The Born supremacy: quantum advantage and training of an Ising Born machine

Brian Coyle, Daniel Mills, Vincent Danos, Elham Kashefi

To cite this version:

Brian Coyle, Daniel Mills, Vincent Danos, Elham Kashefi. The Born supremacy: quantum advantage and training of an Ising Born machine. npj Quantum Information, 2020, 6 (1), pp.60. 10.1038/s41534-020-00288-9 . hal-03096252

HAL Id: hal-03096252

https://hal.science/hal-03096252

Submitted on 4 Jan 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

npj | Quantum Information www.nature.com/npjqi

ARTICLE OPEN

Check for updates

The Born supremacy: quantum advantage and training of an Ising Born machine

Brian Coyle[1] ✉, Daniel Mills[1], Vincent Danos[1,2] and Elham Kashefi[1,3]

The search for an application of near-term quantum devices is widespread. Quantum machine learning is touted as a potential utilisation of such devices, particularly those out of reach of the simulation capabilities of classical computers. In this work, we study such an application in generative modelling, focussing on a class of quantum circuits known as Born machines. Specifically, we define a subset of this class based on Ising Hamiltonians and show that the circuits encountered during gradient-based training cannot be efficiently sampled from classically up to multiplicative error in the worst case. Our gradient-based training methods use cost functions known as the Sinkhorn divergence and the Stein discrepancy, which have not previously been used in the gradient-based training of quantum circuits, and we also introduce quantum kernels to generative modelling. We show that these methods outperform the previous standard method, which used maximum mean discrepancy (MMD) as a cost function, and achieve this with minimal overhead. Finally, we discuss the ability of the model to learn hard distributions and provide formal definitions for 'quantum learning supremacy'. We also exemplify the work of this paper by using generative modelling to perform quantum circuit compilation.

npj Quantum Information (2020)6:60 ; https://doi.org/10.1038/s41534-020-00288-9

INTRODUCTION

As quantum devices with ~80−200 qubits, but without fault tolerance, begin to be built, we near the dawn of the noisy intermediate scale quantum (NISQ)[1] technology era. Because of the low number of qubits, the limited connectivity between them, and the low circuit depth permitted by low decoherence times, these devices cannot perform many of the most famous algorithms thought to demonstrate exponential speedups over classical algorithms[2,3].

In spite of this, NISQ devices could provide efficient solutions to other problems that cannot be solved in polynomial time by classical means. Showing this to be true is referred to as a demonstration of quantum computational supremacy[4–9], with the first such experimental realisation occurring recently[10].

Proposals for demonstrations of quantum computational supremacy on NISQ technology typically involve sampling from the output distribution of random quantum circuits. While a realisation of such an advantage is of great theoretical importance, generating random samples is not obviously independently interesting. We incorporate this sampling into a useful application, keeping the provable quantum advantage, but in a context with more practical applicability.

Specifically, we explore generative modelling in quantum machine learning (QML), which is the task of generalising from a finite set of samples, $\{\mathbf{y}\}^M$, drawn from a data set. By learning the underlying probability distribution from which these samples are drawn, $\pi(\mathbf{y})$, a model should be able to generate new samples from the said distribution.

Generative models range from simple naive Bayes[11] models to complicated neural networks, like generative adversarial networks (GANs)[12]. The intrinsic randomness inherent in quantum mechanics allows for the definition of a new class of generative models that are without a classical analogue. Known as Born machines[13–15], they have the ability to produce statistics according to Born's measurement rule. Specifically, for a state $|\psi\rangle$, a measurement produces a sample $\mathbf{x} \sim p(\mathbf{x}) = |\langle \mathbf{x}|\psi\rangle|^2$. There are several variants, including Bayesian approaches[16], adversarial training methods[17], and adaptations to continuous distributions[18].

Quantum circuit Born machines (QCBM) are a subclass of parameterised quantum circuits (PQCs) and are widely applicable (see ref. [19] for a review). PQCs consist of a quantum circuit which carries parameters that are updated during a training process (typically a classical optimisation routine). The circuit is kept as shallow as possible so as to be suitable for NISQ devices.

