Do Quantum Circuit Born Machines Generalize?

Kaitlin Gili,[1,2] Mohamed Hibat-Allah,[2,3,4] Marta Mauri,[2] Chris Ballance,[1] and Alejandro Perdomo-Ortiz[2,*]

[1] *University of Oxford, Oxford, United Kingdom OX1 2JD*
[2] *Zapata Computing Canada Inc., 325 Front St W, Toronto, ON, Canada M5V 2Y1*
[3] *Vector Institute, MaRS Centre, Toronto, ON, Canada M5G 1M1*
[4] *Department of Physics and Astronomy, University of Waterloo, Waterloo, ON, Canada N2L 3G1*

(Dated: May 16, 2023)

In recent proposals of quantum circuit models for generative tasks, the discussion about their performance has been limited to their ability to reproduce a known target distribution. For example, expressive model families such as Quantum Circuit Born Machines (QCBMs) have been almost entirely evaluated on their capability to learn a given target distribution with high accuracy. While this aspect may be ideal for some tasks, it limits the scope of a generative model's assessment to its ability to *memorize* data rather than *generalize*. As a result, there has been little understanding of a model's generalization performance and the relation between such capability and the resource requirements, e.g., the circuit depth and the amount of training data. In this work, we leverage upon a recently proposed generalization evaluation framework to begin addressing this knowledge gap. We first investigate the QCBM's learning process of a cardinality-constrained distribution and see an increase in generalization performance while increasing the circuit depth. In the 12-qubit example presented here, we observe that with as few as 30% of the valid data in the training set, the QCBM exhibits the best generalization performance toward generating unseen and valid data. Lastly, we assess the QCBM's ability to generalize not only to valid samples, but to high-quality bitstrings distributed according to an adequately re-weighted distribution. We see that the QCBM is able to effectively learn the reweighted dataset and generate unseen samples with higher quality than those in the training set. To the best of our knowledge, this is the first work in the literature that presents the QCBM's generalization performance as an integral evaluation metric for quantum generative models, and demonstrates the QCBM's ability to generalize to high-quality, desired novel samples.

I. Introduction

From classical machine learning (ML), we have seen remarkable applications across a wide variety of industries including image classification and generation [1, 2], language processing [3], and constructing complex recommendation systems [4]. As such, it has become an active area of research to understand how quantum computers may enhance, or outperform, these classical algorithms.

In the pursuit of practical quantum advantage on classical data, unsupervised generative modeling tasks stand out as one of the most promising application candidates given their increased complexity compared to supervised ML tasks, and therefore a better target for seeking advantage with near-term quantum computers [5]. Many quantum generative models have been proposed with very little discussion around their learning potential in the context of generalization [6], despite its importance. One of the most popular quantum circuit families for generative tasks, known as Quantum Circuit Born Machines (QCBMs) [7], have demonstrated remarkable capabilities in modeling target distributions for both toy and real-world datasets [7–16]. It has been shown that these models have the ability to express distributions that are difficult for classical probabilistic models [17–21], further motivating an investigation into these models for quantum advantage applications.

When it comes to generalization, discriminative models have been the primary focus of research - both in the classical and quantum domain [22–24]. These type of generalization studies are based on the so-called generalization error, which describes how well the model is able to classify unseen data after learning with a labelled training set. While the metrics for measuring generalization within classical and quantum discriminative tasks are reasonably intuitive and well-established in the field of ML, this is far from being true in the context of unsupervised generative tasks. The generalization behavior that matters in unsupervised generative modeling is defined as the model's ability to generate new samples from an underlying, unknown probability distribution after training on a finite set of samples. In fact, developing new evaluation metrics is still an active area of research [25–32].

So far in the literature, assessing the quality of these quantum-circuit-based generative models has thus been almost entirely limited to how well they can memorize or reproduce a known target distribution. Despite its intrinsic value for benchmarking purposes, by sticking to a reproducibility/data-copying metric such as minimizing the empirical Kullback-Leibler (KL) divergence or negative log-likelihood (NLL) as the primary method of evaluating a generative model, we are not properly assessing the model's true ability to generalize, since the optimal solution would correspond to memorizing the training data. A recent work [33] specifically claims to focus on a

* alejandro@zapatacomputing.com

learning theory for quantum generative models from the perspective of generalization. Despite the authors' proof of theoretical generalization bounds for quantum generative models trained via Maximum Mean Discrepancy, their approach (based on Ref. [34]) does not explicitly take into account the crucial feature of novelty, which constitutes an essential ingredient to define generalization. The generalization error proposed in [33] quantifies the deviation of the empirically optimized probability distribution encoded by a model after training from the best probability distribution the model can represent, given its expressivity and a desired target distribution [1]. When this deviation is minimized, the learning performance is maximized. However, if a generative model is simply memorizing data, such deviation will be exactly zero, hence implying that this metric is not sensitive to the novelty of the generated samples. Since the comparison between training set and generated queries is out of the scope of Ref. [33], the model's true ability to generalize is not fully assessed.

In other words, most proposals fail to conduct a complete investigation as to how the model learns features of complex target distributions from a finite set of training data, and as a result ignore the role that important resource requirements (e.g., circuit depth and amount of training data) play in understanding the model's generalization capabilities in tasks of practical relevance. Recently, theoretical rigorous results were obtained for certain output distributions from local quantum circuits, proving some challenges in achieving a separation advantage with respect to classical models within this family [35]. While those results focus on worst-case bounds for the entire family of distributions, here we focus on the performance of QCBMs on specific realizations of application-relevant classical distributions.

Recently, a novel evaluation framework has opened the door for investigating generalization in classical and quantum generative models in greater depth [25]. In this work, we leverage such framework to evaluate, for the first time, the generalization capabilities of quantum-circuit-based models. We present the QCBM's generalization performance as an integral component of its evaluation as a generative model, and its ability to perform well with limited training data. In Sec. II, we review the key concepts, models, and metrics used in this work. In Sec. III, we present the main results from this study. First, we investigate the model's validity-based generalization performance, i.e., its ability to learn the valid features of non-exhaustive training datasets and generate novel valid samples post training. We evaluate the generalization throughout training, and observe improved performance at each iteration and overall when increasing the number of circuit layers. We then conduct an initial assessment of the QCBM's scarce-data regime by reducing the number of valid data shown to the model during training, and investigate the minimum data requirement needed to achieve high quality generalization performance. Lastly, we investigate the model's quality-based generalization performance, i.e., its ability to generalize not only to valid data, but also to valid data whose average associated cost is less than the cost of appropriately re-weighted training samples. We see that the QCBM is able to effectively learn the artificially reweighted training set and generate unseen samples with high quality. Finally, in Sec. IV, we conclude with some potential future research directions from this work.

[1] In practice, the target distribution is unknown, so including it in the model evaluation process is only feasible from a theoretical perspective.

II. Key Concept Review

Prior to presenting our generalization results for QCBMs, we provide a review of key concepts utilized throughout this work, including a basic introduction to unsupervised generative models, the QCBM model family, and the validity-based and quality-based frameworks required to assess generalization. Using these concepts, we introduce the overall algorithm and evaluation scheme utilized to quantify the generalization capabilities of QCBMs along with their resource requirements. A visualization of this explanation is provided in Figure 1.

A. Unsupervised Generative Models

Different to discriminative models in supervised learning, unsupervised generative models aim to learn an underlying, unknown, probability distribution $P(x)$ from a finite set of unlabelled training samples. These networks capture correlations within high-dimensional target distributions, often times having limited access to information, which makes generative modeling a much more difficult task compared to discriminative modeling [21, 36, 37]. Many kinds of generative models have been proposed in the literature, and often come with different architectures, training strategies, and limitations [6, 38]. A few notable model families include Generative Adversarial Networks (GANs) [39], Restricted Boltzmann Machines (RBMs) [40], Tensor Network Born Machines (TNBMs) [41] and Quantum Circuit Born Machines (QCBMs) [7]. While some of these models are able to perform data-driven tasks that require generative learning with distributions with continuous variables, we ultimately restrict our subsequent model definitions to networks that learn distributions with discrete variables, since it has been shown that discrete problems give rise to an unambiguous framework for assessing generalization [25, 32], and are more appropriate when working with quantum circuit models. Also, we highlight that the discrete nature of the dataset does not prevent the problem instance from being of practical relevance. For

example, these discrete tasks appear naturally in constrained combinatorial optimization problems.

Given that we have a discrete problem, we define the unsupervised generative task as one that attempts to learn an unknown target distribution $P(x)$ given only a set of training samples from such distribution. This set constitutes the training dataset $\mathcal{D}_{\text{Train}} = \{x_1, x_2, ..., x_D\}$, where each sample x_t is an N-dimensional binary vector such that $x_t \in \{0,1\}^N$ with $t = 1, 2, \ldots, D$. We denote the probability distribution defined by the training set as $P_{\text{train}}(x)$. Post training, the model is queried to generate data that composes the set $\mathcal{D}_{\text{Gen}} = \{x_1, x_2, ..., x_Q\}$, where each x_q is again an N-dimensional bitstring, with $q = 1, 2, \ldots, Q$. A good generative model should learn optimal parameters that make the model a faithful approximation of the original target distribution $P(x)$, implying that the model has the ability to generate data x_q from both inside and outside of the training set that are distributed according to $P(x)$.

