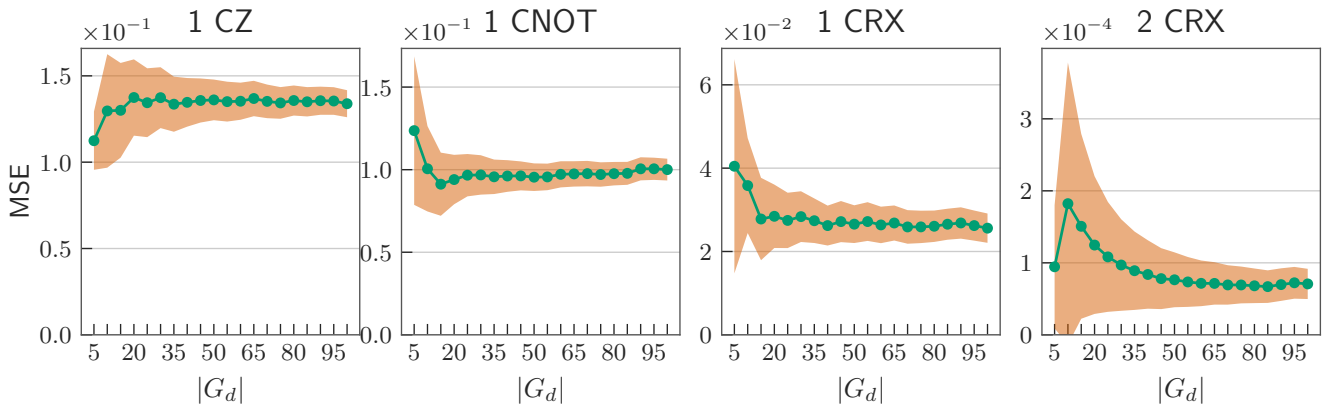


FIG. 24: Mean and confidence interval of validation MSE results for successive smaller set sizes $|G_d|$ for three qubits and four layers configurations of Fig. 5. The plot shows a smooth convergence of the mean for all four configurations for set sizes $\gtrsim 25$.

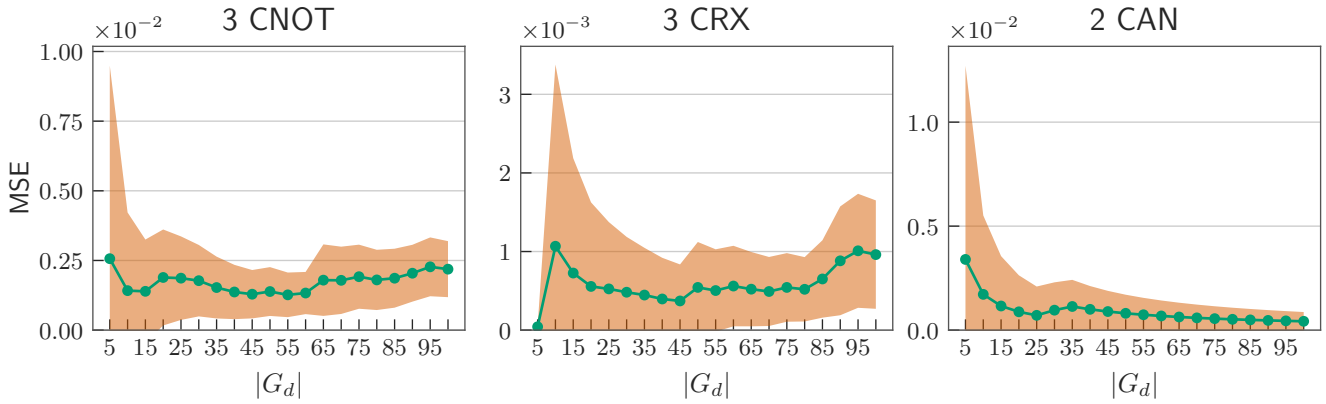


FIG. 25: Mean and confidence interval of validation MSE results for successive smaller set sizes $|G_d|$ for three qubits and four layers configurations of Fig. 6. The plot shows a smooth convergence for 2 CAN. For both other cases the convergence of the mean is less smooth which suggests that the size $|G_d| = 100$ should not be decreased.

All dQNNs utilize single-qubit operations $R_Y R_Z$ and a data reupload scheme with a zero layer on each qubit.

Following the approach of Ref. [46], we initialize every trainable gate randomly in the interval $[0, 2\pi)$ except for the first gate acting on the first qubit. We calculate the gradient based on the mean squared error loss of the complete training data set of 50, respectively, 100, data points for degree 6, respectively, degree 12, Fourier series and determine the variance of these gradients by averaging over the 100 Fourier functions used to calculate the learning capability.

In the left plot of Fig. 27, the results for dQNNs are depicted. No tendency for different choices of hidden qubits is observable indicating that barren plateaus cannot explain the drastically different performances represent by the blue points in Fig. 10 (or Fig. 11).

In the right plot of Fig. 27, the results for layered ansätze are depicted. A clear tendency, in agreement with the literature, can be found showing an exponential decay of the variance of the gradient when increasing the number of qubits (and decreasing the number of layers). However, this does not adequately explain the blue curve in Fig. 6, because the architecture with four qubits (three layers) performs slightly better than the one with three qubits (four layers); the ansatz with six qubits (two layers) performs slightly better than the one with two qubits (six layers); and the ansatz with 12 qubits (one layer) performs slightly better than the one with one qubit (12 layers).

This numerical analysis further supports our claim that the effect of barren plateaus is not sufficient to explain different learning capabilities of different ansätze for dQNNs and layered PQCs.

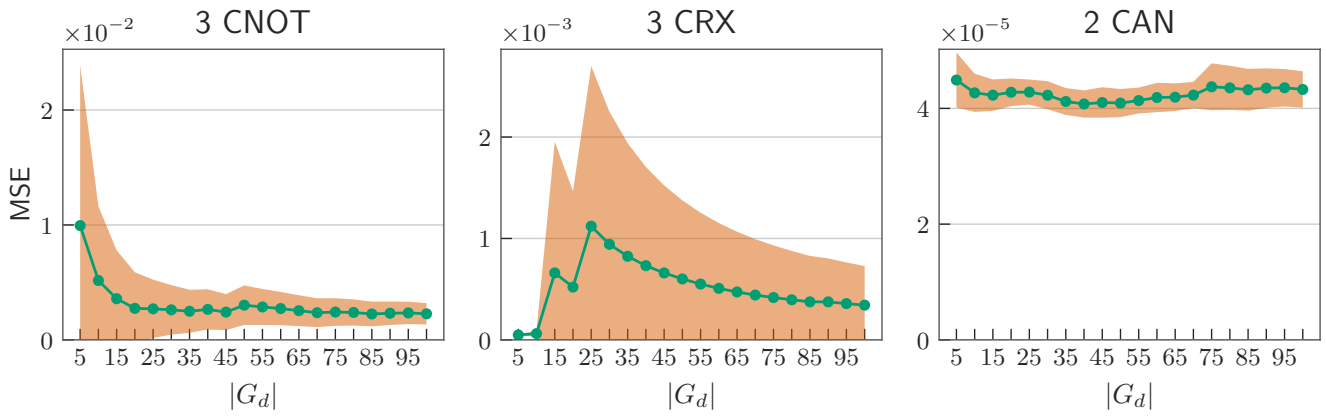


FIG. 26: Mean and confidence interval of validation MSE results for successive smaller set sizes $|G_d|$ for four qubits and three layers configurations of Fig. 6. The plot shows a smooth convergence for all three configurations. However, the shape of the curve for 3 CRX suggests that the size $|G_d| = 100$ should not be decreased.