We ask in this work whether it is possible to have a machine learning application for a PQC, which comes with a provable superior performance over all classical alternatives on near term devices? Such provable guarantees are even more relevant given recent work in QML algorithm 'dequantisations'[20–24].

We take the first steps in answering this question in several ways. We define a subclass of QCBM that we call Ising Born machines (QCIBM). We improve the training of the model over previous methods, which use the maximum mean discrepancy[14] (MMD) with a classical kernel, by introducing quantum kernels into the MMD, as well as by using entirely new cost functions: the Stein discrepancy (SD) and the Sinkhorn divergence (SHD). To do so, we derive their corresponding gradients in the quantum setting.

We show that these novel methods outperform the MMD with classical kernel by achieving a closer fit to the data as measured by the total variation (TV) distance. We derive forms of the SHD, which can either be efficient to compute or result in an upper bound on TV. We observe numerically that the SD provides an upper bound to TV. Next, we show that sampling from this model cannot be simulated efficiently by any classical randomised algorithm, up to multiplicative error in the worst case, subject to common assumptions in complexity theory (namely the non-collapse

---

[1] School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh, UK. [2] CNRS, École Normale Supérieure, 75005 Paris, France. [3] CNRS, LIP6, Sorbonne Université, 4 Place Jussieu, 75005 Paris, France. ✉email: brian.coyle@ed.ac.uk

of the polynomial hierarchy). Furthermore, this holds for many circuit families encountered during training.

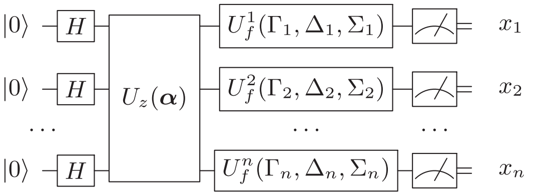
We define a framework in which a provable advantage could be demonstrated, which we refer to as quantum learning supremacy (QLS), and based on distribution learning theory[25]. Based on our classical sampling hardness results, we conjecture that the QCIBM may be a good candidate for a quantum model which could demonstrate this notion of learning supremacy; however, we leave the further investigation of QLS and its potential to be achieved by such models to future work. Finally, we provide a novel utilisation of such generative models in quantum circuit compilation.

RESULTS

The main results of this work are new efficient gradient-based training methods and results on the hardness of simulating the model we introduce using classical computers. First, we define the model used and discuss its connection to previously studied quantum circuit families. We then discuss the efficient training of the model, first recalling a previously known gradient-based training method, which uses the MMD cost function, and then moving onto our new training methods, which use the SD and the SHD. We then discuss the SHD complexity in detail, and further argue, using its connection to the TV distance, why it should be used. We then prove the hardness results mentioned above, namely that many circuits encountered during gradient-based training are hard to classically simulate, before finally discussing the potential use of quantum generative models in learning distributions that are intractable to classical models. In addition, we provide a framework to study these advantages.

Ising Born machine

Here we define the model we use for distribution learning. A generic quantum circuit Born machine consists of a parameterised quantum circuit, which produces samples by measuring the resulting quantum state, and a classical optimisation loop used to learn a data distribution. The circuits we study have the following structure:


$|0\rangle$
H
$|0\rangle$
H
$|0\rangle$
H
$U_z(\boldsymbol{\alpha})$
$U_f^1(\Gamma_1, \Delta_1, \Sigma_1)$
$U_f^2(\Gamma_2, \Delta_2, \Sigma_2)$
$U_f^n(\Gamma_n, \Delta_n, \Sigma_n)$
x_1
x_2
x_n
...

