Generative model benchmarks for superconducting qubits

Kathleen E. Hamilton, Eugene F. Dumitrescu, and Raphael C. Pooser
Computer Science and Engineering Division, Oak Ridge National Laboratory, One Bethel Valley Road, Oak Ridge, Tennessee 37831, USA

(Received 5 December 2018; revised manuscript received 4 May 2019; published 18 June 2019; corrected 7 October 2020)

In this work we demonstrate experimentally how generative model training can be used as a benchmark for small (fewer than five qubits) quantum devices. Performance is quantified using three data analytic metrics: the Kullback-Leibler divergence and two adaptations of the F_1 score. Using the 2×2 bars and stripes data set, we train several different circuit constructions for generative modeling with superconducting qubits. By taking hardware connectivity constraints into consideration, we show that sparsely connected shallow circuits outperform denser counterparts on noisy hardware.

DOI: 10.1103/PhysRevA.99.062323

I. INTRODUCTION

The increasing diversity of programmable noisy intermediate-scale quantum (NISQ) devices has exposed the need for a unified set of benchmark tasks which assess application-centric device capabilities. Quantum machine learning (QML) has been presented as a useful tool for benchmarking quantum hardware [1]. Generative model training was recently proposed as a benchmark task [2–4] for NISQ devices. In this work we use nonadversarial training of a generative model to benchmark superconducting qubit devices. This approach to generative modeling requires training of a single quantum circuit, making it more practical for implementation on current devices.

Generative models, such as adversarial networks [5], have recently spurred significant interest in the development of quantum-circuit analogs [6,7] and adversarial quantum-circuit training [8–10]. The quantum-circuit Born machine (QCBM) is a generative model constructed as a quantum circuit [3,4,11]. Numerical simulation of QCBMs, constructed using the hardware efficient circuit *Ansatz* [12] with many (more than ten) entangling layers and trained with nonadversarial methods, using data-driven quantum-circuit learning (DDQCL), introduced in [4] can reproduce several classes of discrete and continuous distributions [3]. Here we utilize the gradient-based DDQCL methods of [3].

In contrast, NISQ devices accumulate errors due to imperfect gates and environmental decoherence effects. As such, we expect the depth of useful NISQ circuits to be limited. After this point, the output becomes random as dictated by the noise. Quantum machine learning–based benchmarking is a practical method to establish the maximal circuit depth. To experimentally test this hypothesis, we train a set of shallow circuits (fewer than three entangling layers) which are deployed on IBM's Toyko chip, which has 20 superconducting qubits. The entangling layers of all circuits considered can be embedded in a two-rung ladder geometry (e.g., IBM's Melbourne chip [13]) ensuring portability of our benchmark.

Guidelines for benchmarking digital QML algorithms have been proposed [14] in terms of the output correctness. For generative models, correctness refers to the model's ability to reproduce the target distribution. Performance is therefore naturally captured by statistical measures describing the similarity of two distributions, such as the Kullback-Leibler divergence and the F_1 score.

We evaluated several QCBM circuits on superconducting qubits accessed through the IBM Quantum Hub cloud interface. The QCBM circuit, training methodology, and performance metrics are described in Sec. II. In Sec. III we discuss the interplay between circuit design and QCBM performance. Noisy qubits are introduced into QCBM training in Sec. III B. While previous experimental results for machine-learning based benchmarks were executed on direct-access ion trap hardware which can implement all-to-all connectivity [4], our results show comparable performance in superconducting qubits as measured by the Kullback-Leibler divergence.

II. QUANTUM-CIRCUIT BORN MACHINES

A parametrized quantum circuit defining a particular variational manifold of quantum states is referred to as an *Ansatz*. In this work, as in [3], QCBM training is performed with circuits inspired by the hardware efficient *Ansatz* originally applied in the context of the variational quantum eigensolver algorithm [12] (see Fig. 1). The bars and stripes (BAS) data set BAS(2,2) contains six (2×2)-pixel black and white striped images. Each image is represented in the computational basis of a four-qubit register by fixing a qubit-pixel mapping and associating black (white) pixels with the states $|0\rangle$ ($|1\rangle$) (see Appendix C). While the entangling design introduced in [3] contains enough complexity to represent the data set, for larger image sizes it can require a high degree of qubit connectivity that is not available on current superconducting devices.

To generate BAS(2,2) we train three different *Ansätze* (shown in Fig. 1) whose entangling layers are illustrated in Fig. 2. Each circuit is defined on a four-qubit register and specified by the number of entangling layers (L) and the number of controlled-NOT (CNOT) gates contained within each entangling layer (d_C). Current hardware's fixed connectivity presents a challenge when mapping arbitrary data sets. The $d_C = 2$ and $d_C = 4$ entangling layers conform to IBM's

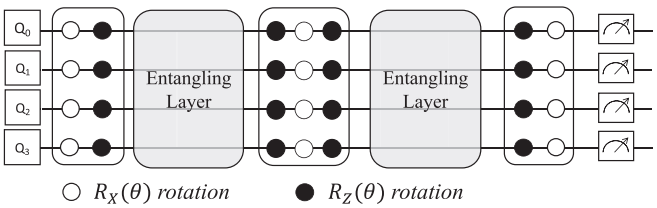

FIG. 1. General circuit construction of a QCBM introduced in [4] based on the hardware efficient variational quantum eigensolver *Ansatz* of [12].

layout, i.e., by restricting CNOT gates to the edges of a four-site square plaquette. The $d_C = 2$ layers are a sparser circuit construction and only take ∼200 ns to apply. As CNOT gates within a single plaquette cannot be simultaneously applied, we decompose the $d_C = 4$ layer into two separate plaquette edge coverings. Thus the $d_C = 4$ circuit takes ∼400 ns to apply, adding additional decoherence compared to $d_C = 2$. Additionally, since plaquettes may be covered in two ways as shown in Fig. 2, alternating the two patterns results in a heterogeneous entangling layers structure for $d_C = 2$. For reference we also use the Chow-Liu tree-based design of [3] to define circuits with $d_C = 3$, though this entangling layer is not embeddable in a single square plaquette.

Many methods exist for training implicit generative models [15]. In this work the rotational parameters are optimized using the algorithm Adam [16]. Overall, we follow the training methods described in [3]: relying on the maximum mean discrepancy (MMD) [17] to define a loss function for circuit training and using the same unbiased estimator to evaluate the gradient. In this work we are modeling a known target distribution: We know it is uniform, we can identify the binary states contained in the distribution, and we may sample classically from the distribution without error.