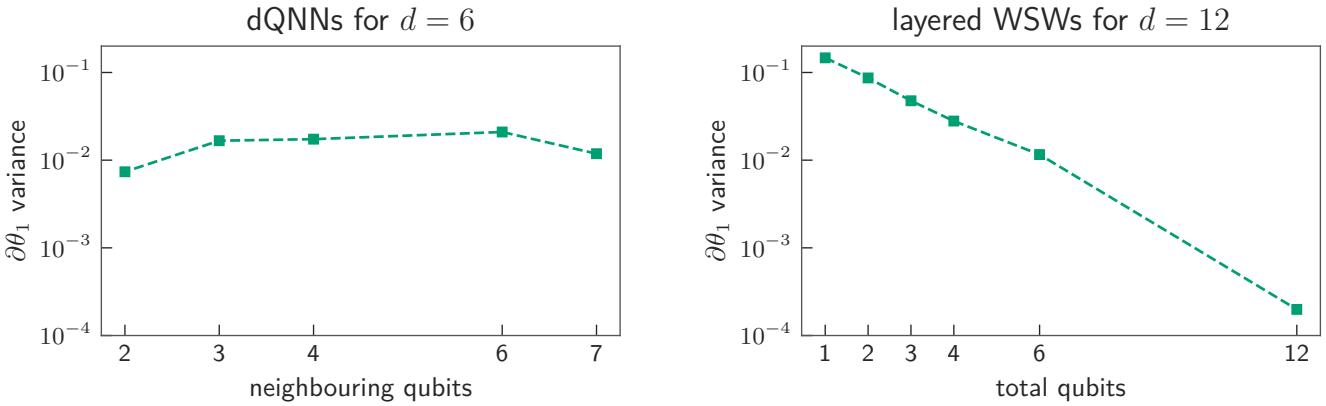


FIG. 27: Investigation of possible barren plateaus via the variance of parameters' gradients [46]. Except for the first gate acting on the first qubit, all gates of the circuit are initialized randomly. The gradient is calculated based on the mean squared error loss of the complete training data set. **Left**: Results for dQNN ansätze enabling degree 6 Fourier series corresponding to the ansätze resulting in the blue curve in Fig. 10 and Fig. 11. **Right**: Results for layered ansätze for degree 12 Fourier series corresponding to the blue curve in Fig. 6.

Appendix J: Additional figures and tables

In this appendix, we provide two more figures and tables. The figures show detailed comparisons of layered ansätze with different entanglement layers and styles for degree 12 in Fig. 28 and for degree 6 in Fig. 29.

Table IV and Table V list the number of single-qubit gates, two-qubit gates, and trainable parameters for the ansätze in Fig. 6 and Fig. 19, respectively.

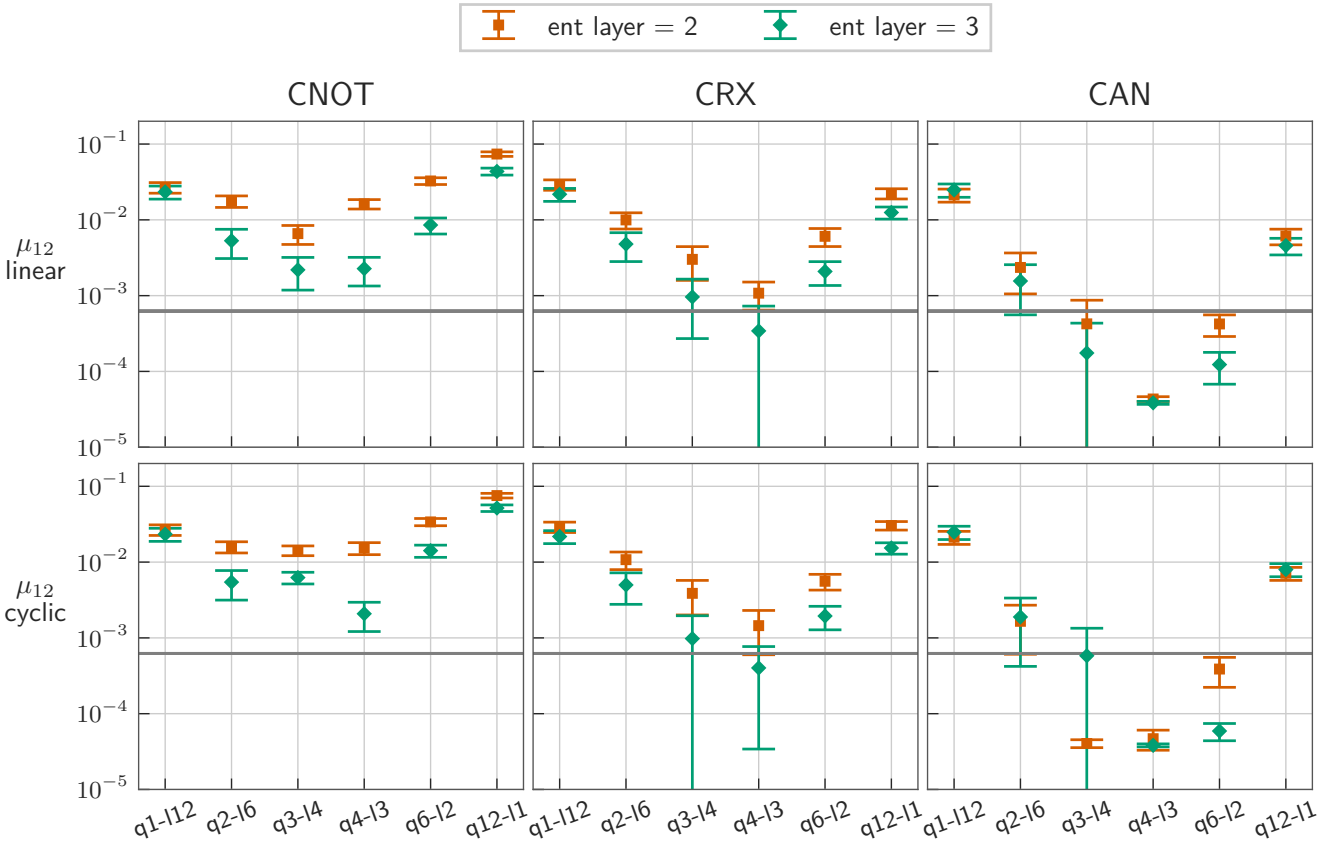


FIG. 28: Results for the learning capability μ_{12} for different WSW ansätze with $R_Y R_Z$ single-qubit gates, simple entangling structure and $\{\text{CNOT}, \text{CRX}, \text{CAN}\}$ entangling gates. The top row shows results for linear entanglement style, while the bottom row shows results for cyclic entanglement (see Fig. 3 for an explanation of the entanglement styles).