We refer to the model's ability to generate data outside of the training set that are still distributed according to the data distribution $P(x)$ as *generalization*. Additionally, we note that the model may generate samples that are not in the training set, but also are not in support of the target distribution. The latter correspond to invalid samples, and we refer to them as noise. The model's ability to distinguish between noisy and valid samples is an important property that should be included in the performance assessment of any classical and quantum generative model, and it is one of the generalization metrics further discussed in Section II C.

B. Quantum Circuit Born Machines (QCBMs)

In the literature, QCBMs are one of the most popular quantum generative modeling families due to their highly expressive power [18] and the ability to perform direct sampling from the circuit as opposed to RBMs that require a costly Gibbs sampling. This model family takes advantage of the Born rule of quantum mechanics to sample from a quantum state $|\psi\rangle$ learned via training of a Parameterized Quantum Circuit (PQC) unitary $U(\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the vector of parameters for all of the single and entangling gates in the circuit. While alternative ansatz connectivities may be implemented, we showcase a line topology in Figure 1 as an appropriate choice for training circuits with very large depths. As the number of entangling gates scales linearly in the number of qubits for each layer, one can increase the number of layers with fewer number of parameters compared to other topologies. Note that alternative topologies, such as the all-to-all entangling connectivity available in ion-trap devices [7, 9, 16, 42], may be better for alternative tasks when the circuit depth can remain low. For a total number of layers L in the line circuit ansatz, each layer alternates between parameterized single-qubit rotation gate and multi-qubit entangling gate sequences until the final layer is reached. Each single-qubit gate sequence consists of an appropriate combination of Pauli X and Pauli Z rotations, $R_X(\theta)$ and $R_Z(\theta)$ respectively on each of the qubits, with $R_m(\theta) = \exp(\frac{-i\theta\sigma_m}{2})$. After each single-qubit gate layer, the entangling layer containing parameterized XX couplers between nearest neighbour qubits is executed, forming a line structure or an all-to-all connectivity as used in some of the cases here. Other topologies can also be explored (see e.g., Ref. [9]).

To minimize the number of variational parameters of the circuit and favor its trainability without sacrificing its expressive power, we can utilize the following strategy to carefully design the single-qubit layers [7], for an even L. In the first single-qubit gate sequence, we choose to decompose the arbitrary single-qubit transformation as $R_Z(\theta_1)\ R_X(\theta_2)\ R_Z(\theta_3)$. Since our initial state is $|00\cdots0>$, the first sequence of $R_Z(\theta_1)$ only adds a global phase to the quantum state, which is irrelevant since it will get washed out once we consider the Born's probabilities. Therefore, we can remove this first sequence of $R_Z(\theta_1)$ on each qubit without reducing the circuit's expressive power. For the next single-qubit sequences after the very first one, we propose a decomposition of the form $R_X(\theta_1)\ R_Z(\theta_2)\ R_X(\theta_3)$: it can be seen that the commutation of R_X with XX would lead to "collapsing" one sequence of R_X from the single-qubit sequence right before the entangling layer into the one right after it. Leveraging this commutation-and-collapse trick, all the single-qubit sequences after the first can also be reduced to $2N$ gates, except for the last one that has $3N$ gates as the final R_X doesn't have any following rotation to collapse into. For N qubits and an even L, this ansatz choice gives a total number of parameters $P = (3L/2+1)N - (L/2)$. When $L = 2$, the parameter count is simply given by $P = 3N - 1$ as there are only $2N$ single qubit gates in the first single-qubit sequence, as previously explained, followed by $N-1$ parametrized XX gates.

During each training iteration i, up to i_{Max}, the quantum circuit is simulated or run on quantum hardware with the current iteration parameter values $\boldsymbol{\theta}_i$, and is then queried to generate samples x_q according to the following model distribution [7]:

$$P_{\text{model}}(x_q|\theta_i) = |\langle x_q|\psi(\theta_i)\rangle|^2. \tag{1}$$

The generated samples are input into a classical cost function that measures the distance between the model output samples x_q and the training samples x_t taken from the underlying target distribution. Typical cost functions include the negative log likelihood (NLL) and the Maximum Mean Discrepancy (MMD) loss [8]. In this work, we utilize the NLL cost function at each iteration defined as:

$$\mathcal{C}(\boldsymbol{\theta}_i) = -\sum_{x_t \in \mathcal{D}_{\text{Train}}} P_{\text{train}}(x_t) \log\{\max[\epsilon, P_{\text{model}}(x_t|\theta_i)]\}, \tag{2}$$

where $\epsilon = 10^{-8}$ mitigates the singularity that occurs when $P_{\text{model}}(x_t|\theta_i) = 0$. A limitation of this cost func-

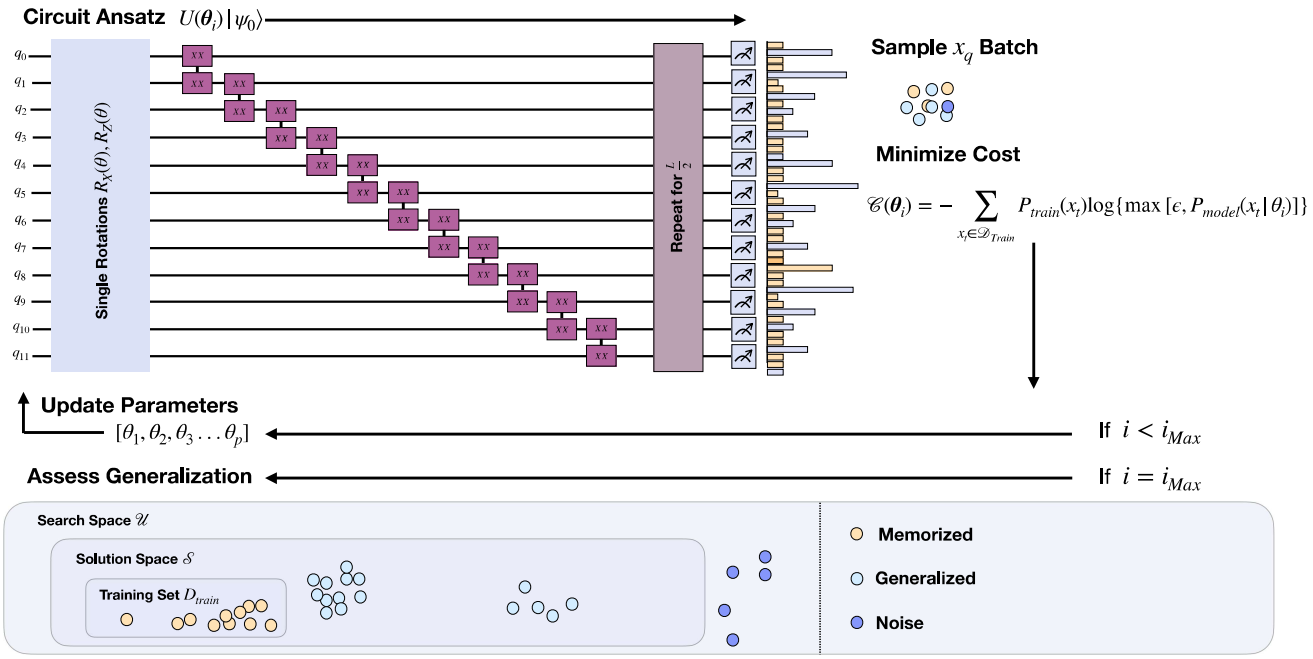

FIG. 1. **A visualization of the QCBM algorithm and generalization evaluation scheme.** Starting with randomly initialized parameters, the 12-qubit circuit ansatz with a line topology of parameterized gates for an even number of layers L is executed on a quantum simulator (note that both single-qubit and entangling gates are taken to be parametrized). Measuring in the computational basis provides samples distributed according to the probabilities encoded in the quantum state $|\psi\rangle$ that results from performing the unitary operation $U(\boldsymbol{\theta})$ on an initial quantum state $|\psi_0\rangle$. Iterative training is implemented up to a number of i_{Max} iterations in order to optimize the circuit parameters via minimization of the cost function $\mathcal{C}(\boldsymbol{\theta}_i)$. A post training sample-based evaluation scheme is conducted in order to assess the generalization capabilities of the QCBM. From a batch of sampled queries, each query x_q can be categorized as a memorized (yellow), generalized (light blue), or noise (dark blue) count. Given the number of each type of count, the Fidelity, Rate, and Coverage metrics (F, R, C) , or their equivalent normalized version, can be computed for the batch of samples, quantifying the generalization capabilities of the model.

tion is that it requires explicit access to $P_{\text{model}}(x_q)$, where we are only able to approximate this value with a finite number of queries Q taken from the trained model. Thus, this cost function is prone to the curse of dimensionality; it becomes challenging to approximate when scaling to larger data dimensions. Once computed, the cost is utilized in a gradient-based or gradient-free optimization scheme that updates the parameter values and feeds them back into the circuit for the next iteration. After a specified number of iterations i_{Max} (once a cost-threshold α is reached or some other convergence criterion is satisfied), the trained model can be queried and its output is used to evaluate its generalization capabilities.