The target distribution $p(x)$ is fixed and defined by the BAS(2,2) data set. For a given set of rotational parameters, we execute a given QCBM circuit, draw N_{shots} samples, and label this distribution $q(x)$. To compare $q(x)$ to $p(x)$ and quantify the overall QCBM performance, we rely on the Kullback-Leibler (KL) divergence. The KL divergence compares the two sampled distributions $p(x)$ and $q(x)$ by computing the density ratio $p(x_i)/q(x_i)$ of individual states,

$$D(p|q) = \sum_i p(x_i) \ln\left(\frac{p(x_i)}{q(x_i)}\right). \tag{1}$$

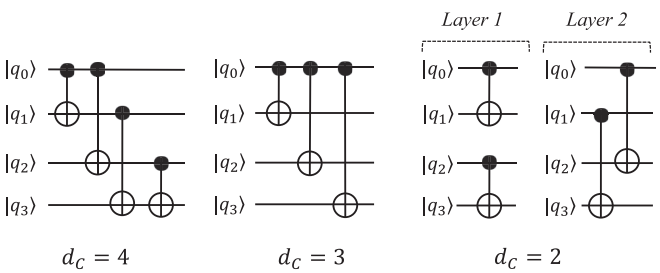

FIG. 2. The CNOT gate sets used to define individual entangling layers. The $d_C = 3$ entangling is the Chow-Liu tree-based design introduced in [3].

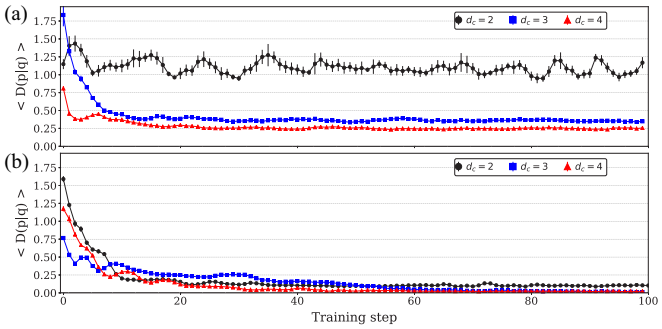

FIG. 3. The KL divergence as a function of training step using $N_{\text{shots}} = 1024$ during training for (a) $L = 1$ and (b) $L = 2$ entangling layers.

As $p(x_i)/q(x_i) \to 1$, $D(p|q) \to 0$, but $D(p|q)$ diverges if $p(x_i) \neq 0$ and $q(x_i) = 0$.

In addition, the performance metric known as the F_1 score [18] can be used. We modify the F_1 score to define an individual value assigned to each BAS(n, m) state and treat the data set as a $2^m + 2^n - 2$ class system. This metric is analogous to measuring the fidelity of each state and we use it to gain insight into how well each circuit *Ansatz* can learn the states of the BAS(n, m) system. The metric is complementary to $D(p|q)$, giving insight into which eigenstates of the distribution are responsible for high KL values. Further details are given in Appendix A.

We note that the number of samples drawn from a circuit during training can be different from the number of samples taken when evaluating performance metrics. When evaluating the KL divergence, we keep the number of shots fixed at $N_{\text{shots}} = 2048$.

III. RESULTS

We first use numerical simulation to train each QCBM in order to estimate how well the target distribution can be learned in the absence of noise. Circuits were constructed using the entangling layers shown in Fig. 2 and trained using the QASM simulator available in IBM Qiskit-Aer. We limit the number of entangling layers to $L = 2$, for a total of six circuits. Each circuit is trained for 100 steps of Adam with learning rate $\alpha = 0.2$ and decay rates ($\beta_1 = 0.9$ and $\beta_2 = 0.999$). The MMD loss function is calculated using Gaussian kernels with $\sigma = 0.1$. Figure 3 shows the overall performance of the three circuit *Ansätze* with noiseless qubits for $L = 1, 2$ when $N_{\text{shots}} = 1024$ shots are drawn during training. For each set of rotational parameters, we evaluate the KL divergence of a given circuit ten times at every training step with $N_{\text{shots}} = 2048$ and report the arithmetic mean value of $D(p|q)$.

A. QCBM training with noiseless qubits

For each value of $\{d_C, L, N_{\text{shots}}\}$ a circuit was trained from a random initialization for $\{\theta^{(t=0)}\}$. Tables I and II show that for most circuits the shot size used during training has a modest effect on performance for the same circuit (fixed d_C and L); however, different shot sizes will lead to different trajectories through $\{\theta\}$ space during training. In particular,

TABLE I. The min[$\langle D(p|q)\rangle$] for $L = 1$ circuits simulated on noiseless qubits. The mean is calculated over ten independent metric evaluations.

L	N_{shots}	$d_C = 2$	$d_C = 3$	$d_C = 4$
1	512	0.95 ± 0.05	0.33 ± 0.02	0.24 ± 0.01
1	1024	0.93 ± 0.03	0.34 ± 0.01	0.23 ± 0.01
1	2048	0.93 ± 0.03	0.33 ± 0.01	0.23 ± 0.01

the large discrepancy for $(d_C = 3, L = 2)$ between $N_{\text{shots}} = 512$ and $N_{\text{shots}} = 2048$ is most likely due to $\{\theta\}$ getting trapped in a suboptimal minimum. For context, we also trained a nonentangling $(L = 0, d_C = 0)$ circuit. With $N_{\text{shots}} = 1024$ this circuit reached a minimum value of $D(p|q) = 1.0(1)$.

In general, Tables I and II show that increasing the complexity of a circuit by increasing the number of rotational parameters will improve performance. For example, the $(d_C = 2, L = 2)$ (28 rotational parameters) and $(d_C = 4, L = 1)$ (16 rotational parameters) circuits contain the same set of CNOT gates; however, the better performance is measured with the $(d_C = 2, L = 2)$ circuit.

In Fig. 3, training reduces the value of $\langle D(p|q)\rangle$ for the $(d_C = 3, 4; L = 1)$ circuits, while $\langle D(p|q)\rangle$ of the $(d_C = 2, L = 1)$ circuit fluctuates about a quasisteady mean value $\sim$1.1. With qubits being entangled pairwise, this *Ansatz* generates a state manifold of the tensor product of two Bell states, up to local rotations. This tensor product structure lacks the complexity to fully learn and describe all of the BAS(2,2) states. The F_1 score supports this claim. In Appendix B we provide additional results for training with smaller learning rates.