(1)

where $x_i \in \{0, 1\}$; the unitaries are defined by Eqs. (2) and (3); S_j indicates the subset of qubits on which each operator, j, is applied; and a boldface parameter indicates a set of parameters, $\boldsymbol{\alpha} = \{\alpha_j\}$.

$$U_z(\boldsymbol{\alpha}) := \prod_j U_z(\alpha_j, S_j) = \prod_j \exp\left(\mathrm{i}\alpha_j \bigotimes_{k \in S_j} Z_k\right) \tag{2}$$

$$U_f(\boldsymbol{\Gamma}, \boldsymbol{\Delta}, \boldsymbol{\Sigma}) := \exp\left(\mathrm{i}\sum_{k=1}^{n} \Gamma_k X_k + \Delta_k Y_k + \Sigma_k Z_k\right) \tag{3}$$

The operators, X_k, Y_k, and Z_k, are the standard Pauli operators acting on qubit k. Restricting to the case $|S_j| \leq 2$ (since only single and two-qubit gates are required for universal quantum computation), the term in the exponential of Eq. (2) becomes exactly an Ising Hamiltonian:

$$\mathcal{H} := \mathrm{i}\sum_{i<j} J_{ij} Z_i Z_j + \mathrm{i}\sum_{k=1}^{n} b_k Z_k, \tag{4}$$

where we are dividing the diagonal unitary parameters, $\boldsymbol{\alpha} = \{J_{ij}, b_k\}$, into local terms that act only on qubit k, $\{b_k\}$, and coupling terms between two qubits i and j, $\{J_{ij}\}$. We call the model a QCIBM.

A measurement on all qubits in the computational basis results in sample vectors, $\mathbf{x} \in \mathcal{X}^n$, where $\mathcal{X} = \{0, 1\}$. These samples are drawn from the distribution, $p_{\boldsymbol{\theta}}(\mathbf{x})$, parameterised by the set of angles, $\boldsymbol{\theta} = \{\boldsymbol{\alpha}, \boldsymbol{\Gamma}, \boldsymbol{\Delta}, \boldsymbol{\Sigma}\}$:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) := \left|\langle \mathbf{x}|U_f(\boldsymbol{\Gamma}, \boldsymbol{\Delta}, \boldsymbol{\Sigma})U_z(\boldsymbol{\alpha})|+\rangle^{\otimes n}\right|^2. \tag{5}$$

We denote the above model and parameters by QCIBM($\boldsymbol{\theta}$) := QCIBM($\boldsymbol{\alpha}$, $\boldsymbol{\Gamma}$, $\boldsymbol{\Delta}$, $\boldsymbol{\Sigma}$). We choose this structure in order to easily recover two well-known circuit classes, namely instantaneous quantum polynomial time[26] (IQP) circuits, and the shallowest depth ($p = 1$) version of the quantum approximate optimisation algorithm[27] (QAOA).

IQP circuits are named to reflect the commuting nature of elements in the produce defining the unitary U_z, while QAOA[27] was originally developed as an approximate version of the quantum adiabatic algorithm[28]. Both of these classes of circuits are known to be routes to demonstrate quantum supremacy[4,6,8,29], and we extend this property here by using the results of ref. [30]. These classes can be recovered by setting the parameters of a QCIBM as follows:

$$\mathrm{IQP}(\{J_{ij}, b_k\}) = \mathrm{QCIBM}\left(\{J_{ij}, b_k\}, \boldsymbol{\Gamma} = \left\{\frac{\pi}{2\sqrt{2}}\right\}, \mathbf{0}, \boldsymbol{\Sigma} = \left\{\frac{\pi}{2\sqrt{2}}\right\}\right), \tag{6}$$

$$\mathrm{QAOA}_{p=1}(\{J_{ij}, b_k\}, \boldsymbol{\Gamma}) = \mathrm{QCIBM}(\{J_{ij}, b_k\}, \boldsymbol{\Gamma} = -\boldsymbol{\Gamma}, \mathbf{0}, \mathbf{0}). \tag{7}$$

We denote, for example $\left\{\frac{\pi}{2\sqrt{2}}\right\}$, to be all parameters of the n single qubit gates set to the same value, $\pi/2\sqrt{2}$. We choose the final gate before the computational basis measurement to be in the form of Eq. (3), rather than the more common Euler decomposition of a single qubit gate decomposition found in the literature[14,16]. This is chosen to make the classical simulation hardness results more apparent in our proofs.