$d = 12$		3 CNOT			3 CRX			2 CAN		
n	L	s	t	p	s	t	p	s	t	p
1	12	90	0	78	90	0	78	64	0	52
2	6	96	21	84	96	21	105	68	70	126
3	4	102	30	90	102	30	120	72	80	140
4	**3**	**108**	**36**	**96**	**108**	**36**	**132**	**76**	**88**	**152**
6	2	120	45	108	120	45	153	84	102	174
12	1	156	66	144	156	66	210	108	140	236

TABLE IV: Comparison of number of single-qubit gates s, number of two-qubit gates t, and trainable parameter count p for the different ansätze in Fig. 6. We count one CAN gate as three two-qubit gates. The number of single-qubit gates includes the nonparameterized data-encoding gates.

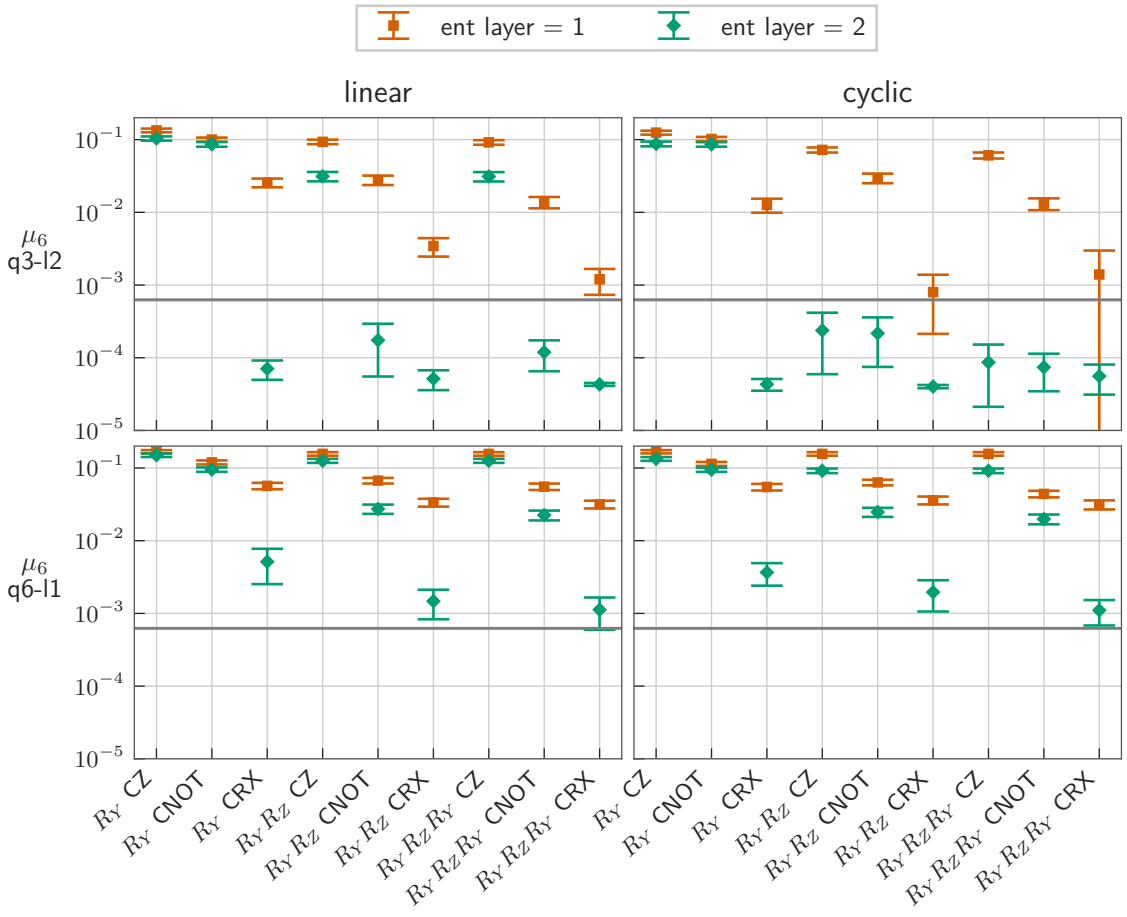


FIG. 29: Comparison of μ_6 for different layered ansätze with simple and linear (left) or cyclic (right) entanglement structure. The top rows shows results for three qubits and two layers, while the bottom row shows results for six qubits and one layer. The number of entanglement layers $\{1, 2\}$ is defined as shown in Fig. 2.

	layered						dQNN					
	CNOT			CR_X			R_Y			$R_Y R_Z$		
$n \times L$	s	t	p	s	t	p	s	t	p	s	t	p
1	9	0	8	9	0	8	4	3	6	9	3	11
2	18	4	16	18	4	20	7	6	12	14	6	18
4	28	6	24	28	6	30	13	18	27	24	18	38
6	**42**	**12**	**36**	**42**	**12**	**48**	**19**	**30**	**43**	**34**	**30**	**58**
9	57	16	48	57	16	64	28	36	55	49	36	76
12	76	24	64	76	24	88	37	66	91	64	66	118

TABLE V: Comparison of number of single-qubit gates s, number of two-qubit gates t, and trainable parameter count p for the different ansätze in Fig. 19. We count one CAN gate in the dQNN ansätze as three two-qubit gates. The number of single-qubit gates includes the non-parameterized data-encoding gates.

Appendix K: Calculations

1. Calculations for circuits representing degree one

Assume $S(x) = R_X(x) = He^{-i\frac{x}{2}D_x}H^\dagger$ where $D_x = \text{diag}(+1,-1)$ is the diagonalized form of σ_X diagonalized by the Hadamard matrix $H^\dagger = H = \frac{1}{\sqrt{2}}(-1)^{i\cdot j}\left|i\right\rangle\left\langle j\right|$. We also assume that measurement is taken in the z basis: $M = \sigma_Z = (-1)^{i\cdot j}\delta^i{}_j\left|i\right\rangle\left\langle j\right| = (-1)^{i\cdot i}\left|i\right\rangle\left\langle i\right|$ and that $W = \begin{pmatrix} W^0_0 & W^0_1 \\ W^1_0 & W^1_1 \end{pmatrix}$ with $W^\dagger = \begin{pmatrix} \overline{W}_0{}^0 & \overline{W}_1{}^0 \\ \overline{W}_0{}^1 & \overline{W}_1{}^1 \end{pmatrix}$ is unitary.