In Fig. 4, the individual F_1 score for each BAS(2,2) state is plotted as a function of training step. For the $(d_C = 2, L = 1)$ circuit, it is clear that the QCBM never learns the states $|1010\rangle$ or $|0101\rangle$.

We deploy the circuits with trained noiseless parameters on IBM's Tokyo chip to evaluate circuit performance in the presence of noise. While we leave more detailed discussion about circuit optimization in the presence of noise to Sec. IV, we show several examples here of how the behavior of $\langle D(p|q)\rangle$ is affected by the addition of noise. Many circuits show a general offset for $\langle D(p|q)\rangle$, but the behavior on noisy qubits can be substantially different from simulation. When the QCBM is actively learning (fewer than 30 training steps), parameter updates which result in large fluctuations on noiseless qubits (see Fig. 5) will only result in small changes in $\langle D(p|q)\rangle$ on noisy qubits. When the QCBM training has converged (more than 60 training steps), $\langle D(p|q)\rangle$ reaches a quasistationary value for

TABLE II. The min[$\langle D(p|q)\rangle$] for $L = 2$ circuits simulated on noiseless qubits. The mean is calculated over ten independent metric evaluations.

L	N_{shots}	$d_C = 2$	$d_C = 3$	$d_C = 4$
2	512	0.013 ± 0.004	0.06 ± 0.01	0.02 ± 0.01
2	1024	0.088 ± 0.008	0.01 ± 0.01	0.02 ± 0.01
2	2048	0.011 ± 0.003	0.13 ± 0.01	0.01 ± 0.01

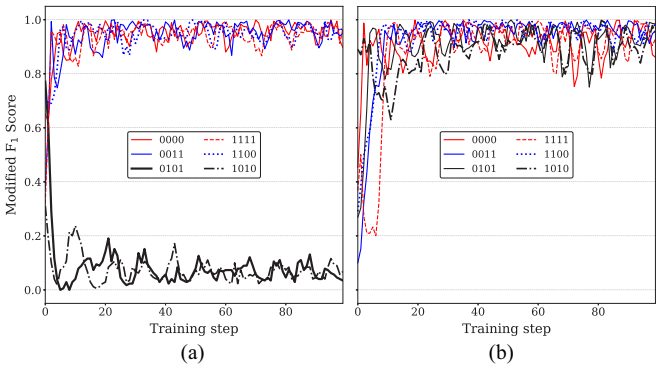

FIG. 4. The F_1 score for each of the six BAS(2,2) states evaluated with $N_{\text{shots}} = 2048$ at each training step for circuits trained with $N_{\text{shots}} = 1024$: (a) the $(d_C = 2, L = 1)$ circuit, (b) the $(d_C = 2, L = 2)$ circuit.

most circuits (cf. Fig. 3). When deployed on hardware, noise can degrade the efficacy of training on noiseless qubits [see Fig. 7(a)]. In contrast, the $(d_C = 2, L = 2)$ or $(d_C = 3, L = 1)$ circuits reach a quasistationary value of $\langle D(p|q)\rangle$ (see Figs. 5 and 6) that is lower than the starting value.

In Tables III and IV we report the best metric values for each d_C, L, and N_{shots} value. The smallest KL value was found with the $(d_C = 2, L = 2)$ circuit. When deployed on hardware, increasing the number of rotational parameters improves performance for the $d_C = 2, 3$ circuits, but not for $d_C = 4$ circuits.

B. QCBM training with noisy qubits

The experiments described in Sec. III explored how closely the value $D(p|q)$ would follow the noiseless learning when measured with noisy qubits. In this section we investigate how well QCBM circuits can be trained with a finite number of steps utilizing noisy qubits. The experiments in this section allow us to explore hardware training within the rotational parameter space.

The goal of these tests is to determine if training a circuit *Ansatz* with noisy qubits can improve the KL metric. In Table IV, the $(d_C = 2, L = 2)$ circuit reaches a minimum

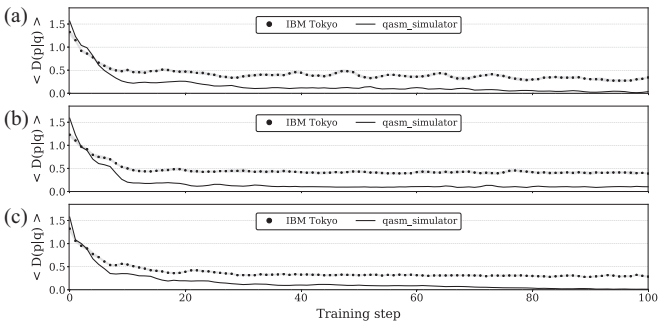

FIG. 5. Comparison of $\langle D(p|q)\rangle$ for ten circuit evaluations of the $(d_C = 2, L = 2)$ circuit *Ansatz* deployed on noiseless qubits (black solid line) and noisy qubits (black circles), trained with (a) $N_{\text{shots}} = 512$, (b) $N_{\text{shots}} = 1024$, and (c) $N_{\text{shots}} = 2048$. The standard deviation of $\langle D(p|q)\rangle$ is shown by the gray shaded regions.

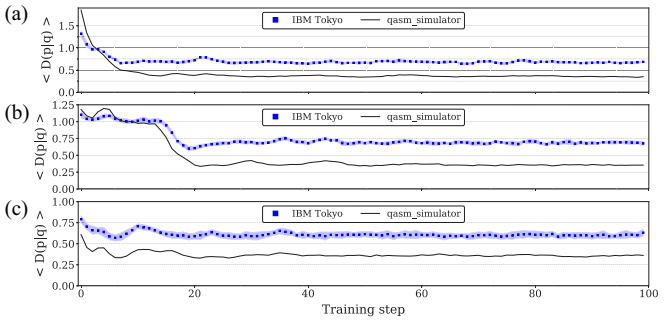

FIG. 6. Comparison of $\langle D(p|q)\rangle$ for ten circuit evaluations of the $(d_C = 3, L = 1)$ circuit *Ansatz* deployed on noiseless qubits (black solid line) and noisy qubits (blue squares), trained with (a) $N_{\text{shots}} = 512$, (b) $N_{\text{shots}} = 1024$, and (c) $N_{\text{shots}} = 2048$. The standard deviation of $\langle D(p|q)\rangle$ is shown by the blue shaded regions.

value of 0.27(1) using θ values trained only with noiseless qubits. In this section the circuit initialization is chosen at equally spaced intervals from the first 60 training steps of each of the curves shown in Fig. 5. This initializes the circuit with a completely random set of parameters ($S = 0$), parameters that have undergone some optimization with Adam ($S = 10, 20, 30$), or parameters that have mostly converged to a localized set of values ($S = 40, 50, 60$). We only train the $(d_C = 2, L = 2)$ circuit, which was able to reach the lowest value of $\langle D(p|q)\rangle$ with pretrained parameters (see Table IV).