To recover IQP circuits, we simply need to generate the final layer of Hadamard gates (up to a global phase) and do so by setting U_f in Eq. (3) as follows:

$$U_f^{\mathrm{IQP}}\left(\left\{\frac{\pi}{2\sqrt{2}}\right\}, \mathbf{0}, \left\{\frac{\pi}{2\sqrt{2}}\right\}\right) = \bigotimes_{k=1}^{n} \mathrm{e}^{\frac{\mathrm{i}\pi}{2\sqrt{2}}(X_k + Z_k)} = \mathrm{i}H^{\otimes n}. \tag{8}$$

To recreate depth 1 QAOA circuits, we need to set the Pauli Z and Y parameters, $\boldsymbol{\Delta}, \boldsymbol{\Sigma} = \mathbf{0}$, since the final gates should be a product of Pauli-X rotations with parameters, $-\boldsymbol{\Gamma}$.

Training the Ising Born machine

Here we introduce the alternative training methods that we use for our purposes and that would be applicable to any generative model. The training procedure is a hybrid of classical and quantum computation, with the only quantum component being the model itself. The remainder of the computation is classical, bringing our scheme into the realm of what is possible for NISQ devices. The procedure can be seen in Fig. 1.

The optimisation procedures we implement are stochastic gradient descents. Parameters, θ_k, are updated at each epoch of training, d, according to the rule $\theta_k^{d+1} \leftarrow \theta_k^d - \eta\,\partial_{\theta_k}\mathcal{L}_B$. The parameter η is the learning rate and controls the speed of the descent. The initial proposals to train QCBMs were gradient-free[15,31], but gradient-based methods have also been proposed[14,16,32]. In this work, we advocate for increasing the classical computational power required in training to achieve better performance, rather than increasing the quantum resources, for example by adding extra ancillae[16] or adding costly and potentially unstable (quantum) adversaries[17,33,34].

For gradient-based methods, a cost function or metric is required, $\mathcal{L}_B(p_{\boldsymbol{\theta}}(\mathbf{x}), \pi(\mathbf{y}))$ to compare the Born Machine distribution, $p_{\boldsymbol{\theta}}(\mathbf{x})$, and the data distribution, $\pi(\mathbf{y})$. Good cost functions will