There have been several proposals to mitigate the singularities and scalability issues arising from the use of the NLL (or, equivalently, the KL Divergence). The first proposal to mitigate this was to use the Maximum Mean Discrepancy (MMD) cost function [8]. Other papers have proposed other types of divergences, such as the Sinkhorn divergence and the Stein discrepancy [17], and other f-divergences [43]. Although changing the cost function might mostly help to reduce significantly the resources needed to estimate the cost function itself from the quantum device's samples, this might not suffice to address other trainability issues such as the presence of barren plateaus. Recently, we proposed a synergistic approach which leverages tensor networks' solutions to the problem, and use this approximation to initialize the training of the QCBMs ([44] and [45]). This strategy was shown not to exhibit such exponential vanishing gradients, and therefore it is expected to successfully scale to much larger problem instances. For the number of qubits studied here, we observed excellent performance via training under the NLL cost function.

C. Generalization Metrics

Evaluating unsupervised generative models has remained a challenging task in both the classical and quantum regime. Oftentimes, memorization is the center of this assessment. In the search for evaluating the model's true learning power over its ability to data-copy, one can aim to measure the model's generalization capability. However, this has proven to be a difficult task on its own [6]. Prominent metrics such as Precision p and

Recall r [29, 30, 46] aim to assess the quality of the generated data. However, these two numbers lack specific information regarding novelty since they include samples that are part of the training set. For example, one might obtain a perfect precision with samples coming exclusively from the training set. As mentioned in Section II A, generalization is the phenomenon that occurs when the model is able to produce samples outside of the training set $\mathcal{D}_{\text{train}}$ that are still distributed according to the target distribution $P(x)$. Evaluating generalization thus requires metrics that capture this capability, and the typical methods utilized in literature for measuring the quality of QCBMs, such as the KL Divergence [7, 9] fail in this regard. This metric will return a perfect score if $P_{\text{model}} = P_{\text{train}}$, and does not provide specific information regarding the model's ability to generalize to $P(x)$ from a finite set of training data. Furthermore, we note that one may split the training set into two, and utilize one as a test set for measuring the KL Divergence with respect to the model output. However, this measure only provides a single perspective as opposed to the detailed picture of the model's generalization capabilities provided by our metrics described next.

In this work, we leverage upon a recently proposed model-agnostic and sample-based framework [25] that places novelty at the forefront of the evaluation to assess the learning capabilities of QCBMs. The first step proposed in the framework evaluates the model's validity-based generalization, and it entails a few requirements. The first is that the target distribution $P(x)$ must describe samples that exist in a *valid* solution space $\mathcal{S}$ that is a subset of a given search space $\mathcal{U}$ of 2^N discrete states. The solution space is defined by some desired feature, where bitstrings that have this feature are considered valid, and are considered noise otherwise. However, only valid bitstrings that are unseen (i.e., not in the training set) are considered to be generalized samples. Generated bitstrings that can be found in the training set are considered to be memorized samples (see Figure .1). As a consequence, it is important that the training set is non-exhaustive so that there is room for generalization.

By drawing a specified number of queries Q, a model is assessed for its validity-based generalization capabilities by computing the *Fidelity (F)*, *Rate (R)*, and *Coverage (C)* metrics' values, defined as:

$$F = \frac{|\mathcal{G}_{\text{sol}}|}{|\mathcal{G}_{\text{new}}|}, \tag{3}$$

$$R = \frac{|\mathcal{G}_{\text{sol}}|}{Q}, \tag{4}$$

$$C = \frac{|g_{\text{sol}}|}{|\mathcal{S}| - D}. \tag{5}$$

In the formulas above, $\mathcal{G}_{\text{new}}$ is the multi-subset of unseen queries (noisy or valid), $\mathcal{G}_{\text{sol}}$ is the multi-subset of unseen and valid queries, and g_{sol} is the subset of unique unseen and valid queries. We note that F and R do not require a priori knowledge of $P(x)$ in order to be computed, while a limitation of C is that it requires knowledge of the size of the solution space $|\mathcal{S}|$.

Each of the F, R, C values provides unique information about the model's generalization capabilities, and therefore about its learning capabilities. Conceptually, the fidelity describes how well the model can generalize to valid samples rather than produce noise; the rate describes the frequency at which the model generates unseen valid samples; and the coverage describes the portion of the unseen valid space the model is able to learn. Altogether, these metrics provide a 3D picture of the model's capability to learn valid features in the dataset and produce novel valid samples.

In addition to the generalization metrics introduced above, we also define the typical, aforementioned, precision metric p and the pre-generalization exploration metric E. While these values do not quantify generalization performance directly, they do add information when compared alongside the generalization metrics. p is computed as follows:

$$p = \frac{|\mathcal{G}_{\text{train}}| + |\mathcal{G}_{\text{sol}}|}{Q}, \tag{6}$$

where $\mathcal{G}_{\text{train}}$ is the number of queries that were memorized from the training set. E is computed as follows:

$$E = \frac{|\mathcal{G}_{\text{new}}|}{Q}, \tag{7}$$

where the quantity shows the fraction of unseen samples that were queried from the model.

However, we note that when one is varying the size of the training set D, which can be controlled by increasing or decreasing the variable ϵ such that $D = \epsilon|\mathcal{S}|$, the metric computations may be individually affected as discussed below.

When computing the coverage metric across ϵ, we propose to use a normalized coverage $\tilde{C} = C/\overline{C}$, where $\overline{C}$ is taken to be the expected value of coverage, computed by:

$$\overline{C} = 1 - \left(1 - \frac{1}{|\mathcal{S}|(1-\epsilon)}\right)^{Q}. \tag{8}$$

This estimator, factoring in ϵ and the number of queries Q, indicates which coverage C one should expect when the generative model has perfectly learned the target distribution and generates samples accordingly [25]. When the number of queries is much smaller compared to the number of unseen solutions, i.e., $Q \ll |S|(1-\epsilon)$, we can approximate $\overline{C}$ as $\frac{Q}{|S|(1-\epsilon)}$. In this regime, the normalized coverage can be estimated as

$$\tilde{C} \approx \frac{|g_{\text{sol}}|}{Q}. \tag{9}$$

Interestingly, this expression does not need an explicit knowledge of $|S|$.

We additionally propose a normalized rate $\tilde{R} = R/\overline{R}$, where:

$$\overline{R} = 1 - \epsilon. \tag{10}$$

The latter is the rate one should expect if the target distribution is learned perfectly. More specifically, the probability that a generated query is valid and unseen (i.e, rate) corresponds to the portion of the valid space that is unseen for a given ϵ, i.e., $(\mathcal{S} - D)/\mathcal{S}$.

For the fidelity F, we note that proposing a normalized F is non-trivial as this metric does not have an ideal value that depends on ϵ. If the target distribution is learned exactly, then we should see $F = 1$ independently of ϵ. But there is a non-trivial dependence on ϵ for any other model which is not perfect. To illustrate this dependence, imagine one obtains the same model after training under two different values of ϵ. These two models will yield, for example, the same values for the precision, since p does not depend on the size of the training set but only on the probability of a query being in the valid sector. Since a larger ϵ implies that the portion of seen and valid data (the training set) is larger, this automatically implies that the sector of unseen and valid data is smaller and therefore the probability for a query to land there is smaller. Since the probability of landing in noise (unseen but invalid) remains the same, but the fraction of unseen and valid which appears in both the numerator and denominator of F changes, this yields different values for F for the same model. We leave the development of a normalized fidelity to future work, and highlight that this comment only becomes relevant when comparing across different ϵ values. When comparing models with the same ϵ, or assessing individual models, the metrics can be still used for accessing generalization.

Altogether, we are able to utilize these metrics in the first part of the framework to obtain a well-rounded picture of the QCBM's validity-based generalization capabilities.

The second step proposed in the framework, introduced in [25] as the quality-based generalization evaluation, requires each sample from the target distribution $P(x)$ to have an associated cost, so that all the training samples can be ranked by this value. The discrete training distribution is no longer expected to be uniform probabilities over all valid bitstrings, but rather each bitstring's sampling probability is re-weighted individually by its cost. With this method, one can utilize generative models to learn desired bitstrings for optimization-based tasks that correspond to real-world, relevant problems. A model is assessed for its quality-based generalization capabilities by looking at its *Utility* U, computed as

$$U = \langle c(\boldsymbol{x}) \rangle_{\boldsymbol{x} \in P_5}, \tag{11}$$

where $c(\boldsymbol{x})$ is the corresponding cost of a sample $\boldsymbol{x}$. P_5 corresponds to the set of samples with the lowest 5% costs of unseen and valid queries. One is therefore looking to determine if the utility of the model is lower when trained on a uniform distribution rather than on a reweighted distribution, and if the utility of the model is lower than that of each respective training distribution. The former question tells us if the model is able to learn the 'reweighting bias' introduced in the training set, and the latter tells us if the model is able to go beyond the samples in the training set and optimize further for generalized high quality samples. We note that other quality-based metric variations can be introduced as dependent on the objective of the task.

III. Results

We showcase our results on the validity-based and quality-based generalization capabilities of QCBMs. After providing the details of our simulation, we utilize the framework described in Section II C to monitor the $(F, \tilde{R}, \tilde{C})$ values of the QCBM throughout training with various circuit depths. Taking the metric values from the last training iteration, we then compare the models across ϵ in order to investigate the minimum data requirements to obtain high quality generalization performance. Lastly, we investigate the QCBM's ability to go beyond the validity generalization, and to generate high-quality (low cost) samples post learning from a re-weighted training set.