As in Sec. III A, the training is done with three shot sizes $N_{\text{shots}} = (512, 1024, 2048)$, but we evaluate KL metric with $N_{\text{shots}} = 2048$. The arithmetic mean value of $D(p|q)$ is calculated from ten circuit evaluations at every training step. We report the following values: the initial mean value $\langle\cdots\rangle_i$, the final value after training $\langle\cdots\rangle_f$, and the minimum KL value observed over training.

For completely random initial parameters ($S = 0, 10$), training with noisy qubits is able to reduce $\langle D(p|q)\rangle$. However, training that begins at later points tends to return higher values of $\langle D(p|q)\rangle$ or shows minimal improvement of $\langle D(p|q)\rangle$ after ten training steps. We discuss the effects of noise and shot size on circuit training in Sec. IV.

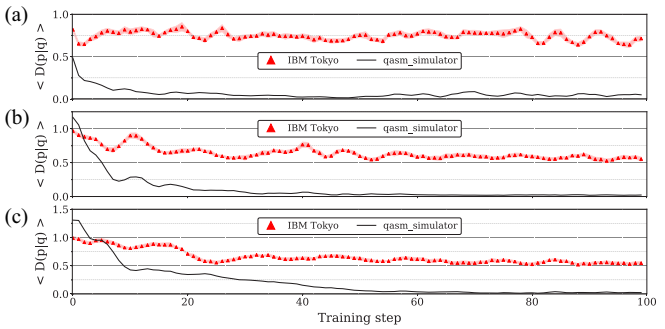

FIG. 7. Comparison of $\langle D(p|q)\rangle$ for ten circuit evaluations of the $(d_C = 4, L = 2)$ circuit *Ansatz* deployed on noiseless qubits (black solid line) and noisy qubits (red triangles), trained with (a) $N_{\text{shots}} = 512$, (b) $N_{\text{shots}} = 1024$, and (c) $N_{\text{shots}} = 2048$. The standard deviation of $\langle D(p|q)\rangle$ is shown by the red shaded regions.

TABLE III. The min$[\langle D(p|q)\rangle]$ for circuits evaluated on IBM's Tokyo chip. The mean is calculated over ten independent metric evaluations.

L	N_{shots}	$d_C = 2$	$d_C = 3$	$d_C = 4$
1	512	0.91 ± 0.01	0.64 ± 0.01	0.59 ± 0.02
1	1024	0.81 ± 0.02	0.60 ± 0.02	0.54 ± 0.01
1	2048	0.86 ± 0.01	0.57 ± 0.02	0.58 ± 0.01

The code used to train QCBM circuits on IBM hardware was adapted from open-source software which is publicly available [19].

IV. DISCUSSION

Effective classical machine learning relies on proper tuning of hyperparameters and avoiding overfitting. By limiting the number of training steps and rotational parameters our models try to fit, we believe that we have avoided circuit *Ansätze* that are too complex for the data set. The hyperparameters of Adam were optimized using noiseless simulation and good rotational parameters were learned for the circuits in this paper, with the exception of the $(d_C = 2, L = 1)$ circuit, which we will exclude from discussion in this section. In this section we will use the Kullback-Leibler divergence to discuss the qualitative changes in performance due to qubit noise and finite sampling.

A. Device noise

When simulated with noiseless qubits, increasing the number of rotational parameters improves the capabilities of the QCBM. The lowest $\langle D(p|q)\rangle \sim 0.01$ values were found for $L = 2$, regardless of d_C value. This same convergence is not seen when circuits are deployed on noisy hardware. Current quantum devices have many sources of noise including qubit decoherence, gate infidelity, and measurement errors. In this study we assume the training will be able to compensate for noise in the single-qubit gates and the limited circuit size will mitigate decoherence effects. In this initial study we have not included any readout error mitigation and designed entangling layers to reduce the noise from two-qubit CNOT gates. With the addition of noise the $(d_C = 2, L = 2)$ circuit returned the lowest value $\langle D(p|q)\rangle = 0.27 \pm 0.02$ using values pretrained via noiseless simulation. Comparable values are found when a circuit is trained on noisy qubits; the lowest value found after training was $\langle D(p|q)\rangle = 0.29 \pm 0.01$ (see Tables V–VII). Understanding how training is affected by the loss function space

TABLE IV. The min$[\langle D(p|q)\rangle]$ for circuits evaluated on IBM's Tokyo chip. The mean is calculated over ten independent metric evaluations.

L	N_{shots}	$d_C = 2$	$d_C = 3$	$d_C = 4$
2	512	0.27 ± 0.02	0.48 ± 0.02	0.64 ± 0.02
2	1024	0.39 ± 0.01	0.39 ± 0.01	0.53 ± 0.01
2	2048	0.28 ± 0.02	0.59 ± 0.01	0.52 ± 0.01

TABLE V. Training on IBM's Tokyo chip ($N_{\text{shots}} = 512$). The mean value is calculated over ten independent metric evaluations.

S	$\langle D(p\|q)\rangle_i$	$\langle D(p\|q)\rangle_f$	min $\langle D(p\|q)\rangle$
0	1.29 ± 0.05	0.39 ± 0.02	0.39 ± 0.02
10	0.48 ± 0.02	0.35 ± 0.02	0.35 ± 0.02
20	0.46 ± 0.02	0.34 ± 0.02	0.34 ± 0.02
30	0.32 ± 0.02	0.34 ± 0.01	0.32 ± 0.02
40	0.41 ± 0.02	0.30 ± 0.02	0.29 ± 0.02
50	0.36 ± 0.02	0.30 ± 0.03	0.30 ± 0.03
60	0.36 ± 0.02	0.31 ± 0.02	0.30 ± 0.01
70	0.32 ± 0.02	0.30 ± 0.01	0.29 ± 0.01
80	0.29 ± 0.02	0.32 ± 0.03	0.29 ± 0.02

is an active area of research for classical machine learning [20,21]. We will use this concept to frame our discussion in this section using τ_U ($\tau_{U'}$) for the loss function space of a noiseless (noisy) circuit.