Example 1. The ansatz type $W(\boldsymbol{\theta})S(x)$ on 1 qubit and depth 1 does not learn all Fourier functions of degree 1 since the Fourier coefficient c_0 is equal to zero for all parameters $\boldsymbol{\theta}$.

$$\begin{aligned}
WS\left|0\right\rangle &= \frac{1}{2}(-1)^{0\cdot j}(-1)^{j\cdot l}e^{-i\frac{x}{2}\lambda_j}W^q_i\left|i\right\rangle \\
\left\langle 0\right|S^\dagger W^\dagger MWS\left|0\right\rangle &= \frac{1}{4}(-1)^{0\cdot k}(-1)^{k\cdot l'}(-1)^{0\cdot j}(-1)^{j\cdot l}e^{i\frac{x}{2}(\lambda_k-\lambda_j)}\overline{W}_{i'}{}^{l'}W^i_l M^{i'}_i \\
&= \frac{1}{4}(-1)^{i\cdot i}(-1)^{k\cdot l'}(-1)^{j\cdot l}e^{i\frac{x}{2}(\lambda_k-\lambda_j)}\overline{W}_i{}^{l'}W^i_l \\
c_0\colon k=j\Rightarrow c_0 &= \frac{1}{4}(-1)^{i\cdot i}(-1)^{j\cdot l'}(-1)^{j\cdot l}\overline{W}_i{}^{l'}W^i_l \\
&= \frac{1}{4}(-1)^{j\cdot l'}(-1)^{j\cdot l}(-1)^{i\cdot i}\overline{W}_i{}^{l'}W^i_l \\
&= \frac{1}{4}(1+(-1)^{1\cdot l'}(-1)^{1\cdot l})(-1)^{i\cdot i}\overline{W}_i{}^{l'}W^i_l \\
&= \begin{cases} 0, & \text{if } \ l'\neq l \\ \frac{1}{2}(-1)^{i\cdot i}\overline{W}_i{}^{l}W^i_l, & \text{if } \ l=l \end{cases} \\
&= \frac{1}{2}(\overline{W}_0{}^{l}W^0_l - \overline{W}_1{}^{l}W^1_l) \\
&= \frac{1}{2}(\overline{W}_0{}^{0}W^0_0 + \overline{W}_0{}^{1}W^0_1 - (\overline{W}_1{}^{0}W^1_0 + \overline{W}_1{}^{1}W^1_1)) \\
&= 0
\end{aligned}$$

Example 2. The ansatz $S(x)W(\boldsymbol{\theta})$ does not learn all Fourier functions of degree one since the Fourier coefficient c_0 is equal to zero for all parameters $\boldsymbol{\theta}$.

$$\begin{aligned}
SW\left|0\right\rangle &= \frac{1}{2}(-1)^{l\cdot j}(-1)^{j\cdot q}e^{-i\frac{x}{2}\lambda_j}W^q_0\left|q\right\rangle \\
\left\langle 0\right|W^\dagger S^\dagger MSW\left|0\right\rangle &= \frac{1}{4}(-1)^{i'\cdot k}(-1)^{k\cdot l'}(-1)^{i\cdot j}(-1)^{j\cdot l}e^{i\frac{x}{2}(\lambda_k-\lambda_j)}\overline{W}_{l'}{}^0W^l{}_0M^{i'}{}_i \\
&= \frac{1}{4}(-1)^{i\cdot i}(-1)^{i\cdot k}(-1)^{k\cdot l'}(-1)^{i\cdot j}(-1)^{j\cdot l}e^{i\frac{x}{2}(\lambda_k-\lambda_j)}\overline{W}_{l'}{}^0W^l{}_0 \\
c_0\colon k=j\Rightarrow c_0 &= \frac{1}{4}(-1)^{i\cdot i}(-1)^{i\cdot j}(-1)^{j\cdot l'}(-1)^{i\cdot j}(-1)^{j\cdot l}\overline{W}_{l'}{}^0W^l{}_0 \\
&= \frac{1}{4}(-1)^{i\cdot i}(-1)^{j\cdot l'}(-1)^{j\cdot l}\overline{W}_{l'}{}^0W^l{}_0 \\
&= \frac{1}{4}(1-1)(-1)^{j\cdot l'}(-1)^{j\cdot l}\overline{W}_{l'}{}^0W^l{}_0 \\
&= 0
\end{aligned}$$

Example 3. The ansatz $U = W(\boldsymbol{\theta})S(x)W(\boldsymbol{\theta}) = R_Y(\theta)S(x)R_Y(\theta)$ does not learn all Fourier functions of degree 1 since the coefficients c_0 and c_1 are real for all parameters $\boldsymbol{\theta}$.

$$\begin{aligned}
WSW\left|0\right\rangle &= \frac{1}{2}(-1)^{l_1\cdot j}(-1)^{j\cdot l_0}e^{-i\frac{x}{2}\lambda_j}W^q_{l_1}W^{l_0}_0\left|q\right\rangle \\
\left\langle 0\right|W^\dagger S^\dagger W^\dagger MWSW\left|0\right\rangle &= \frac{1}{4}(-1)^{i\cdot i}(-1)^{l'_1\cdot j}(-1)^{j\cdot l'_0}(-1)^{l_1\cdot j}(-1)^{j\cdot l_0}e^{-i\frac{x}{2}\lambda_j}\overline{W}_{l'_0}{}^0\overline{W}_i{}^{l'_1}W^i_{l_1}W^{l_0}_0
\end{aligned}$$

If we choose $W = R_Y = \begin{pmatrix} \cos(\frac{\Theta}{2}) & -\sin(\frac{\Theta}{2}) \\ \sin(\frac{\Theta}{2}) & \cos(\frac{\Theta}{2}) \end{pmatrix}$, then all entries W_{ij} are real and, hence, the coefficients c_0 and c_1 in the expectation value above are real. Further calculations show that $c_0 = 0$ and $|c_1| = 0.5$ for $W = R_X$ and $W = R_Z$.

2. Calculations for circuits representing higher degrees

Example 4 (one qubit, L layers)**.** We consider the one qubit, L layer case with
$S(x) = e^{-i\frac{x}{2}\sigma_x} = He^{-i\frac{x}{2}\lambda_i} |i\rangle\langle i| H^\dagger$
and $W(\boldsymbol{\theta}) = W^l{}_k |l\rangle\langle k|$

$$\begin{aligned} U(x)|0\rangle &= W^L(\boldsymbol{\theta})S(x)\cdots W^1(\boldsymbol{\theta})S(x)W^0(\boldsymbol{\theta})|0\rangle \\ &= e^{-i\frac{x}{2}(\lambda_{j_1}+\cdots+\lambda_{j_L})} W^q_{j_L}\cdots W^{j_2}_{j_1} W^{j_1}_0 |q\rangle \\ &= e^{-i\frac{x}{2}\Lambda_{\boldsymbol{j}}} W^q_{j_L}\cdots W^{j_1}_0 |q\rangle \\ \Rightarrow \langle 0| U^\dagger(x) M U(x)|0\rangle &= e^{-i\frac{x}{2}(\Lambda_{\boldsymbol{j}}-\Lambda_{\boldsymbol{k}})} M^{\boldsymbol{i}'}_{\boldsymbol{i}} \overline{W}^{\,0}_{k_1}\cdots \overline{W}^{\,k_l}_{i'} W^i_{j_L}\cdots W^{j_1}_0 \end{aligned}$$

such that $\Omega = \{\frac{1}{2}(\Lambda_{\boldsymbol{k}} - \Lambda_{\boldsymbol{j}})\} = [-L, -(L-1), \cdots, -1, 0, 1, \cdots, L-1, L]$.