A. Simulation Details

For all numerical experiments in Section III B, we train a 12-qubit QCBM circuit with a line topology to learn a cardinality-constrained target distribution $P(x)$. This dataset has a solution space $\mathcal{S}$ defined by bitstrings that have a specific number k of 1s (e.g. '10001011' for $k = 4$). Thus the target distribution is uniform over the solution space:

$$P(x) = \frac{1}{|\mathcal{S}|} \quad \forall x \in \mathcal{S}. \tag{12}$$

The definition of such target distribution is only formal: in reality, it is not needed to have *a priori* knowledge of $|\mathcal{S}|$ for our metrics to be determined. The coverage metric (Eq. (5)) seems to require such knowledge. This requirement can be met when addressing benchmarking instances, such as the ones investigated in this work. However, in real-world cases, one is usually interested in comparing models (either different models or two different versions of the same model), hence one can simply compute their C ratios, which mitigates the fact that $|\mathcal{S}|$ is not known. Additionally, if one is aiming at estimating the coverage of a standalone model, one could simply use the asymptotic limit of the normalized coverage, given by Eq. (9), which quantifies the total number

of unique unseen and valid samples over the total number of queries and does not require knowledge of $|\mathcal{S}|$. This value would provide a sufficient estimation with regards to the number of unique values covered from the solution space that were not seen during training. The important notion in this framework is having a target distribution which is uniform and whose support is given by samples in a well-defined valid sector. Although in the case of the cardinality-constrained data set $|\mathcal{S}|$ can be estimated exactly, there are several real-world instances where it is easy to determine whether a sample belongs to the valid space, but it is intractable to determine *a priori* the size of the support. Examples of this class of problems can be found in Appendix 6 of Garey and Johnson's comprehensive book on NP-Complete problems [47]. For example, the *zero-one integer programming* problem described as MP1 in Appendix 6 is an NP-Complete problem where it is easy to check whether a given sample satisfies the constraints, but where it is intractable to find the whole set of bitstrings which satisfy the constraints.

We note that for this specific study, it is important to have a large $|\mathcal{S}|$ for assessing generalization, so that we can have appropriately sized training sets when spanning over ϵ. As such, we choose $k = 6$ for all runs, where $|\mathcal{S}| = \binom{12}{6} = 924$. The circuits are trained via a gradient-free Covariance Matrix Adaptation Evolution Strategy (CMA-ES) optimizer [48] with a NLL loss function on the Qulacs quantum simulator [49]. While we optimize with NLL, we display the cost values in the form of the KL Divergence, as it is easier to visualize the success of the training process. Indeed, we know that for $P_{\text{train}} = P_{\text{model}}$, we have KL $= 0$. Each circuit is initialized randomly, and the maximum number of training iterations is $i_{\text{Max}} = 10,000$. Lastly, $10,000$ samples are generated for the generalization evaluation procedure.

To assess the QCBM's quality-based generalization in Section III C, we introduce a re-weighted *Evens* dataset, where the solution space $\mathcal{S}$ is defined by bitstrings containing an even number of 1s (e.g. '01010011'), and a cost is assigned to each sample that quantifies the sample's degree of *separation* γ. We define *separation* as the largest bit-separation between 1s in the bitstring (e.g. γ('11010001') $= 4$). As we would like to optimize for the samples with the lowest cost, we focus on the negative separation $c = -\gamma$. We utilize this cost to reweight the training distribution via a softmax function on the training set bitstrings, namely:

$$P_b(x) = \frac{\exp\left(-\beta c(x)\right)}{\sum_{i=1}^{|\mathcal{D}_{\text{Train}}|} \exp\left(-\beta c(x_i)\right)}. \tag{13}$$

Following Ref. [50], we set $\beta = \beta_1 \equiv 1/T$ where T is the standard deviation of the costs that is interpreted as a 'temperature constant' [12]. Note that by adjusting β, one can tune the degree of reweighting introduced into the uniform distribution: for instance, choosing $\beta = \beta_2 \equiv 2/T$ increases the impact of the reweighting procedure. The artifact of reweighting the target distribution allows one to determine whether or not the model is able to learn this 'bias' induced by the bitstring costs in addition to the validity constraint.

The only additional change from the numerical experiments in Section III B is that we use an all-to-all $L = 2$ ansatz rather than a line topology. In our numerical experiments, we found that using an all-to-all topology provided better validity-based generalization performances compared to the line topology. This observation is possibly related to a trainability issue of the latter topology at the number of layers needed to describe the reweighted dataset, since the uniform even distribution can be constructed exactly using a two-layered QCBM with a line topology [51]. In general, for an arbitrary real-world dataset or more generic cases where one does not necessarily have an intuition of the type of circuit ansatz which might be suitable for the data, one needs to treat the exploration of the ansatz as an additional hyperparameter to be adjusted. Additionally, we use a fixed value of ϵ, with $\epsilon = 0.1$. As the *Evens* dataset is easier for the QCBM to learn, we use fewer layers and are thus able to avoid having too many parameters by implementing the all-to-all ansatz. We also remark that the choice of all-to-all compared to line topology on the *Evens* dataset improves trainability significantly even though the line topology is sufficient to represent this state.

Note that because the Evens solution space for the quality-based generalization assessment is much larger, 10% of the solution space accounts for a similar number of samples as used in the validity-based investigation. This might have helped in seeing a comparable performance with less percentage of data than the former dataset, although in reality the properties of each distribution to be learned can play a significant role and this data efficiency capability needs to be studied on a case-by-case basis. For the validity-based generalization demonstration in this work, we emphasize that the model is able to learn from a few training data, and as such, we span over ϵ intentionally. In a practical context, however, there is no need to know this percentage since all the metrics, quality and validity-based, can be computed without its knowledge.

B. Validity-Based Generalization

1. *Increasing Expressivity*

We show an example of the validity-based generalization for QCBM models trained with various circuit depths $L \in \{2, 4, 8, 16\}$. We only ran experiments with an even number of layers, where, as explained before, every layer of single qubit gates is followed by a layer of entangling gates. For the expressivity study presented here, we utilize only 30% ($\epsilon = 0.3$) of the solution space in all runs, where the training data is uniformly sampled from all valid data (i.e., the solution space). In Figure 3, we demonstrate that this is an adequate value for seeing good generalization performance. We run 5 independent

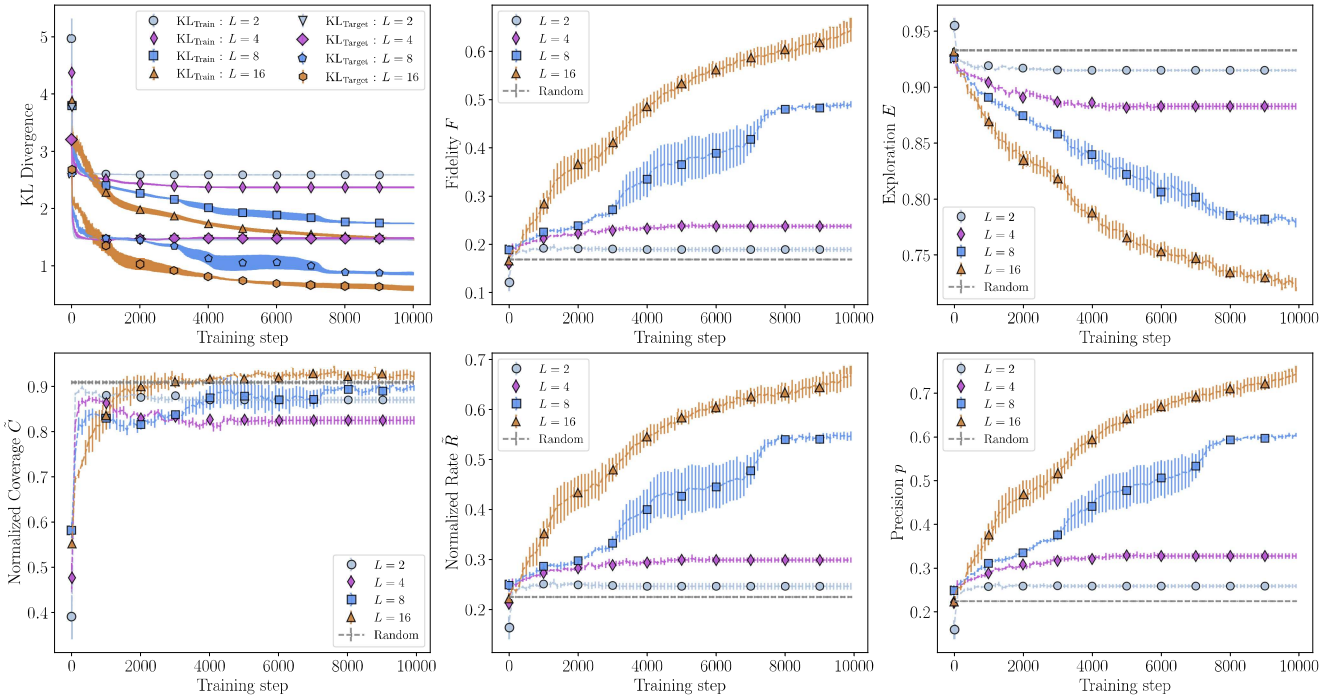

FIG. 2. **The QCBM's validity-based generalization performance throughout training across various circuit depths.** For each model with a different circuit depth $L \in \{2, 4, 8, 16\}$ and for $\epsilon = 0.3$, we show various metrics per training iteration including: the KL divergence of the model distribution relative to both the training and the target distribution (top left), the fidelity F (top middle), the exploration E (top right), the normalized coverage $\tilde{C}$ (bottom left), the normalized rate $\tilde{R}$ (bottom middle), and the precision p (bottom right). Note that these are average values over 5 independent trainings, where the error bars are computed by $\sigma/\sqrt{5}$. We see that the generalization performance increases throughout training, and that while for the majority of metrics all models are able to beat the random search baseline, the generalization performance is best for the deepest circuit model.

trainings of each model, and plot the result averages with error bars in Figure 2.