For a circuit with R rotational parameters, the loss function space τ is defined over the R-dimensional set of all possible parameter values. We will compare the noiseless and noisy qubit performances to draw conclusions about how the addition of noise affects the space τ_U of a single circuit *Ansatz* (cf. Figs. 5–7) and rely on several assumptions made without explicit models of these spaces. First, varying the value of d_C modifies the encoded degrees of entanglement. The local and global optimal parameters of circuits with different d_C and L will therefore be quite different. Also, for circuits with the same values of d_C and L noise will cause the spaces (τ_U, $\tau_{U'}$) to differ.

In the absence of qubit noise the training has largely converged after approximately 50 steps of training. With the weight decay implemented in Adam, this implies that the optimizer is taking small steps within a localized region of τ_U. Our first observation is trivial: Just as the optima of τ_U are expected to be different for different d_C and L values, the minimum that Adam converges to in τ_U is not guaranteed to be a minimum in $\tau_{U'}$ and using Adam to optimize over τ_U instead may drive the system further from the ideal parameters for $\tau_{U'}$. However, small changes in parameters can lead to a good minimum within the space $\tau_{U'}$. Second, the stability of τ_U does not necessarily predict the stability of $\tau_{U'}$. Small changes in parameters can lead to fluctuations in $\langle D(p|q)\rangle$ or possible

TABLE VI. Training on IBM's Tokyo chip ($N_{\text{shots}} = 1024$). The mean value is calculated over ten independent metric evaluations.

S	$\langle D(p\|q)\rangle_i$	$\langle D(p\|q)\rangle_f$	min $\langle D(p\|q)\rangle$
0	1.28 ± 0.05	0.44 ± 0.02	0.43 ± 0.02
10	0.48 ± 0.03	0.44 ± 0.02	0.43 ± 0.01
20	0.49 ± 0.02	0.46 ± 0.02	0.43 ± 0.02
30	0.42 ± 0.02	0.44 ± 0.02	0.42 ± 0.02
40	0.45 ± 0.02	0.50 ± 0.02	0.45 ± 0.02
50	0.44 ± 0.01	0.47 ± 0.02	0.44 ± 0.01
60	0.41 ± 0.04	0.44 ± 0.03	0.41 ± 0.04
70	0.47 ± 0.02	0.46 ± 0.03	0.45 ± 0.03
80	0.45 ± 0.03	0.46 ± 0.02	0.42 ± 0.01

TABLE VII. Training on IBM's Tokyo chip ($N_{\text{shots}} = 2048$). The mean value is calculated over ten independent metric evaluations.

S	$\langle D(p\|q)\rangle_i$	$\langle D(p\|q)\rangle_f$	min $\langle D(p\|q)\rangle$
0	1.30 ± 0.06	0.34 ± 0.02	0.34 ± 0.02
10	0.58 ± 0.01	0.37 ± 0.02	0.37 ± 0.02
20	0.37 ± 0.02	0.35 ± 0.02	0.30 ± 0.02
30	0.30 ± 0.02	0.36 ± 0.02	0.30 ± 0.02
40	0.33 ± 0.01	0.35 ± 0.01	0.33 ± 0.01
50	0.38 ± 0.02	0.43 ± 0.03	0.38 ± 0.02
60	0.29 ± 0.02	0.34 ± 0.02	0.29 ± 0.02
70	0.30 ± 0.03	0.29 ± 0.02	0.29 ± 0.02
80	0.29 ± 0.02	0.30 ± 0.03	0.29 ± 0.02

degradation on noisy qubits (cf. Fig. 7, $N_{\text{shots}} = 512$). On the other hand, the convergence in τ_U to an improved value can be seen in $\tau_{U'}$ (cf. Fig. 5, $N_{\text{shots}} = 1024$); the relative stability of the KL divergence implies that Adam is exploring a region of τ_U which is quasistable in $\tau_{U'}$.

Rotational parameters learned during training are dependent on hardware noise and variability. For all values of N_{shots}, training on hardware improved $\langle D(p|q)\rangle$ when the circuit was initialized with a random set of parameters or pretrained parameters obtained from a low number of Adam steps ($S < 40$). On the other hand, continued training on hardware after the training in the simulator has already converged yields no improvement in $\langle D(p|q)\rangle$. The hardware-trained parameters overall yielded less of an improvement than trained parameters from the simulator due to the inherent noise of the quantum computer. Therefore, interleaving error mitigation steps with each training step is expected to improve performance of hardware-trained parameters.

B. Sampling

In Sec. III we trained multiple circuits from random initial values using noiseless qubits. For each circuit, Adam trains a unique QCBM and defines a unique path in a 16-dimensional (28-dimensional) space for $L = 1$ ($L = 2$) circuits. Within 100 training steps the optimizer is able to find local minima; however, it is not guaranteed to converge to the global optima (see Table II). The noise introduced by smaller N_{shots} values could improve exploration during training.

Sampling a circuit with a high number of shots can improve the KL metric evaluation by reducing the probability of erroneously populated states. However, reducing the sampling error by increasing N_{shots} alone may not be sufficient to counteract the effects of noise on the overall performance of a given circuit.

V. CONCLUSION

As quantum devices become available there is a growing need for a cohesive set of benchmarks quantifying hardware performance. We have observed that while limited connectivity between qubits and noisy gates are not a significant obstacle to circuit learning, our results show that circuit *Ansatz* design can affect generative modeling performance.

There are six possible CNOT gates that can be defined between pixels of the BAS(2,2) images, and the $d_C = 2, 4$ circuits show that the distribution can be modeled by placing CNOT gates between neighboring pairs of pixels. While larger image sizes require long-range correlations, efficient encoding of larger data sets into hardware with fixed qubit connectivity remains an open question (see Appendix C). For the BAS(2,2) data set, adding more CNOT gates to a single qubit in each entangling layer led to minimal increases in performance on noisy qubits. When deployed on hardware, the ($d_C = 2, L = 2$) circuit outperformed all other circuits.