Example 5 (n qubits, one layer)**.** We consider the n qubit, one layer case with
$S(x) = e^{-i\frac{x}{2}\sigma_x\otimes n} = H^{\otimes n}e^{-i\frac{x}{2}D_x}\otimes\ldots\otimes e^{-i\frac{x}{2}D_x}H^{\dagger\otimes n} = H^{\otimes n}e^{-i\frac{x}{2}(\lambda_{j_1}+\cdots+\lambda_{j_n})}|j_1\cdots j_n\rangle\langle j_1\cdots j_n| H^{\dagger\otimes n}$
and $W(\boldsymbol{\theta}) = W^{l_1\cdots l_n}_{m_1\cdots m_n} |l_1\cdots l_n\rangle\langle m_1\cdots m_n|$

$$\begin{aligned} U(x)|0\cdots 0\rangle &= W^1(\boldsymbol{\theta})S(x)W^0(\boldsymbol{\theta})|0\cdots 0\rangle \\ &= e^{-i\frac{x}{2}(\lambda_{j_1}+\cdots+\lambda_{j_n})} \\ &\qquad W^{q_1\cdots q_n}_{j_1\cdots j_n} W^{j_1\cdots j_n}_{0\cdots 0} |q_1\cdots q_n\rangle \\ &= e^{-i\frac{x}{2}\Lambda_{\boldsymbol{j}}} W^{\boldsymbol{q}}_{\boldsymbol{j}} W^{\boldsymbol{j}}_{\boldsymbol{0}} |\boldsymbol{q}\rangle \\ \Rightarrow \langle 0| U^\dagger(x) M U(x)|0\rangle &= e^{-i\frac{x}{2}(\Lambda_j-\Lambda_{\boldsymbol{k}})} M^{\boldsymbol{i}'}_{\boldsymbol{i}} \overline{W}^{\,\boldsymbol{0}}_{\boldsymbol{k}} \overline{W}^{\,\boldsymbol{k}}_{\boldsymbol{i}'} W^{\boldsymbol{i}}_{\boldsymbol{j}} W^{\boldsymbol{j}}_{\boldsymbol{0}} \end{aligned}$$

such that $\Omega = \{\frac{1}{2}(\Lambda_{\boldsymbol{k}} - \Lambda_{\boldsymbol{j}})\} = [-n, -(n-1), \cdots, -1, 0, 1, \cdots, n-1, n]$.

Example 6 (n qubits, L layer)**.** We consider the n qubit, L layer case with
$S(x) = e^{-i\frac{x}{2}\sigma_x\otimes n} = H^{\otimes n}e^{-i\frac{x}{2}D_x}\otimes\ldots\otimes e^{-i\frac{x}{2}D_x}H^{\dagger\otimes n} = H^{\otimes n}e^{-i\frac{x}{2}(\lambda_{j_1}+\cdots+\lambda_{j_n})}|j_1\cdots j_n\rangle\langle j_1\cdots j_n| H^{\dagger\otimes n}$
and $W(\boldsymbol{\theta}) = W^{l_1\cdots l_n}_{m_1\cdots m_n} |l_1\cdots l_n\rangle\langle m_1\cdots m_n|$

$$\begin{aligned} U(x)|0\cdots 0\rangle &= W^L(\boldsymbol{\theta})S(x)\cdots W^1(\boldsymbol{\theta})S(x)W^0(\boldsymbol{\theta})|0\cdots 0\rangle \\ &= W^L(\boldsymbol{\theta})S(x)\cdots W^2(\boldsymbol{\theta})S(x)e^{-i\frac{x}{2}(\lambda_{j_{11}}+\cdots+\lambda_{j_{1n}})} W^{q_1\cdots q_n}_{j_{11}\cdots j_{1n}} W^{j_{11}\cdots j_{1n}}_{0\cdots 0} |q_1\cdots q_n\rangle \\ &= e^{-i\frac{x}{2}(\lambda_{j_{11}}+\cdots+\lambda_{j_{1n}}+\cdots+\lambda_{j_{L1}}+\cdots+\lambda_{j_{Ln}})} W^{q_1\cdots q_n}_{j_{L1}\cdots j_{Ln}}\cdots W^{j_{21}\cdots j_{2n}}_{j_{11}\cdots j_{1n}} W^{j_{11}\cdots j_{1n}}_{0\cdots 0} |q_{L1}\cdots q_{Ln}\rangle \\ &= e^{-i\frac{x}{2}(\Lambda_{\boldsymbol{j_1}}+\cdots+\Lambda_{\boldsymbol{j_L}})} W^{\boldsymbol{q}}_{\boldsymbol{j_L}}\cdots W^{\boldsymbol{j_1}}_{\boldsymbol{0}} |\boldsymbol{q}\rangle \\ \Rightarrow \langle 0| U^\dagger(x) M U(x)|0\rangle &= e^{-i\frac{x}{2}(\Lambda_{\boldsymbol{j_1}}+\cdots+\Lambda_{\boldsymbol{j_L}}-\Lambda_{\boldsymbol{k_1}}-\cdots-\Lambda_{\boldsymbol{k_L}})} M^{\boldsymbol{i}'}_{\boldsymbol{i}} \overline{W}^{\,\boldsymbol{k_L}}_{\boldsymbol{i}'}\cdots \overline{W}^{\,\boldsymbol{0}}_{\boldsymbol{k_1}} W^{\boldsymbol{i}}_{\boldsymbol{j_L}}\cdots W^{\boldsymbol{j_1}}_{\boldsymbol{0}} \end{aligned}$$

such that $\Omega = \{\frac{1}{2}(\Lambda_{\boldsymbol{k_1}}+\cdots+\Lambda_{\boldsymbol{k_L}} - \Lambda_{\boldsymbol{j_1}}+\cdots+\Lambda_{\boldsymbol{j_L}})\} = [-nL, -(nL-1), \cdots, -1, 0, 1, \cdots, nL-1, nL]$.

Appendix L: Presenting selected circuits

In the following we present the circuits analyzed in Fig. 18 of the preliminary results in Appendix F and in Fig 5 in first part of the results Sec. IV A. The three circuits that lead to the results of Fig. 18 are given in Fig. 30, Fig. 31, and Fig. 32. The three circuits that lead to the results of Fig. 5 are given in Fig. 33, Fig. 34, Fig. 35 and Fig. 36.