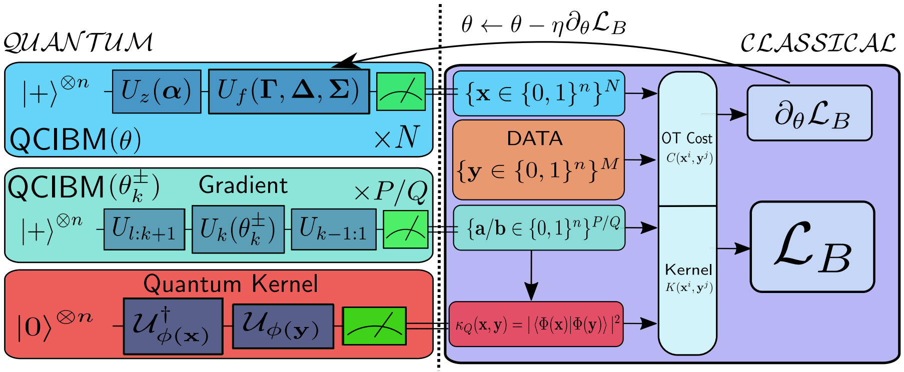

QUANTUM
CLASSICAL
$\theta \leftarrow \theta - \eta\partial_\theta\mathcal{L}_B$
$|+\rangle^{\otimes n}$
$U_z(\boldsymbol{\alpha})$
$U_f(\boldsymbol{\Gamma},\boldsymbol{\Delta},\boldsymbol{\Sigma})$
QCIBM(θ)
$\times N$
QCIBM$(\theta_k^\pm)$
Gradient
$\times P/Q$
$|+\rangle^{\otimes n}$
$U_{l:k+1}$
$U_k(\theta_k^\pm)$
$U_{k-1:1}$
Quantum Kernel
$|0\rangle^{\otimes n}$
$\mathcal{U}^\dagger_{\phi(\mathbf{x})}$
$\mathcal{U}_{\phi(\mathbf{y})}$
$\{\mathbf{x}\in\{0,1\}^n\}^N$
DATA
$\{\mathbf{y}\in\{0,1\}^n\}^M$
$\{\mathbf{a}/\mathbf{b}\in\{0,1\}^n\}^{P/Q}$
$\kappa_Q(\mathbf{x},\mathbf{y}) = |\langle\Phi(\mathbf{x})|\Phi(\mathbf{y})\rangle|^2$
OT Cost
$C(\mathbf{x}^i,\mathbf{y}^j)$
Kernel
$K(\mathbf{x}^i,\mathbf{y}^j)$
$\partial_\theta\mathcal{L}_B$
$\mathcal{L}_B$

Fig. 1 The hybrid training procedures we propose in this work. We have a quantum generator along with auxiliary circuits used to compute the gradient of the various cost functions with respect to the parameters. The training procedure proceeds as follows. First, the QCIBM is sampled from N times via measurements. These samples, along with M data samples $\mathbf{y} \sim \pi(\mathbf{y})$, are used to evaluate a cost function, $\mathcal{L}_B$, where $B \in \{\text{MMD, SD, SHD}\}$ is one of the efficiently computable cost functions. For each updated parameter, θ_k, two parameter-shifted circuits are also ran to generate samples, $\mathbf{a}, \mathbf{b} \sim p_{\theta\pm}$, which are used to compute the corresponding gradients, $\partial_\theta\mathcal{L}_B$. For all costs functions and gradients, either a kernel (if a quantum kernel is used, the circuit in this figure must be run) is computed for each pair of samples (as is the case for MMD and SD) or an optimal transport cost function is evaluated (as is the case for SHD).

have several characteristics. They should be efficient to compute, measured both by sample and computational complexity. They should also be powerful in the sense that they are sensitive to differences between the two distributions. In this work, we will assess sensitivity by using the TV metric as a benchmark:

$$\mathrm{TV}(p_{\boldsymbol{\theta}}, \pi) := \frac{1}{2}\sum_{\mathbf{x}} |p_{\boldsymbol{\theta}}(\mathbf{x}) - \pi(\mathbf{x})|. \tag{9}$$

As discussed later, this is a particularly strong metric. The cost functions we use are typically easier to deal with than TV, and we will remark on their relationship to TV.

One cost function commonly used to train generative models is the Kullback–Leibler (KL) divergence. The KL divergence is also relatively strong, in the sense that it upper bounds TV through Pinsker's inequality:

$$\mathrm{TV}(p_{\boldsymbol{\theta}}, \pi) \leq \sqrt{\frac{1}{2}D_{\mathrm{KL}}(p_{\boldsymbol{\theta}}||\pi)}, \tag{10}$$

where $D_{\mathrm{KL}}(p_{\boldsymbol{\theta}}||\pi)$ is the KL divergence of π from $p_{\boldsymbol{\theta}}$. Unfortunately, it is difficult to compute, having a high sample complexity, so neither its gradient nor the KL divergence itself can be evaluated efficiently when training parameterised circuits[14].

The first efficient gradient method to train Born machines was proposed by ref. [14]. There the MMD is used to define the cost function. We extend this methodology in two ways. The first is an alteration to the MMD itself, and the second is by introducing new cost functions. From the MMD, the following cost function[35,36] can be defined:

$$\mathcal{L}_{MMD} := \underset{\substack{\mathrm{x}\sim p_\theta\\ \mathrm{y}\sim p_\theta}}{\mathbb{E}}(\kappa(\mathrm{x},\mathrm{y})) + \underset{\substack{\mathrm{x}\sim \pi\\ \mathrm{y}\sim \pi}}{\mathbb{E}}(\kappa(\mathrm{x},\mathrm{y})) - 2\underset{\substack{\mathrm{x}\sim p_\theta\\ \mathrm{y}\sim \pi}}{\mathbb{E}}(\kappa(\mathrm{x},\mathrm{y})). \tag{11}$$