Using a noise-robust stochastic optimizer allowed us to train quantum circuits in the presence of noisy hardware. The provided metrics showed the hardware's capability to reproduce desired probability distributions in the presence of both systematic and statistical noise. We also observed that measurement shot noise can minimally affect the training of a QCBM. However, classical effects such as the optimizer getting trapped in local minima are more significant.

Since the hardware is both noisy and has somewhat sparse connectivity, choosing entangling layers with sufficient sparseness to avoid excessive systematic error while still providing enough complexity to reproduce the distribution represents a trade-off that can be explored using the metrics as a guide. Evaluating the metric for a few entangling layer designs gives insight into which entanglement circuits are good at providing the complexity to represent certain distributions with low noise.

Further development of this benchmark should focus on improvements to the noise-resilience of circuit training which will lead to better estimates of the hardware's innate capabilities. Areas of development include incorporating error mitigation [22] into circuit training to counteract the effects of measurement (readout) and gate errors and exploring other classical optimizers to find the most robust methods for a given hardware device. The benchmark presented in this work is a useful measure of a quantum computer ability to reproduce a discrete probability distribution, and we demonstrated its utility by analyzing the performance of a superconducting quantum computer. While fully noise-robust circuit learning remains an open question, as a benchmark it shows promising avenues for future application and refinement.

ACKNOWLEDGMENTS

This work was supported as part of the ASCR Testbed Pathfinder Program at Oak Ridge National Laboratory under FWP Project No. ERKJ332. This research used quantum computing system resources of the Oak Ridge Leadership Computing Facility. Oak Ridge National Laboratory manages access to the IBM Q System as part of the IBM Q Network.

This manuscript has been authored by UT-Battelle, LLC, under Contract No. DE-AC0500OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for the United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan.

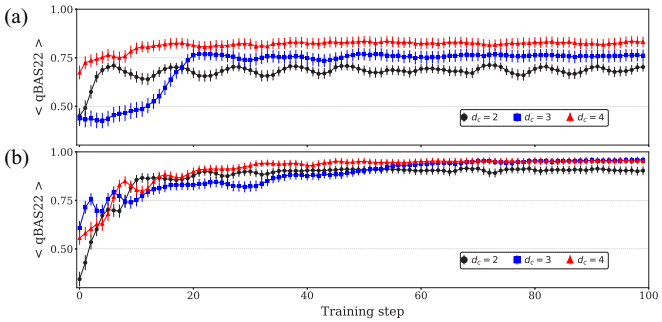

FIG. 8. Weighted mean taken over 11 independent distributions of ⟨qBAS22⟩ scores evaluated at each training step using $N_{\text{qBAS}} = 15$ and 10 000 samples (error bars are the weighted variance). The circuit training and metric evaluation are both done on noiseless qubits: (a) the $L = 1$ circuits and (b) the $L = 2$ circuits.

APPENDIX A: ALTERNATE PERFORMANCE METRICS

In this Appendix we use two alternate metrics to evaluate how well a circuit modeled the BAS(2,2) distribution; both are derived from the F_1 score. First we use the qBAS22 score derived in Ref. [4]; then we introduce a state-based F_1 score.

An advantage to using the qBAS22 metric is that it remains finite even if a BAS state is absent from the sampled distribution. In [4] the terms of the F_1 are defined as follows: The precision is the number of counts returned in BAS states divided by the total number of counts and the recall is the number of BAS states returned divided by the total number of BAS states. The number of samples used to calculate the qBAS22 score (N_{qBAS}) is different from the number of shots (N_{shots}) used to sample from a circuit. For BAS(2,2), $N_{\text{qBAS}} = 15$, which was derived in the Appendix of [4].

We focus on circuits trained with $N_{\text{shots}} = 1024$ and calculate the ⟨qBAS22⟩ scores at each training step for the six circuits introduced in the main text. At each training step we sample from a given circuit 11 times each with $N_{\text{shots}} = 1024$, generating 11 independent distributions $\{q_i(x)\}_{i=1}^{11}$. For a single distribution $q_i(x)$ we generate a set of subsamples $\{s_j\}_{j=1}^{10\,000}$ by drawing 10 000 samples of size $N_{\text{qBAS}} = 15$ (sampling done with replacement). The set $\{s_j\}$ is used to calculate $\mu(\text{qBAS22})_i \pm \sigma_i$, the qBAS22 score of $q_i(x)$. The reported ⟨qBAS22⟩ values are the weighted arithmetic mean of means taken over the 11 independent distributions: $\langle\text{qBAS22}\rangle = \frac{1}{11}\sum_{i=1}^{11}\frac{1}{\sigma_i^2}\mu(\text{qBAS22})_i$. The uncertainty is given by the variance of the mean.

We evaluate this metric for circuits trained with noiseless qubits and for circuits trained on hardware. In Fig. 8 we show the qBAS22 score for circuits that are trained with noiseless qubits and the metric is evaluated with noiseless qubits. The qBAS22 score for the ($d_C = 2, L = 1$) circuit [which does not completely model the entire BAS(2,2) data set] is the lowest performing circuit. Of the six circuits shown in Fig. 8 the ($L = 2; d_C = 3, 4$) circuits have the highest qBAS22 scores (0.96 ± 0.04 and 0.95 ± 0.4, respectively).

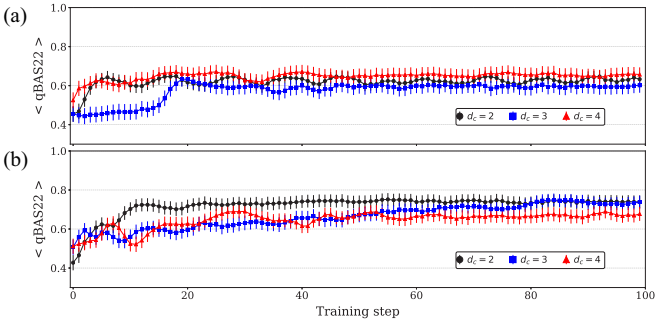

FIG. 9. Weighted mean taken over 11 independent distributions of ⟨qBAS22⟩ scores evaluated at each training step using $N_{\mathrm{qBAS}} = 15$ and 10 000 samples (error bars are the weighted variance). The circuit training is done on noiseless qubits and the metric is evaluated on IBM's Tokyo chip: (a) the $L = 1$ circuits and (b) the $L = 2$ circuits.