Across all results, we see that the deepest QCBM with $L = 16$ produces the best performance. Indeed, we see a direct correlation between increasing the number of layers, or the expressivity of the circuit, and a higher generalization performance quantified by the validity-based metrics. For shallower circuits with $L \in \{2, 4\}$, we barely see any increase in generalization performance across the average $(F, \tilde{R}, \tilde{C})$ values throughout training. Additionally, we see very little training in general as evident from looking at both the corresponding KL divergence, exploration, and precision trends. These models barely beat the random baseline [2]. After increasing the expressivity to $L \in \{8, 16\}$, we begin to see the model learn. The average $(F, \tilde{R}, \tilde{C})$ values steadily increase, where we see that the $L = 16$ model is able to reach the average values $(0.65, 0.67, 0.92)$, thus achieving high quality performance well above the average random baseline $(0.17, 0.23, 0.91)$. These metrics clearly show that the $L = 16$ QCBM is producing more unseen and valid samples with each new iteration step, and is learning the valid samples, distinguishing them from the noise. At the same time, the model's total exploration decreases as expected, as it begins to produce both valid samples from inside and outside of the training set. Table I displays individual values for the different metrics and Table II includes values for the random baseline.

While not always an attainable metric, due to the fact that the target distribution is typically unknown, one can compute the target KL Divergence $\mathrm{KL}_{\mathrm{Target}} = \mathrm{KL}(P(x)||P_{\mathrm{model}})$ and compare it to the usual KL Divergence, relative to the training set $\mathrm{KL}_{\mathrm{Train}} = \mathrm{KL}(P_{\mathrm{train}}||P_{\mathrm{model}})$, in order to see if the QCBM's output distribution is closer to the target than the training distribution. We see in Figure 2 that for $L = 16$, we have $\mathrm{KL}_{\mathrm{Target}}$ is smaller compared to $\mathrm{KL}_{\mathrm{Train}}$, indicating that the model is not overfitting to the training set. Note that we take the definition of *overfitting* to be *memorization* of the training distribution. We highlight that the values for $\mathrm{KL}_{\mathrm{Train}}$ are quite high compared to $\mathrm{KL}_{\mathrm{Target}}$, and according to this metric alone we could infer that the model does not have good generalization. However, displaying this metric alongside the $(F, \tilde{R}, \tilde{C})$ values, we are able

[2] The random baseline for each metric is computed from samples randomly drawn from the set of 2^N possible bitstrings [25].

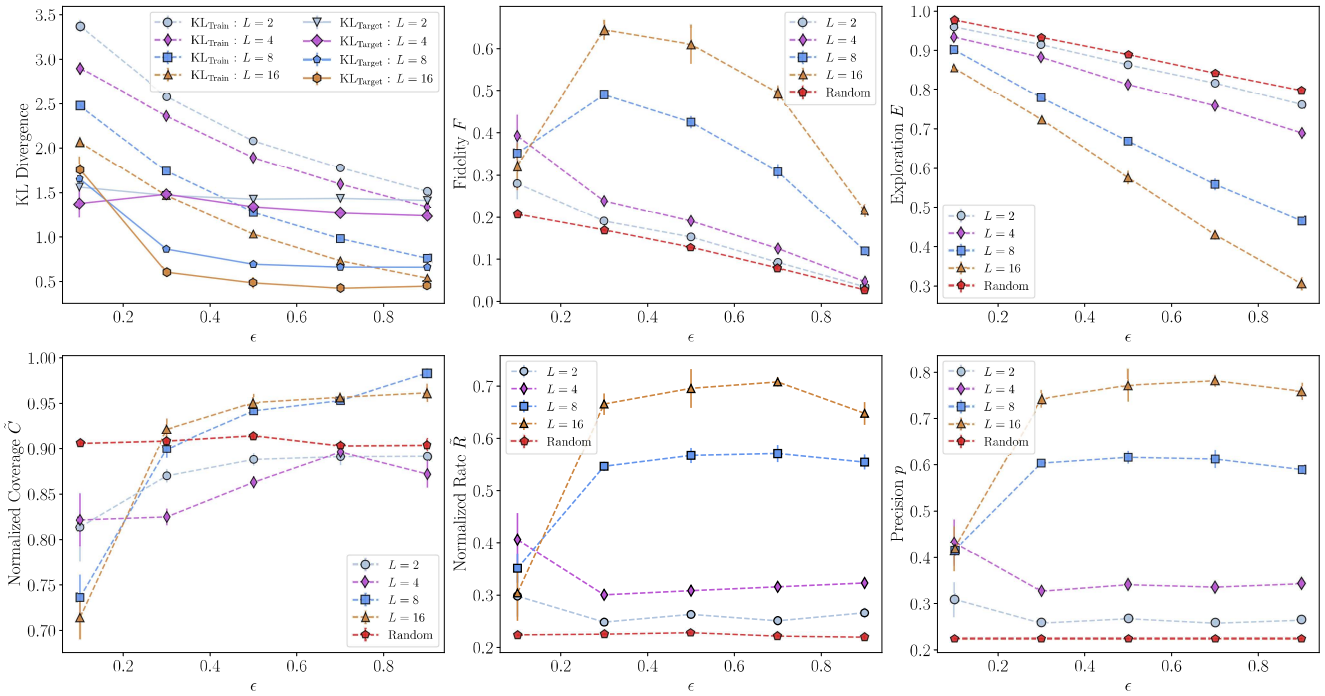

FIG. 3. **The QCBM's validity-based generalization performance on various training dataset sizes.** For each model with a different circuit depth $L \in \{2, 4, 8, 16\}$, we show various metrics across $\epsilon \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ values at the last iteration in training: the KL divergence of the model distribution relative to both the training and the target distribution (top left), the fidelity F (top middle), the exploration E (top right), the normalized coverage $\tilde{C}$ (bottom left), the normalized rate $\tilde{R}$ (bottom middle), and the precision p (bottom right). Note that these are average values over 5 independent trainings, where the error bars are computed as $\sigma/\sqrt{5}$. We see that the generalization performance increases for increasing circuit depth across all ϵ values. We see that for $L = 16$, with as few as 30% of the solution space used for training, the QCBM is able to exhibit great generalization performance on average: $(F, \tilde{R}, \tilde{C}) = (0.65, 0.67, 0.92)$, compared to that of the random baseline (in red): $(0.17, 0.23, 0.91)$.

to see that good generalization performance occurs even when $\mathrm{KL}_{\mathrm{Train}}$ is higher than zero. These numerics further support that achieving a $\mathrm{KL}_{\mathrm{Train}} = 0$ does not necessarily mean the model is producing good generalization performance: this metric alone should not be utilized to assess a generative model unless one is interested only in its data-copying capability.

We make a similar argument with the precision p and the exploration E. While simply providing p or E would not be enough to quantify the model's generalization performance, observing these values alongside the $(F, \tilde{R}, \tilde{C})$ metrics give us a broader picture. As the fidelity increases, despite the exploration decreasing, we can infer that this is simply because the model is disregarding noisy unseen samples. Additionally, as the rate increases alongside increasing precision, we can infer that a decent-sized portion of the valid samples being generated are unseen.

Interestingly, circuits with both two and four layers produce a similar $\mathrm{KL}_{\mathrm{Target}}$, and even though the 2-layers circuit displays a higher coverage, it contains more noise (less rate and fidelity than the 4-layers circuit). Although the models happen to have roughly the same $\mathrm{KL}_{\mathrm{Target}}$ value, our metrics allow one to disentangle the different generalization contributions and provide insights into the strengths and weaknesses of the models. This example also shows how judging performance just based on $\mathrm{KL}_{\mathrm{Train}}$ can be very misleading (with the 4-layers circuit largely outperforming the 2-layers one), while the generalization metrics are more faithful to the $\mathrm{KL}_{\mathrm{Target}}$.