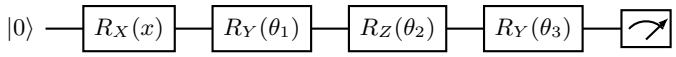


FIG. 30: SW circuit that yields the results in the first row of Fig. 18.

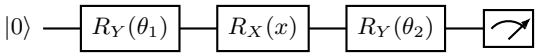


FIG. 31: WSW circuit that yields the results in the second row of Fig. 18.

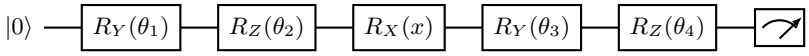


FIG. 32: WSW circuit that yields the results in the third row of Fig. 18.

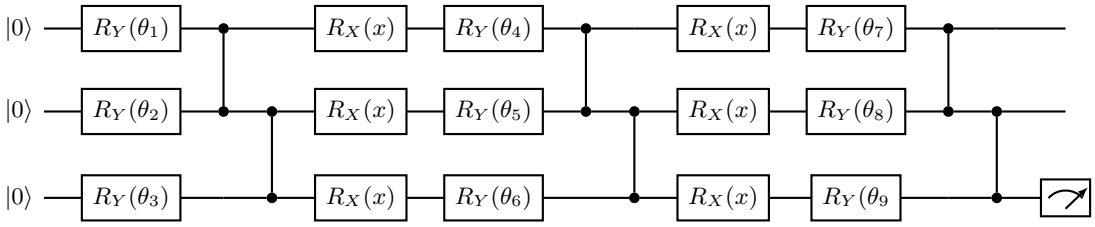


FIG. 33: Circuit that yields the results in the first row of Fig. 5.

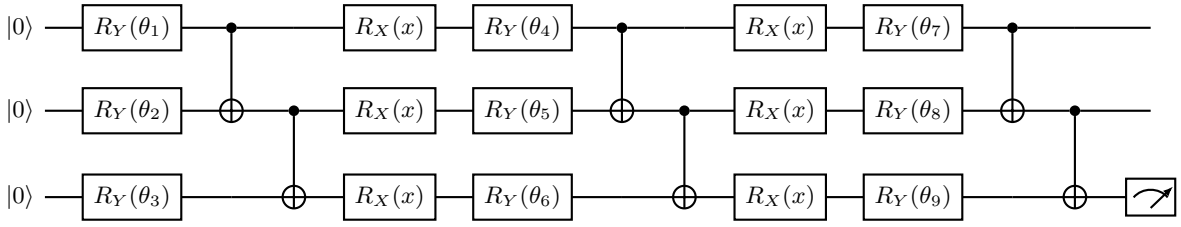


FIG. 34: Circuit that yields the results in the second row of Fig. 5.

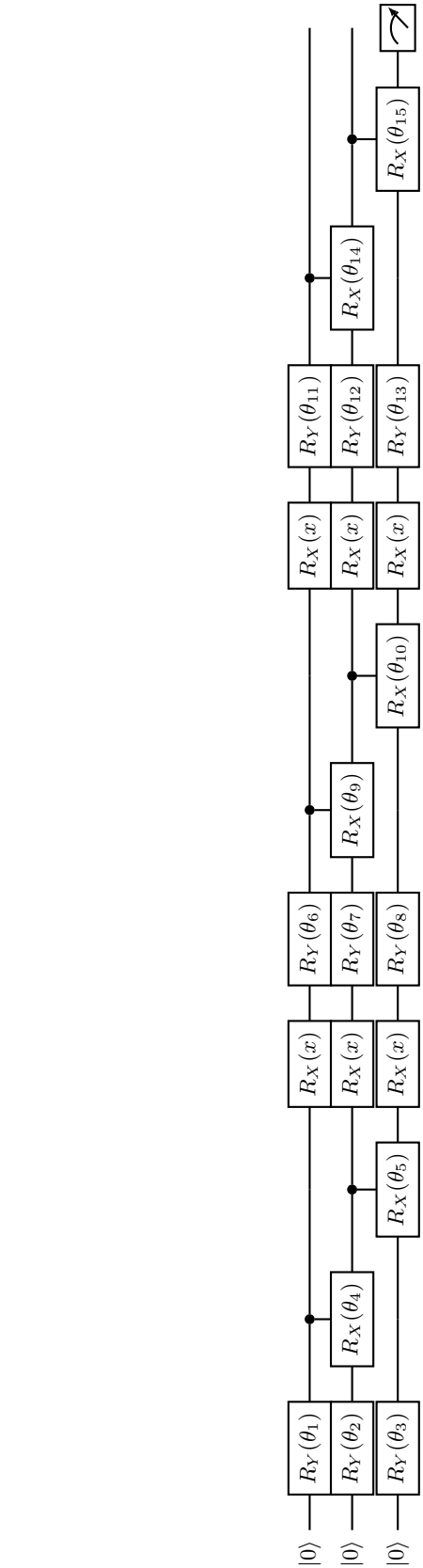


FIG. 35: Circuit that yields the results in the third row of Fig. 5.

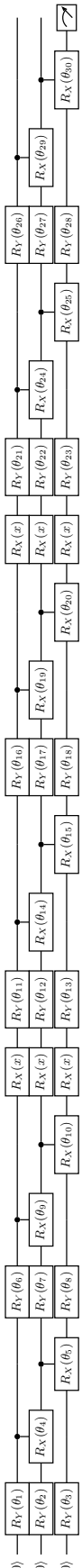


FIG. 36: Circuit that yields the results in the fourth row of Fig. 5.

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

[2] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, Phys. Rev. A **98**, 032309 (2018).

[3] M. Schuld and N. Killoran, Quantum machine learning in feature hilbert spaces, Phys. Rev. Lett. **122**, 040504 (2019).

[4] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature **567**, 209 (2019).

[5] V. Dunjko and P. Wittek, A non-review of Quantum Machine Learning: trends and explorations, Quantum Views **4**, 32 (2020).

[6] M. Schuld and F. Petruccione, *Machine Learning with Quantum Computers* (Springer, 2021).

[7] 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).

[8] J. Preskill, Quantum Computing in the NISQ era and beyond, Quantum **2**, 79 (2018).

[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, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, Noisy intermediate-scale quantum algorithms, Rev. Mod. Phys. **94**, 015004 (2022).

[10] A. Matic, M. Monnet, J. Lorenz, B. Schachtner, and T. Messerer, Quantum-classical convolutional neural networks in radiological image classification, in *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Los Alamitos, CA, USA, 2022) pp. 56–66.

[11] A. Sagingalieva, M. Kordzanganeh, A. Kurkin, A. Melnikov, D. Kuhmistrov, M. Perelshtein, A. Melnikov, A. Skolik, and D. V. Dollen, Hybrid quantum resnet for car classification and its hyperparameter optimization, Quantum Machine Intelligence **5**, 38 (2023).

[12] S. Mangini, A. Marruzzo, M. Piantanida, D. Gerace, D. Bajoni, and C. Macchiavello, Quantum neural network autoencoder and classifier applied to an industrial case study, Quantum Machine Intelligence **4**, 13 (2022).