However, the device noise strongly affects the $d_c = 3, 4$ circuits. In Fig. 9 we present ⟨qBAS22⟩ scores for circuits trained with noiseless qubits, but evaluate the metric on IBM's Tokyo chip. We see that for $L = 1$ circuits the $d_C = 2$ circuit performs comparably to the $d_C = 3, 4$ circuits, even though this circuit is known to only fit four out of the six BAS(2,2) states. When the circuit size is increased to $L = 2$, the $d_c = 2, 3$ circuits have comparable performance after 100 steps of training (0.75 ± 0.03 and 0.75 ± 0.03, respectively), outperforming the $d_c = 4$ circuit (0.69 ± 0.03). Similar behavior is seen in the KL metric reported in the main text (cf. Table IV).

In Table VIII we present the qBAS22 score evaluated on IBM's Tokyo chip for the ($d_C = 2, L = 2$) circuit trained on hardware. As in Sec. III B, the circuits are pretrained using noiseless simulation for a fixed number of steps and then deployed on IBM's Tokyo hardware to execute ten steps of Adam training. The best performance of a circuit trained on hardware for ten steps of Adam was $\langle \mathrm{qBAS22} \rangle = 0.74 \pm 0.03$.

The qBAS22 metric and the KL metric give a measure of the global performance of a circuit, but there is also a need for local metrics. We adapt a *modified* F_1 score [18] and apply it to the individual BAS(n, m) states to define a metric that measures how well a circuit learns each state and can be applied to uniform or nonuniform discrete distributions. However, it requires that the user specify the exact form

TABLE VIII. The ⟨qBAS22⟩ scores evaluated on IBM's Tokyo chip at each training step for circuits trained on IBM's Tokyo chip.

S	$\langle \mathrm{qBAS22} \rangle_i$	$\langle \mathrm{qBAS22} \rangle_f$	max ⟨qBAS22⟩
0	0.42 ± 0.04	0.74 ± 0.03	0.74 ± 0.03
10	0.71 ± 0.03	0.72 ± 0.03	0.74 ± 0.03
20	0.70 ± 0.03	0.71 ± 0.03	0.72 ± 0.03
30	0.74 ± 0.03	0.73 ± 0.03	0.74 ± 0.03
40	0.73 ± 0.03	0.69 ± 0.04	0.73 ± 0.03
50	0.73 ± 0.03	0.71 ± 0.03	0.73 ± 0.03
60	0.74 ± 0.03	0.72 ± 0.03	0.74 ± 0.03
70	0.72 ± 0.03	0.71 ± 0.03	0.72 ± 0.03
80	0.72 ± 0.03	0.71 ± 0.03	0.73 ± 0.03

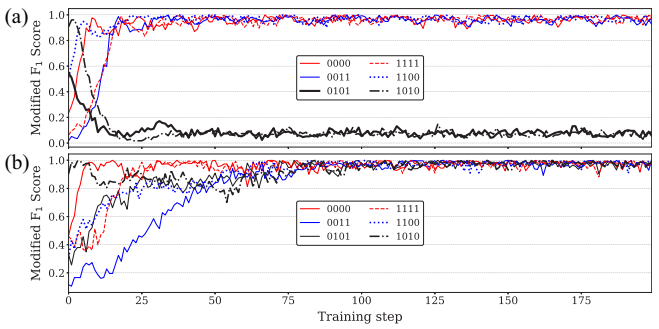

FIG. 10. The F_1 score per BAS state of a circuit trained on a noiseless simulator with 200 steps of Adam and $\alpha = 0.05$: (a) the ($d_C = 2, L = 1$) circuit and (b) the ($d_C = 2, L = 2$) circuit.

of the target distribution. For benchmarking tasks where the performance is measured with regard to a known distribution this is not a problem, but it may limit the usability of the F_1 score metric for future applications.

The modified F_1 score defines the precision and true positive rate of a model with respect to the uniform BAS(2,2) distribution [$p_i = 1/6$ if $|x_i\rangle$ is a BAS(2,2) state]. Device noise (such as readout errors) leads to a number of incorrectly measured states, but in our initial approximation, for each state $|x_i\rangle$ of the BAS data set we define the number of true positives as $\mathrm{TP}(x_i) = q(x_i)$ if $q(x_i) < p(x_i)$ and $\mathrm{TP}(x_i) = p(x_i)$ if $q(x_i) > p(x_i)$. We define the number of false positives and false negatives using the difference $\Delta = |q(x_i) - p(x_i)|$. If $q(x_i) > p(x_i)$ then $\mathrm{FP}(x_i) = \Delta$ and $\mathrm{FN}(x_i) = 0$; if $q(x_i) < p(x_i)$ then $\mathrm{FN}(x_i) = \Delta$ and $\mathrm{FP}(x_i) = 0$. With these values we can define for each state x_i the true positive rate

$$\mathrm{TPR}(x_i) = \frac{\mathrm{TP}(x_i)}{[\mathrm{TP}(x_i) + \mathrm{FN}(x_i)]} \tag{A1}$$

and the precision

$$P(x_i) = \frac{\mathrm{TP}(x_i)}{[\mathrm{TP}(x_i) + \mathrm{FP}(x_i)]}. \tag{A2}$$

The modified F_1 score is defined in terms of the precision and true positive rate,

$$F_1(x_i) = 2\left(\frac{P(x_i) \times \mathrm{TPR}(x_i)}{P(x_i) + \mathrm{TPR}(x_i)}\right). \tag{A3}$$

APPENDIX B: ALTERNATE LEARNING RATES

In this Appendix we highlight the specific case of the ($d_C = 2, L = 1$) circuit. In Fig. 3 the KL value oscillated around $D(p|q) \sim 1.1$ and in Secs. III and IV we argue that this behavior is due to the circuit being overly simplistic and not from a too-large learning rate.

To prove this we randomly initialize the ($d_C = 2, L = 1$) circuit and train it with $N_{\mathrm{shots}} = 1024$ and different learning rates $\alpha = \{0.05, 0.3\}$ and train for 200 steps of Adam. In Figs. 10 and 11 we show the state-by-state F_1 score (evaluated with $N_{\mathrm{shots}} = 2048$). For either learning rate ($\alpha = 0.05, 0.3$) the ($d_C = 2, L = 1$) circuit fails to learn the states $|1010\rangle$ and $|0101\rangle$. In contrast, the ($d_C = 2, L = 2$) circuit is able to learn all six BAS states.