2. *Reducing the Amount of Training Data*

We investigate the effect of the size of the training set on the model's ability to learn the valid correlations and generalize accordingly. We span over various percentage portions of the solution space $\epsilon \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ to use in the training set. Figure 3 showcases the average generalization performance at the last training iteration across 5 independent trainings for circuit depths $L \in \{2, 4, 8, 16\}$. We see that the trend of increasing circuit depth for enhanced performance extends to multiple ϵ values, thus suggesting that having enough expressivity is crucial for generalization to occur in this data set. Independent of the number of layers, it seems that

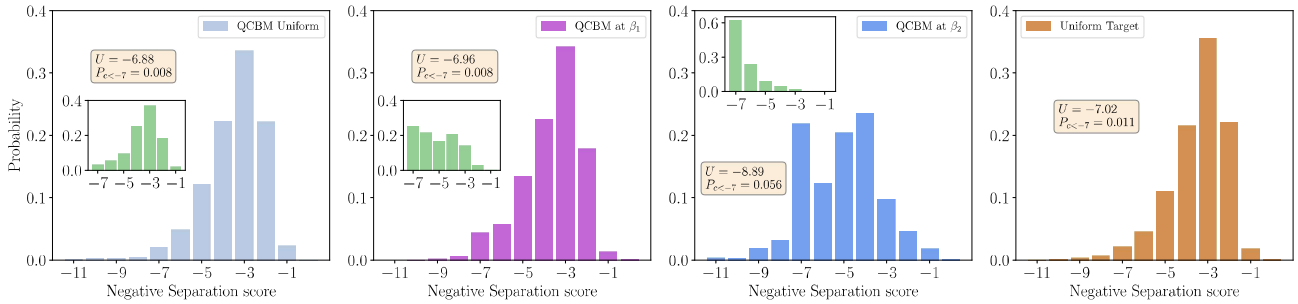

FIG. 4. **QCBM median output probability distributions when learning uniform vs. re-weighted datasets over 15 independent runs.** Each model is trained on a set of bitstrings in the *Evens* dataset, where each bitstring is mapped to its respective negative separation score, and to each training set a different degree of reweighting is applied. Note that we showcase the typical run across all models, as computed by the median $(F, \tilde{R}, \tilde{C})$ scores. We also note that the number of queries used to plot the histograms is 10000, except for the (green) histograms in the insets which correspond to the 204 samples used as the training sets ($\epsilon = 0.1$) for each degree of reweighting. Starting from the left, we show: the QCBM output when trained on a uniform training set; the QCBM output when trained on a re-weighted training set via a softmax function at inverse temperature $\beta_1 = 1/T$; the QCBM output when trained on a re-weighted training set via a softmax function at inverse temperature $\beta_2 = 2/T$; and 10,000 samples from the underlying uniform target distribution. While it is possible to observe values up to $c < -7$ when we sample 10,000 from the target distribution, across all training sets containing only 204 samples, the models do not "see" bitstrings with an associated $c < -7$. We highlight the quality-based generalization metric U for each distribution as well as the cumulative probability for generating samples where $P_{c<-7}$. With these two values, we are able to detect that the QCBM is able to achieve a lower U when we increase the level of reweighting in the training set, as well as a higher $P_{c<-7}$. Therefore, we see that the QCBM is able to effectively learn the 'reweighting bias' and generalize to data with lower negative separation costs than that of the training set.

when $\epsilon = 0.1$, the amount of training data is too low for the model to properly learn. When $\epsilon = 0.9$, we enter the regime where there is too much training data, which leaves very little room for generalization. Overall, we see that the model achieves good performance with access to 30% of the solution space during training. Table I displays individual values for the different metrics and Table II includes values for the random baseline.

For increasing ϵ, we see that the average exploration E decreases. Such behaviour is expected as we are artificially decreasing the number of unseen bitstrings that the model can generate by reducing the size of the unseen space. For the average $(F, \tilde{R}, \tilde{C})$ values, we note some interesting individual trends. F decreases after $\epsilon = 0.3$, whereas $\tilde{R}$ and $\tilde{C}$, and even the precision p, continue to increase slightly before decreasing at $\epsilon = 0.9$. We believe this discrepancy in the fidelity is related to the increase in the number of noisy unseen samples relative to the number of valid samples that the model could generate as we increase ϵ. This is similar to the effect we described in Section II A.

The models achieve better performance than the random search baseline, with the only exception of coverage when the circuit depths are too low ($L = 2, 4$). We see that $\mathrm{KL}_{\mathrm{Train}}$ tends to decrease with decreasing ϵ, and never encounters a turning point or a plateau. However, although $\mathrm{KL}_{\mathrm{Target}}$ presents the same decreasing trend until $\epsilon = 0.5$, it begins to plateau or change less dramatically until the largest value explored of $\epsilon = 0.9$. These two trends confirm that the QCBM model is indeed generalizing and not memorizing the data set, as memorization would imply an increased value for $\mathrm{KL}_{\mathrm{Target}}$. The fact that $\mathrm{KL}_{\mathrm{Target}} < \mathrm{KL}_{\mathrm{Train}}$ is the expected behaviour for a model which is generalizing since this means that the learned model distribution is closer to the target distribution than to the distribution from the training set. The decreasing in the value of $\mathrm{KL}_{\mathrm{Train}}$ can be easily explained from this behaviour since the training set is closer as well to the target distribution for larger values of ϵ. This in turns explain the closing of the gap between the values of $\mathrm{KL}_{\mathrm{Train}}$ and $\mathrm{KL}_{\mathrm{Target}}$.

In summary, we demonstrate that QCBMs are able to go beyond memorizing a training distribution, and learn valid features in an underlying target distribution. They exhibit strong validity-based generalization across $(F, \tilde{R}, \tilde{C})$ values. However, we note that the QCBM requires deeper circuits in order to obtain good generalization performance, and this may pose a challenge for obtaining good results on near-term hardware. We believe that understanding how the hardware's noise and connectivity may impact the generalization capabilities of these models is important future work.

C. Quality-Based Generalization

We assess the QCBM's quality-based generalization capability to see if the model is able to go beyond learning validity features in a dataset and learn an adequately reweighted version of it. As defined in Section

III A, each bitstring in the *Evens* dataset's solution space is given an associated score for its negative separation $c = -\gamma$, such that we can create a re-weighted training distribution according to the softmax function defined in Eq. (13). For 12 qubits, the negative separation cost is $c \in [-11, -1]$ with $c(\text{`}100000000001') = -11$ and $c(\text{`}111111111111') = -1$ as the limiting cases[3].

In Figure 4, we show the results of the typical behaviour, which we take to be the median value of the $(F, \tilde{R}, \tilde{C})$ results across 15 independent runs. Starting from the left, the first three distributions are the histograms of the valid samples generated after training the QCBM on the inset distributions (green). Across all $\epsilon = 0.1$ training sets (204 samples), the models do not see bitstrings with an associated $c < -7$. The distribution in the rightmost panel corresponds to 10,000 queries from the underlying uniform target distribution across all bitstrings in the *Evens* dataset mapped to their negative separation score. We share the average metric values across 15 independent runs in Table III and for the typical run in Table IV.

For all output distributions the precision and the validity-based generalization performance are very high, as supported by the results in Table IV. However, the distributions differ in their quality-based generalization capabilities, which we assess by computing the utility U metric described in Eq. (11) and the cumulative probability for $c < -7$, denoted as $P_{c<-7}$. We introduce the latter metric as a method to ensure that we are not simply reaching the lowest scores due to over-sampling, as upon sampling enough, it is reasonable to assume that the model will reach bitstrings with scores $c < -7$. If the model's $P_{c<-7}$ is higher than that of the uniform target, we see this as a fair way to assess that the model has learned the 'reweighting bias' introduced in the training set.

When trained on a uniform training set, the model generalizes to the underlying uniform target distribution and generates some missing low cost bitstrings such that it has a $U = -6.88$ and a $P_{c<-7} = 0.008$ for the median run. [4] When trained with a re-weighted softmax training set, we see that the utility decreases to $U = -6.96$, while $P_{c<-7}$ stays the same. However, when we turn up the degree of reweighting in the softmax function by halving the temperature constant T, we see that the model begins to learn beyond the validity-based features and on average generates more valid and unseen bistrings that have a lower associated cost than the minimum shown to the model through the training set. Remarkably, we see that the utility drops to $U = -8.89$ and the cumulative probability grows to $P_{c<-7} = 0.056$. These results support that the QCBM is able to effectively learn the 'reweighting bias' in the distribution, provided it is strong enough, and generalize to data outside of the training set with lower associated costs than that of the training set. This non-negligible quality-based generalization performance indicates that the QCBM may be useful for practical tasks in optimization, where one is interested in generating samples of minimal cost, by using them in the Generator-Enhanced Optimization (GEO) framework proposed in Ref. [50]. In future work, it would be beneficial to see if we can model higher-dimensional distributions with QCBMs for performing practical tasks with constrained optimization problems.

[3] When there is only one or no '1' in a bitstring, the cost is assigned to be zero. Those are only marginal cases, and they do not affect our analysis.

[4] We take the median run to be the median value out of all $F + \tilde{R} + \tilde{C}$ values, where we use the sum as a combined optimal score.

IV. Outlook

In the pursuit of obtaining a better fundamental understanding of quantum generative models such as QCBMs for real-world tasks, it is imperative to investigate not only their data-copying capability, but also their learning capabilities via the assessment of generalization performance. In this work, we conduct the first formal quantitative investigation of the QCBM's ability to learn feature patterns by demonstrating its generalization performance on uniform and re-weighted distributions. Our results show that the QCBM exhibits good validity-based generalization performance when learning a desired feature in a cardinality-constrained dataset. Overall, we see generalization performance tradeoffs when tuning various parameters. For instance, we show that one needs to increase the circuit depth, thus enhancing its expressivity, to obtain improved generalization performance. Additionally, we see that the model only requires access to 30% of the solution space to generalize well. We put forth these trends as an initial investigation into the QCBM's generalization capabilities. As we scale up the circuit width, we believe these trends will serve as a good starting point for testing.