[13] A. Skolik, S. Jerbi, and V. Dunjko, Quantum agents in the Gym: a variational quantum algorithm for deep Q-learning, Quantum **6**, 720 (2022).

[14] H. Hohenfeld, D. Heimann, F. Wiebe, and F. Kirchner, Quantum deep reinforcement learning for robot navigation tasks, IEEE Access **12**, 87217 (2024).

[15] 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. **6**, 625 (2021).

[16] M. Schuld, I. Sinayskiy, and F. Petruccione, The quest for a quantum neural network, Quantum Inf. Process. **13**, 2567 (2014).

[17] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, The power of quantum neural networks, Nat. Comput. Sci. **1**, 403 (2021).

[18] 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**, 4910 (2022).

[19] 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).

[20] 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).

[21] F. J. Schreiber, J. Eisert, and J. J. Meyer, Classical surrogates for quantum learning models, Phys. Rev. Lett. **131**, 100803 (2023).

[22] Y. Gujju, A. Matsuo, and R. Raymond, Quantum machine learning on near-term quantum devices: Current state of supervised and unsupervised techniques for real-world applications, Phys. Rev. Appl. **21**, 067001 (2024).

[23] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, Nat. Phys. **17**, 1013 (2021).

[24] J. Jäger and R. V. Krems, Universal expressiveness of variational quantum classifiers and quantum kernels for support vector machines, Nat. Commun. **14**, 576 (2023).

[25] S. Jerbi, C. Gyurik, S. C. Marshall, R. Molteni, and V. Dunjko, Shadows of quantum machine learning, Nat. Commun. **15**, 5676 (2024).

[26] J. Bowles, S. Ahmed, and M. Schuld, Better than classical? the subtle art of benchmarking quantum machine learning models, arXiv:2403.07059 (2024).

[27] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549**, 242 (2017).

[28] S. Hadfield, Z. Wang, B. O'Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, From the quantum approximate optimization algorithm to a quantum alternating operator ansatz, Algorithms **12**, 34 (2019).

[29] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, and R. Wolf, Training deep quantum neural networks, Nat. Commun. **11**, 808 (2020).

[30] X. Pan, Z. Lu, W. Wang, Z. Hua, Y. Xu, W. Li, W. Cai, X. Li, H. Wang, Y.-P. Song, *et al.*, Deep quantum neural networks on a superconducting processor, Nat. Commun. **14**, 4006 (2023).

[31] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, Data re-uploading for a universal quantum classifier, Quantum **4**, 226 (2020).

[32] F. J. Gil Vidal and D. O. Theis, Input redundancy for parameterized quantum circuits, Frontiers in Physics **8**, 297 (2020).

[33] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, Phys. Rev. A **103**, 032430 (2021).

[34] W. Li, Z. Lu, and D.-L. Deng, Quantum Neural Network Classifiers: A Tutorial, SciPost Phys. Lect. Notes , 61 (2022).

[35] 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).

[36] S. E. Rasmussen, N. J. S. Loft, T. Baekkegaard, M. Kues, and N. T. Zinner, Reducing the amount of single-qubit rotations in vqe and related algorithms, Adv. Quantum Technol. **3**, 2000063 (2020).

[37] T. Hubregtsen, J. Pichlmeier, P. Stecher, and K. Bertels, Evaluation of parameterized quantum circuits: on the relation between classification accuracy, expressibility, and entangling capability, Quantum Machine Intelligence **3**, 9 (2021).

[38] M. Ballarin, S. Mangini, S. Montangero, C. Macchiavello, and R. Mengoni, Entanglement entropy production in Quantum Neural Networks, Quantum **7**, 1023 (2023).

[39] S. A. Wilkinson and M. J. Hartmann, Evaluating the performance of sigmoid quantum perceptrons in quantum neural networks, arXiv:2208.06198 (2022).

[40] Z. Yu, H. Yao, M. Li, and X. Wang, Power and limitations of single-qubit native quantum neural networks, in *Advances in Neural Information Processing Systems*, Vol. 35 (Curran Associates, Inc., New Orleans, LA, USA, 2022) pp. 27810–27823.

[41] S. Arunachalam and R. de Wolf, Guest column: A survey of quantum learning theory, ACM Sigact News **48**, 41 (2017).

[42] M. C. Caro, Quantum learning theory, `https://mediatum.ub.tum.de/node?id=1634443` (2022).

[43] L. Funcke, T. Hartung, K. Jansen, S. Kühn, and P. Stornati, Dimensional Expressivity Analysis of Parametric Quantum Circuits, Quantum **5**, 422 (2021).

[44] A. Katabarwa, S. Sim, D. E. Koh, and P.-L. Dallaire-Demers, Connecting geometry and performance of two-qubit parameterized quantum circuits, Quantum **6**, 782 (2022).

[45] E. R. Anschuetz and B. T. Kiani, Quantum variational algorithms are swamped with traps, Nat. Commun. **13**, 7760 (2022).

[46] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nat. Commun. **9**, 4812 (2018).

[47] 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).

[48] C. Ortiz Marrero, M. Kieferová, and N. Wiebe, Entanglement-induced barren plateaus, PRX Quantum **2**, 040316 (2021).

[49] Z. Holmes, A. Arrasmith, B. Yan, P. J. Coles, A. Albrecht, and A. T. Sornborger, Barren plateaus preclude learning scramblers, Phys. Rev. Lett. **126**, 190501 (2021).

[50] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, PRX Quantum **3**, 010313 (2022).

[51] K. Sharma, M. Cerezo, L. Cincio, and P. J. Coles, Trainability of dissipative perceptron-based quantum neural networks, Phys. Rev. Lett. **128**, 180505 (2022).

[52] 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).

[53] C.-C. Chen, M. Watabe, K. Shiba, M. Sogabe, K. Sakamoto, and T. Sogabe, On the expressibility and overfitting of quantum circuit learning, ACM Transactions on Quantum Computing **2**, 1 (2021).

[54] Y. Du, Z. Tu, X. Yuan, and D. Tao, Efficient measure for the expressivity of variational quantum algorithms, Physical Review Letters **128**, 080506 (2022).

[55] K. Bu, D. E. Koh, L. Li, Q. Luo, and Y. Zhang, Statistical complexity of quantum circuits, Physical Review A **105**, 062431 (2022).

[56] M. C. Caro, H.-Y. Huang, N. Ezzell, J. Gibbs, A. T. Sornborger, L. Cincio, P. J. Coles, and Z. Holmes, Out-of-distribution generalization for learning quantum dynamics, Nat. Commun. **14**, 3751 (2023).

[57] T. Haug and M. S. Kim, Generalization of quantum machine learning models using quantum fisher information metric, Physical Review Letters **133**, 050603 (2024).

[58] M. C. Caro, E. Gil-Fuster, J. J. Meyer, J. Eisert, and R. Sweke, Encoding-dependent generalization bounds for parametrized quantum circuits, Quantum **5**, 582 (2021).