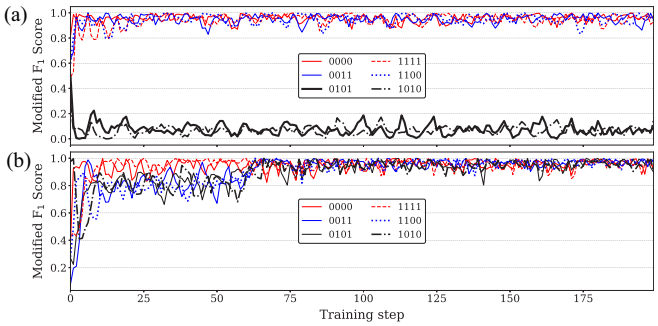

FIG. 11. The F_1 score per BAS state of a circuit trained on a noiseless simulator with 200 steps of Adam and $\alpha = 0.3$: (a) the $(d_C = 2, L = 1)$ circuit and (b) the $(d_C = 2, L = 2)$ circuit.

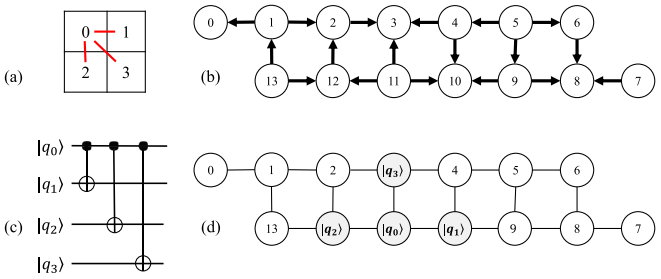

FIG. 12. (a) Pixels of a BAS(2,2) image with the edges of the Chow-Liu tree defined from the mutual information (red). (b) Connectivity graph of IBM's Melbourne chip [13]. (c) The $d_C = 3$ entangling layer defined using the Chow-Liu tree in (a). (d) The $d_C = 3$ layer embedded into IBM's Melbourne chip.

APPENDIX C: CONNECTIVITY, CORRELATION LOCALITY, AND HARDWARE EMBEDDING

We define local or nonlocal connections with respect to the image pixels of the BAS(2,2) data set. There are six possible pairs that can be formed from the four pixels of each image (four local and two nonlocal). The nearest-neighbor pairs of pixels [(0,1),(0,2),(1,3),(2,3)] form the local connections, while the remaining pairs [(0,3),(1,2)] are nonlocal.

If the hardware supports all-to-all connectivity then all local and nonlocal connections can be mapped to CNOT gates and implemented in a single QCBM. With limited qubit connectivity, it is possible to embed two nonlocal connections into hardware but often at the cost of removing local connections. The $d_C = 2, 4$ layers construct QCBMs with four local connections and zero nonlocal connections, whereas the $d_C = 3$ layers construct QCBMs with one nonlocal and two local connections. In Fig. 12 we show the construction and hardware embedding of a $d_C = 3$ entangling layer from the edges of a Chow-Liu tree rooted at pixel 0. Understanding the trade-offs between local or nonlocal connections will be necessary to construct QCBMs that can model larger images or more complicated distributions.

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

[2] A. Perdomo-Ortiz, M. Benedetti, J. Realpe-Gómez, and R. Biswas, Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers, Quantum Sci. Technol. **3**, 030502 (2018).

[3] J.-G. Liu and L. Wang, Differentiable learning of quantum circuit Born machine, Phys. Rev. A **98**, 062324 (2018).

[4] M. Benedetti, D. Garcia-Pintos, O. Perdomo, V. Leyton-Ortega, Y. Nam, and A. Perdomo-Ortiz, A generative modeling approach for benchmarking and training shallow quantum circuits. npj Quantum Information **5**, 45 (2019).

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, in *NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, Canada*, Vol. 2 (MIT Press, Cambridge, MA, 2014), pp. 2672–2680.

[6] S. Lloyd and C. Weedbrook, Quantum Generative Adversarial Learning, Phys. Rev. Lett. **121**, 040502 (2018).

[7] P.-L. Dallaire-Demers and N. Killoran, Quantum generative adversarial networks, Phys. Rev. A **98**, 012324 (2018).

[8] M. Benedetti, E. Grant, L. Wossnig, and S. Severini, Adversarial quantum circuit learning for pure state approximation, New J. Phys. **21**, 043023 (2019).

[9] L. Hu, S.-H. Wu, W. Cai, Y. Ma, X. Mu, Y. Xu, H. Wang, Y. Song, D.-L. Deng, C.-L. Zou, and L. Sun, Quantum generative adversarial learning in a superconducting quantum circuit, Sci. Adv. **5**, eaav2761 (2019).

[10] J. Zeng, Y. Wu, J.-G. Liu, L. Wang, and J. Hu, Learning and inference on generative adversarial quantum circuits, Phys. Rev. A **99**, 052306 (2019).

[11] S. Cheng, J. Chen, and L. Wang, Information perspective to probabilistic modeling: Boltzmann machines versus Born machines, Entropy **20**, 583 (2018).

[12] 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 (London) **549**, 242 (2017).

[13] IBM Q Team, 16 qubit backend: IBM Q team, IBM Q 16 Melbourne backend specification version 1.1.0, 2018, retrieved from https://ibm.biz/qiskit-melbourne.

[14] K. Michielsen, M. Nocon, D. Willsch, F. Jin, T. Lippert, and H. De Raedt, Benchmarking gate-based quantum computers, Comput. Phys. Commun. **220**, 44 (2017).

[15] S. Mohamed and B. Lakshminarayanan, Learning in implicit generative models, arXiv:1610.03483.

[16] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.

[17] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, A kernel two-sample test, Journal of Machine Learning Research **13**, 723 (2012).

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

[19] J.-G. Liu and L. Wang, Quantum circuit Born machine, https://github.com/GiggleLiu/QuantumCircuitBornMachine.

[20] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, Visualizing the loss landscape of neural nets, Advances in Neural Information Processing Systems, 6389 (2018).

[21] P. Chaudhari and S. Soatto, Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks, in *2018 Information Theory and Applications Workshop (ITA)* (IEEE, 2018), pp. 1–10.

[22] A. Kandala, K. Temme, A. D. Córcoles, A. Mezzacapo, J. M. Chow, and J. M. Gambetta, Error mitigation extends the computational reach of a noisy quantum processor, Nature (London) **567**, 491 (2019).

Correction: The previously published Figure 1 contained an error and has been replaced.