We see that the QCBM's generalization performance is highly dependent on the number of valid data given to the model during training, and as such, it is of interest to dive deeper and conduct a more thorough future investigation into the QCBM's scarce data regime. In general, obtaining an improved understanding on the selection of challenging distributions that QCBMs can learn well is an important part of future work. As we scale to larger quantum circuit models and aim to use QCBMs in a more practical sense, it is paramount to obtain a better understanding of what practical tasks could be handled well by QCBMs. Our quality-based generalization results indicate that QCBMs are a good candidate for constrained optimization tasks, as the model can learn the correlations behind valid samples with associated low costs. In future work, we intend to scale up the size of the QCBM models and assess their capabilities on practical optimization tasks, such as portfolio optimization, where generalization capabilities have been shown to be a valu-

able asset [25, 50]. We foresee more elaborate training techniques for assessing the generalization performance of larger and deeper circuit sizes would be needed.

As this study is the first formal assessment of QCBM's generalization performance, we hope our investigations will open the door for tackling future questions regarding the interplay of trainability [52, 53], expressibility [54], and generalization [25, 33, 35] in these models. In our study, we fail to see signs of over-parameterization, and in accordance with classical ML, we feel that this behavior might come to light if we continued to increase the number of layers. It remains an open research question to understand over-parameterization in quantum models, especially in the context of quantum generative models. For example, it would be interesting to see if phenomena such as double-descent which is present in deep learning models [55] due to over-parametrization happens in QCBMs as well. Additionally, we believe it will be important to investigate the generalization performance of QCBMs on quantum hardware, and investigate how noise impacts the training resources required. In the pursuit of quantum advantage, it would be interesting to utilize the generalization framework in Ref. [25] to compare the QCBM's learning and generalization capabilities with those of other classical state-of-the-art generative models.

Most importantly, we aim to motivate the community to place importance on assessing the *generalization* over *memorization* capabilities when introducing a new quantum or classical generative model. We hope to see a shift in emphasis in the evaluation scheme towards prioritizing (or at least including) generalization performances.

Acknowledgments

We would like to thank Manuel Rudolph and John Realpe Gomez for helpful discussions and insights. We also acknowledge Dax Enshan Koh for reviewing our manuscript. K.G. would like to recognize the Army Research Office (ARO) (contract W911NF-20-1-0038) and UKRI (MR/S03238X/1) for providing funding through a QuaCGR PhD Fellowship. M.H. would like to acknowledge funding from MITACS through MITACS Accelerate. Lastly, K.G. would like to acknowledge the places that inspired this work: Oxford, UK; Santorini, Greece; Muscat, Oman; Lisbon, Portugal; Berlin, Germany and Boston, Greensboro, USA. Thank you to all of the local people who became a part of the journey. M.M. and K.G. would like to acknowledge places where collaboration for this work happened in person: Lake Como, Italy; Toronto, Canada; and Chicago, USA.

[1] H. Huang, P. S. Yu, and C. Wang, "An introduction to image synthesis with generative adversarial nets," (2018), arXiv:1803.04469 [cs.CV].

[2] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020) pp. 8110–8119.

[3] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, Journal of the American Medical Informatics Association **18** (2011).

[4] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, arXiv:1606.07792 (2016).

[5] A. Perdomo-Ortiz, M. Benedetti, J. Realpe-Gómez, and R. Biswas, Quantum Science and Technology **3**, 030502 (2018).

[6] J. Tian, X. Sun, Y. Du, S. Zhao, Q. Liu, K. Zhang, W. Yi, W. Huang, C. Wang, X. Wu, M.-H. Hsieh, T. Liu, W. Yang, and D. Tao, arXiv:2206.03066 (2022).

[7] M. Benedetti, D. Garcia-Pintos, Y. Nam, and A. Perdomo-Ortiz, npj Quantum Information **5** (2018).

[8] J.-G. Liu and L. Wang, PRA **98**, 062324 (2018).

[9] D. Zhu, N. M. Linke, M. Benedetti, K. A. Landsman, N. H. Nguyen, C. H. Alderete, A. Perdomo-Ortiz, N. Korda, A. Garfoot, C. Brecque, *et al.*, Science advances **5** (2019), 10.1126/sciadv.aaw9918.

[10] K. E. Hamilton, E. F. Dumitrescu, and R. C. Pooser, Physical Review A **99**, 062323 (2019).

[11] C. Zoufal, A. Lucchi, and S. Woerner, npj Quantum Information **5** (2019), 10.1038/s41534-019-0223-2.

[12] J. Alcazar, V. Leyton-Ortega, and A. Perdomo-Ortiz, Machine Learning: Science and Technology **1**, 035003 (2020).

[13] B. Coyle, M. Henderson, J. C. J. Le, N. Kumar, M. Paini, and E. Kashefi, Quantum Science and Technology **6**, 024013 (2021).

[14] M. Benedetti, B. Coyle, M. Fiorentini, M. Lubasch, and M. Rosenkranz, Phys. Rev. Applied **16**, 044057 (2021).

[15] M. S. Rudolph, N. B. Toussaint, A. Katabarwa, S. Johri, B. Peropadre, and A. Perdomo-Ortiz, Phys. Rev. X **12**, 031010 (2022).

[16] D. Zhu, W. Shen, A. Giani, S. R. Majumder, B. Neculaes, and S. Johri, "Copula-based risk aggregation with trapped ion quantum computers," (2022), arXiv:2206.11937.

[17] B. Coyle, D. Mills, V. Danos, and E. Kashefi, npj Quantum Information **6** (2020).

[18] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, Physical Review Research **2**, 033125 (2018).

[19] I. Glasser, R. Sweke, N. Pancotti, J. Eisert, and I. Cirac, Advances in neural information processing systems **32** (2019).

[20] R. Sweke, J.-P. Seifert, D. Hangleiter, and J. Eisert, Quantum **5**, 417 (2021).

[21] X. Gao, E. R. Anschuetz, S.-T. Wang, J. I. Cirac, and M. D. Lukin, Phys. Rev. X **12**, 021037 (2022).

[22] A. Abbas, D. Sutter, A. Figalli, and S. Woerner, "Effective dimension of machine learning models," (2021), arXiv:2112.04807 [cs.LG].

[23] M. C. Caro, H.-Y. Huang, M. Cerezo, K. Sharma, A. Sornborger, L. Cincio, and P. J. Coles, Nature Com-

munications **13** (2022).
[24] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," (2017), arXiv:1611.03530 [cs.LG].
[25] K. Gili, M. Mauri, and A. Perdomo-Ortiz, arXiv:2201.08770 (2022).
[26] A. M. Alaa, B. van Breugel, E. Saveliev, and M. van der Schaar, arXiv:2102.08921 (2021).
[27] A. Borji, Computer Vision and Image Understanding **215**, 103329 (2022).
[28] C. Meehan, K. Chaudhuri, and S. Dasgupta, arXiv:2004.05675 (2020).
[29] M. S. M. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly, in *NeurIPS* (2018).
[30] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila, arXiv:1904.06991 (2019).
[31] M. F. Naeem, S. J. Oh, Y. Uh, Y. Choi, and J. Yoo, in *International Conference on Machine Learning* (PMLR, 2020) pp. 7176–7185.
[32] S. Zhao, H. Ren, A. Yuan, J. Song, N. Goodman, and S. Ermon, Advances in Neural Information Processing Systems **31** (2018).
[33] Y. Du, Z. Tu, B. Wu, X. Yuan, and D. Tao, "Theory of quantum generative learning models with maximum mean discrepancy," (2022).
[34] G. K. Dziugaite, D. M. Roy, and Z. Ghahramani, arXiv:1505.03906 (2015).
[35] M. Hinsche, M. Ioannou, A. Nietner, J. Haferkamp, Y. Quek, D. Hangleiter, J.-P. Seifert, J. Eisert, and R. Sweke, "A single t-gate makes distribution learning hard," (2022).
[36] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, arXiv:2001.06937 (2020).
[37] L. Ruthotto and E. Haber, GAMM-Mitteilungen (2021), 10.1002/gamm.202100008.
[38] I. G. Y. Bengio and A. Courville, MIT Press (2016).
[39] I. Goodfellow, arXiv:1701.00160 (2016).
[40] G. Hinton, "A practical guide to training restricted boltzmann machines," (Springer Berlin Heidelberg, 2012) pp. 599–619.
[41] Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, PRX **8**, 031012 (2018).
[42] A. Rad, A. Seif, and N. M. Linke, "Surviving the barren plateau in variational quantum circuits with bayesian learning initialization," (2022).
[43] C. Leadbeater, L. Sharrock, B. Coyle, and M. Benedetti, Entropy **23** (2021), 10.3390/e23101281.
[44] M. S. Rudolph, J. Miller, J. Chen, A. Acharya, and A. Perdomo-Ortiz, arXiv:2208.13673 (2022).
[45] M. S. Rudolph, J. Chen, J. Miller, A. Acharya, and A. Perdomo-Ortiz, arXiv preprint arXiv:2209.00595 (2022).
[46] L. Simon, R. Webster, and J. Rabin, "Revisiting precision and recall definition for generative model evaluation," (2019), arXiv:1905.05441 [cs.LG].
[47] M. Garey and D. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness* (W.H. Freeman and Co., NY, 1979).
[48] N. Hansen, arXiv:1604.00772 (2016).
[49] Y. Suzuki, Y. Kawase, Y. Masumura, Y. Hiraga, M. Nakadai, J. Chen, K. M. Nakanishi, K. Mitarai, R. Imai, S. Tamiya, T. Yamamoto, T. Yan, T. Kawakubo, Y. O. Nakagawa, Y. Ibe, Y. Zhang, H. Yamashita, H. Yoshimura, A. Hayashi, and K. Fujii, Quantum **5**, 559 (2021).
[50] J. Alcazar, M. G. Vakili, C. B. Kalayci, and A. Perdomo-Ortiz, arXiv:2101.06250 (2021).
[51] M. Hibat-Allah, L. Serrano, A. Acharya, J. Chen, J. Realpe-Gomez, J. Carrasquilla, and A. Perdomo-Ortiz, In preparation.
[52] J. Mcclean, S. Boixo, V. Smelyanskiy, R. Babbush, and H. Neven, Nature Communications **9** (2018).
[53] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Nature communications **12**, 1 (2021).
[54] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, PRX Quantum **3**, 010313 (2022).
[55] P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever, Journal of Statistical Mechanics: Theory and Experiment **2021**, 124003 (2021).