[59] E. Peters and M. Schuld, Generalization despite overfitting in quantum machine learning models, Quantum **7**, 1210 (2023).

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

[61] L. G. Wright and P. L. McMahon, The capacity of quantum neural networks, in *CLEO: Science and Innovations* (Optica Publishing Group, 2020) pp. JM4G–5.

[62] D. J. C. MacKay, *Information theory, inference and learning algorithms* (Cambridge university press, 2003).

[63] G. Friedland, A. Metere, and M. Krell, A practical approach to sizing neural networks, arXiv:1810.02328 (2018).

[64] M. Lewenstein, A. Gratsea, A. Riera-Campeny, A. Aloy, V. Kasper, and A. Sanpera, Storage capacity and learning capability of quantum neural networks, Quantum Science and Technology **6**, 045002 (2021).

[65] S. Kullback and R. A. Leibler, On Information and Sufficiency, The Annals of Mathematical Statistics **22**, 79 (1951).

[66] K. Nakaji and N. Yamamoto, Expressibility of the alternating layered ansatz for quantum computation, Quantum **5**, 434 (2021).

[67] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, Expressive power of parametrized quantum circuits, Phys. Rev. Res. **2**, 033125 (2020).

[68] Y. Wu, J. Yao, P. Zhang, and H. Zhai, Expressivity of quantum neural networks, Phys. Rev. Res. **3**, L032049 (2021).

[69] T. Haug, K. Bharti, and M. S. Kim, Capacity and quantum geometry of parametrized quantum circuits, PRX Quantum **2**, 040309 (2021).

[70] K. Beer, D. List, G. Mueller, T. J. Osborne, and C. Struckmann, Training quantum neural networks on nisq devices, arXiv:2104.06081 (2021).

[71] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, Circuit-centric quantum classifiers, Phys. Rev. A **101**, 032308 (2020).

[72] E. Fontana, I. Rungger, R. Duncan, and C. Cîrstoiu, Spectral analysis for noise diagnostics and filter-based digital error mitigation, arXiv:2206.08811 (2022).

[73] E. Fontana, I. Rungger, R. Duncan, and C. Cîrstoiu, Efficient recovery of variational quantum algorithms landscapes using classical signal processing, arXiv:2208.05958 (2022).

[74] Y. Liao and J. Zhan, Expressibility-enhancing strategies for quantum neural networks, arXiv:2211.12670 (2022).

[75] J. Wen, Z. Huang, D. Cai, and L. Qian, Enhancing the expressivity of quantum neural networks with residual connections, Communications Physics **7**, 220 (2024).

[76] B. Casas and A. Cervera-Lierta, Multidimensional fourier series with quantum circuits, Phys. Rev. A **107**, 062612 (2023).

[77] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, Adaptive Computation and Machine Learning Series (MIT Press, 2012).

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

[79] C. M. Bishop, *Pattern Recognition and Machine Learning*, Information Science and Statistics (MTM, 2023).

[80] T. Elsken, J. H. Metzen, and F. Hutter, Neural architecture search: A survey, Journal of Machine Learning Research **20**, 1 (2019).

[81] S. Jerbi, C. Gyurik, S. Marshall, H. Briegel, and V. Dunjko, Parametrized quantum policies for reinforcement learning, in *Advances in Neural Information Processing Systems*, Vol. 34 (Curran Associates, Inc., Online, 2021) pp. 28362–28375.

[82] G. E. Crooks, Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition, arXiv:1905.13311 (2019).

[83] M. Broughton, G. Verdon, T. McCourt, A. J. Martinez, J. H. Yoo, S. V. Isakov, P. Massey, R. Halavati, M. Y. Niu, A. Zlokapa, E. Peters, O. Lockwood, A. Skolik, S. Jerbi, V. Dunjko, M. Leib, M. Streif, D. Von Dollen, H. Chen, S. Cao, R. Wiersema, H.-Y. Huang, J. R. McClean, R. Babbush, S. Boixo, D. Bacon, A. K. Ho, H. Neven, and M. Mohseni, Tensorflow quantum: A software framework for quantum machine learning, arXiv:2003.02989 (2020).

[84] Quantum AI team and collaborators, Qsim, 10.5281/zenodo.4023103 (2020), zenodo.

[85] CirqDevelopers, Cirq, 10.5281/zenodo.6599601 (2022), zenodo.

[86] R.-E. Plessix, A review of the adjoint-state method for computing the gradient of a functional with geophysical applications, Geophysical Journal International **167**, 495 (2006).

[87] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, Yao.jl: Extensible, efficient framework for quantum algorithm design, Quantum **4**, 341 (2020).

[88] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980 (2014).

[89] F. Gaitan, Finding flows of a navier–stokes fluid through quantum computing, npj Quantum Information **6**, 61 (2020).

[90] M. Kiffner and D. Jaksch, Tensor network reduced order models for wall-bounded flows, Phys. Rev. Fluids **8**, 124101 (2023).

[91] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral methods: evolution to complex geometries and applications to fluid dynamics* (Springer Science & Business Media, 2007).

[92] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch, Variational quantum algorithms for nonlinear problems, Phys. Rev. A **101**, 010301(R) (2020).

[93] J. Stokes, J. Izaac, N. Killoran, and G. Carleo, Quantum Natural Gradient, Quantum **4**, 269 (2020).

[94] M. Kobayashi, K. Nakaji, and N. Yamamoto, Overfitting in quantum machine learning and entangling dropout, Quantum Machine Intelligence **4**, 1 (2022).

[95] D. Heimann, Code for reproducing the results (2023), `https://github.com/dfki-ric-quantum/learning_capability`.

[96] C. J. Wood, Special session: Noise characterization and error mitigation in near-term quantum computers, in *2020 IEEE 38th International Conference on Computer Design (ICCD)* (IEEE, Hartford, CT, USA, 2020) pp. 13–16.

[97] K. Georgopoulos, C. Emary, and P. Zuliani, Modeling and simulating the noisy behavior of near-term quantum computers, Phys. Rev. A **104**, 062432 (2021).

[98] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, 2010).

[99] X. Y. Jin, A. Kamal, A. P. Sears, T. Gudmundsen, D. Hover, J. Miloshi, R. Slattery, F. Yan, J. Yoder, T. P. Orlando, S. Gustavsson, and W. D. Oliver, Thermal and residual excited-state population in a 3d transmon qubit, Phys. Rev. Lett. **114**, 240501 (2015).

[100] M. A. Nielsen, A simple formula for the average gate fidelity of a quantum dynamical operation, Physics Letters A **303**, 249 (2002).

[101] A. Gilchrist, N. K. Langford, and M. A. Nielsen, Distance measures to compare real and ideal quantum processes, Phys. Rev. A **71**, 062310 (2005).

[102] C. Moussa, J. N. van Rijn, T. Bäck, and V. Dunjko, Hyperparameter importance of quantum neural networks across small datasets, in *Discovery Science: 25th International Conference, DS 2022, Montpellier, France, October 10–12, 2022, Proceedings* (Springer, 2022) pp. 32–46.