V. Appendix

We present supplemental data for the results displayed in Section III for both validity and quality-based generalization performance.

A. Validity-Based Generalization

To complement our results in Section III B, we show the metrics' results for the last training iteration in Figure 2 and for the various ϵ values in Figure 3 in Table I. All metric values are averages over 5 independent trainings, where the error is given by the standard deviation over the square-root of the number of runs. Similarly, the average random baseline values with corresponding errors are shown in Table II.

L	ϵ	F	$\tilde{R}$	$\tilde{C}$	p	$\mathrm{KL}_{\mathrm{Target}}$	$\mathrm{KL}_{\mathrm{Train}}$
2	0.1	0.28 ± 0.04	0.29 ± 0.04	0.81 ± 0.04	0.31 ± 0.04	1.6 ± 0.2	3.37 ± 0.07
2	0.3	0.189 ± 0.005	0.248 ± 0.007	0.869 ± 0.007	0.259 ± 0.005	1.46 ± 0.02	2.578 ± 0.006
2	0.5	0.152 ± 0.003	0.263 ± 0.005	0.888 ± 0.007	0.268 ± 0.004	1.418 ± 0.008	2.079 ± 0.008
2	0.7	0.092 ± 0.002	0.251 ± 0.004	0.891 ± 0.009	0.259 ± 0.004	1.427 ± 0.006	1.771 ± 0.006
2	0.9	0.035 ± 0.001	0.27 ± 0.01	0.891 ± 0.03	0.265 ± 0.002	1.403 ± 0.006	1.506 ± 0.005
4	0.1	0.39 ± 0.05	0.41 ± 0.05	0.82 ± 0.03	0.43 ± 0.05	1.4 ± 0.2	2.89 ± 0.07
4	0.3	0.238 ± 0.005	0.300 ± 0.005	0.825 ± 0.009	0.328 ± 0.006	1.47 ± 0.02	2.36 ± 0.01
4	0.5	0.190 ± 0.004	0.309 ± 0.004	0.863 ± 0.007	0.341 ± 0.008	1.33 ± 0.02	1.587 ± 0.008
4	0.7	0.125 ± 0.001	0.316 ± 0.003	0.896 ± 0.009	0.336 ± 0.005	1.266 ± 0.005	1.587 ± 0.008
4	0.9	0.047 ± 0.002	0.32 ± 0.01	0.87 ± 0.01	0.344 ± 0.006	1.237 ± 0.009	1.329 ± 0.009
8	0.1	0.35 ± 0.03	0.35 ± 0.03	0.74 ± 0.03	0.41 ± 0.03	1.6 ± 0.1	2.48 ± 0.02
8	0.3	0.491 ± 0.007	0.546 ± 0.009	0.89 ± 0.01	0.603 ± 0.005	0.86 ± 0.03	1.73 ± 0.01
8	0.5	0.43 ± 0.01	0.57 ± 0.01	0.941 ± 0.004	0.62 ± 0.01	0.69 ± 0.02	1.27 ± 0.02
8	0.7	0.31 ± 0.02	0.57 ± 0.02	0.952 ± 0.004	0.61 ± 0.02	0.66 ± 0.02	0.98 ± 0.02
8	0.9	0.119 ± 0.005	0.55 ± 0.01	0.983 ± 0.005	0.589 ± 0.009	0.662 ± 0.009	0.761 ± 0.009
16	0.1	0.32 ± 0.06	0.30 ± 0.05	0.71 ± 0.02	0.42 ± 0.05	1.2 ± 0.1	2.07 ± 0.02
16	0.3	0.64 ± 0.02	0.67 ± 0.02	0.92 ± 0.01	0.74 ± 0.02	0.61 ± 0.05	1.46 ± 0.02
16	0.5	0.61 ± 0.05	0.69 ± 0.04	0.950 ± 0.009	0.78 ± 0.04	0.46 ± 0.06	1.03 ± 0.04
16	0.7	0.49 ± 0.02	0.708 ± 0.009	0.956 ± 0.005	0.78 ± 0.01	0.43 ± 0.02	0.73 ± 0.02
16	0.9	0.22 ± 0.02	0.65 ± 0.02	0.96 ± 0.01	0.76 ± 0.02	0.45 ± 0.03	0.54 ± 0.03

TABLE I. **Validity-based generalization values over various circuit depths and training set sizes.** The table lists the average $(F, \tilde{R}, \tilde{C})$ values, along with the precision p and the KL divergences relative to the target and the training set, across 5 independent trainings with the associated error for each model. Note that these are all values computed after sampling from the fully trained model ($i_{\mathrm{Max}} = 10\mathrm{k}$) when learning the cardinality-constrained target distribution.

ϵ	F	$\tilde{R}$	$\tilde{C}$	p
0.1	0.206 ± 0.002	0.224 ± 0.002	0.906 ± 0.003	0.225 ± 0.001
0.3	0.169 ± 0.001	0.225 ± 0.002	0.909 ± 0.003	0.225 ± 0.001
0.5	0.128 ± 0.001	0.228 ± 0.002	0.914 ± 0.003	0.225 ± 0.001
0.7	0.0790 ± 0.0007	0.222 ± 0.002	0.904 ± 0.005	0.225 ± 0.001
0.9	0.0276 ± 0.0005	0.219 ± 0.004	0.904 ± 0.007	0.225 ± 0.001

TABLE II. **Random search baselines for the validity-based generalization metrics across various ϵ values.** We show the average $(F, \tilde{R}, \tilde{C})$ values alongside the precision p, when no training is conducted and samples are taken at random. The results are averaged from 5 independent runs, and displayed with their corresponding standard errors.

B. Quality-Based Generalization

To complement our results in Section III C, we show the metrics' results for each output distribution in Figure 4, containing different degrees of reweighting. Note that in Table III all values are averages over 15 independent trainings, where their error is given by the standard deviation over the square-root of the number of runs. In Table IV, we show the median score of $F + \tilde{R} + \tilde{C}$ for each distribution, which directly corresponds to the model outputs displayed in Figure 4.

Distribution	F	$\tilde{R}$	$\tilde{C}$	p
Uniform	0.74 ± 0.07	0.69 ± 0.06	0.76 ± 0.02	0.77 ± 0.06
Reweighted T	0.74 ± 0.07	0.69 ± 0.06	0.74 ± 0.02	0.77 ± 0.06
Reweighted $T/2$	0.85 ± 0.07	0.75 ± 0.05	0.59 ± 0.03	0.87 ± 0.06

TABLE III. **Average validity-based generalization metrics when training on uniform vs reweighted *Evens* distributions.** We show the average $(F, \tilde{R}, \tilde{C})$ values, along with the precision p across 15 independent runs with the associated error for each model. Note that these are all values computed after sampling from the fully trained model ($i_{\text{Max}} = 10\text{k}$).

Distribution	F	$\tilde{R}$	$\tilde{C}$	p	U	$P_{c<-7}$
Uniform	0.99	0.89	0.81	0.99	-6.88	0.008
Reweighted T	1.0	0.91	0.79	1.0	-6.96	0.008
Reweighted $T/2$	1.0	0.82	0.79	1.0	-8.89	0.056

TABLE IV. **Median validity-based generalization and quality-based metrics when training on uniform vs reweighted *Evens* distributions.** We show the median $(F, \tilde{R}, \tilde{C})$ values, along with the precision p for 15 independent runs. Additionally, we show U and $P_{c<-7}$ values that correspond to the median $(F, \tilde{R}, \tilde{C})$ score. Note that these are all values computed after sampling from the fully trained model ($i_{\text{Max}} = 10\text{k}$).