The MMD has some very favourable properties; it is a metric on the space of probability distributions, and it is relatively easy to compute (due to low sample complexity). The function, κ in Eq. (11) is a kernel function, a measure of similarity between points in the sample space $\mathbf{x} \in \mathcal{X}^n$. A popular choice for this function is the Gaussian mixture kernel[14]:

$$\kappa_{\mathrm{G}}(\mathbf{x}, \mathbf{y}) := \frac{1}{c}\sum_{i=1}^{c} \exp\left(-\frac{||\mathbf{x}-\mathbf{y}||_2^2}{2\sigma_i}\right). \tag{12}$$

The parameters, σ_i, are bandwidths that determine the scale at which the samples are compared, and $||\cdot||_2$ is the ℓ_2 norm.

Recent works[37,38] on the near term advantage of using quantum computers in QML have explored quantum kernels, which can be evaluated on a quantum computer. To gain such an advantage, these kernels should be difficult to compute on a classical device. In particular, we will adopt the following kernel[37] in which the samples are encoded in a quantum state, $|\phi(\mathbf{x})\rangle$, via a feature map, $\phi{:}\mathbf{x} \to |\phi(\mathbf{x})\rangle$. The kernel is the inner product between vectors:

$$\kappa_{\mathrm{Q}}(\mathbf{x}, \mathbf{y}) := |\langle\phi(\mathbf{x})|\phi(\mathbf{y})\rangle|^2. \tag{13}$$

The inner product in Eq. (13) is evaluated on a quantum computer and is conjectured to be hard to compute on a classical one[37], given only a classical description of the quantum states. The state $|\phi(\mathbf{x})\rangle$ is produced by acting an encoding unitary on an initial state, $|\phi(\mathbf{x})\rangle = \mathcal{U}_{\phi(\mathbf{x})}|0\rangle^{\otimes n}$. Explicitly, the kernel is then given by:

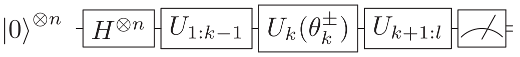
$$\kappa_{\mathrm{Q}}(\mathbf{x}, \mathbf{y}) = \left|\langle 0^{\otimes n}|\mathcal{U}^\dagger_{\phi(\mathbf{x})}\mathcal{U}_{\phi(\mathbf{y})}|0^{\otimes n}\rangle\right|^2, \tag{14}$$

which can be calculated by measuring, in the computational basis, the state which results from running the circuit given by $\mathcal{U}_{\phi(\mathbf{y})}$, followed by that of $\mathcal{U}^\dagger_{\phi(\mathbf{x})}$. This is seen in Fig. 1. The kernel, Eq. (14), is the observed probability of measuring the all-zero outcome, 0^n. If this outcome is not observed after polynomially many measurements, the value of the kernel for this particular pair of samples $(\mathbf{x}, \mathbf{y})$ is set to zero. Intuitively, this means the feature map has mapped the original points to points with at most exponentially small overlap in the Hilbert space and therefore will not contribute to the MMD.

It is also necessary to derive an expression for the gradient of the cost function. For the MMD, the gradient with respect to the kth parameter[14], carried by the kth unitary gate, $U_k(\theta_k)$, is given by:

$$\frac{\partial\mathcal{L}_{\mathrm{MMD}}}{\partial\theta_k} = 2\underset{\substack{\mathrm{a}\sim p_{\theta_k}^-\\ \mathrm{x}\sim p_\theta}}{\mathbb{E}}(\kappa(\mathrm{a},\mathrm{x})) - 2\underset{\substack{\mathrm{b}\sim p_{\theta_k}^+\\ \mathrm{x}\sim p_\theta}}{\mathbb{E}}(\kappa(\mathrm{b},\mathrm{x})) - 2\underset{\substack{\mathrm{a}\sim p_{\theta_k}^-\\ \mathrm{y}\sim \pi}}{\mathbb{E}}(\kappa(\mathrm{a},\mathrm{y})) + 2\underset{\substack{\mathrm{b}\sim p_{\theta_k}^+\\ \mathrm{y}\sim \pi}}{\mathbb{E}}(\kappa(\mathrm{b},\mathrm{y})), \tag{15}$$

where $p_{\theta_k}^\pm$ are output distributions generated by running the following auxiliary circuits[39,40] for each unitary gate, $U_k(\theta_k)$:



(16)

where $\theta_k^\pm := \theta_k \pm \pi/2$ and $U_{l:m} := U_l U_{l+1} \ldots U_{m-1} U_m$ are the unitary gates in the Born machine. This gradient occurs because the form of the unitary gates in our case are exponentiated Pauli operators $U_k(\theta_k) = \exp(i\theta_k\Sigma_k)$, with $\Sigma_k^2 = \mathbb{I}$. With the unitaries in this form, the gradient of the probabilities outputted from the parameterised state, with respect to a parameter θ, is given by[14,40]:

$$\frac{\partial p_{\boldsymbol{\theta}}(\mathbf{z})}{\partial \theta_k} = p_{\theta_k}^{-}(\mathbf{z}) - p_{\theta_k}^{+}(\mathbf{z}). \tag{17}$$

There is a slight difference between Eq. (17) and that of ref. [14], due to a different parameterisation of the unitaries above.

The gradients of the cost functions which we introduce next will also require the parameter-shifted circuits in Eq. (16). For more details on kernel methods and the MMD, see Supplementary Material Section II.

SD training

So far, we have only proposed a change of kernel in the MMD method of training QCIBMs. We now consider changing the cost function altogether. We endeavour to find costs which are efficient to compute for quantum models, yet stronger than MMD.

The first cost we propose is called the SD. SD has become popular for goodness-of-fit tests[41], i.e. testing whether samples come from a particular distribution or not, as opposed to the MMD, which is typically used for kernel two-sample tests[36]. This discrepancy is based on Stein's method[42], which is a way to bound distance metrics between probabilities including, for example, the other integral probability metrics (IPM) we utilise in this work. For details on IPMs, see Supplementary Material Section I.

We use the discrete version of the SD[43] since, in its original form[41], it only caters for the case where the distributions are supported over a continuous space. The discretisation is necessary since the QCIBM outputs binary strings and so the standard gradient w.r.t. a sample, $\mathbf{x}$, $\nabla_{\mathbf{x}}$, is undefined. As such, we need to use a discrete 'shift' operator, $\Delta_{\mathbf{x}}$, instead, which is an operator defined by $[\Delta_{\mathbf{x}} f(\mathbf{x})]_i := f(\mathbf{x}) - f(\neg_i \mathbf{x})$ for a function f, where $\neg_i$ flips the ith element of the binary vector $\mathbf{x}$.

Fortunately, the discretisation procedure is relatively straightforward (the necessary definitions and proofs can be found in Supplementary Material Section III). The discrepancy is derived[41,44] from the (discrete) Stein identity[43], given by:

$$\mathop{\mathbb{E}}_{\mathbf{x}\sim\pi}[\mathcal{A}_\pi \phi(\mathbf{x})] = \mathop{\mathbb{E}}_{\mathbf{x}\sim\pi}[\mathbf{s}_\pi(\mathbf{x})\phi(\mathbf{x}) - \Delta_{\mathbf{x}}\phi(\mathbf{x})] = 0, \tag{18}$$

$$\mathcal{A}_\pi \phi(\mathbf{x}) := \mathbf{s}_\pi(\mathbf{x})\phi(\mathbf{x}) - \Delta_{\mathbf{x}}\phi(\mathbf{x}), \tag{19}$$

where $\mathop{\mathbb{E}}_{\mathbf{x}\sim\pi}$ denotes the expectation value over the distribution, π. This holds for any function $\phi : \mathcal{X}^n \to \mathbb{C}$ and probability mass function π on $\mathcal{X}^n$. The function $s_\pi(\mathbf{x}) = \Delta_{\mathbf{x}} \log(\pi(\mathbf{x}))$ is the Stein score function of the distribution π, and $\mathcal{A}_\pi$ is a so-called Stein operator of π. Now, the SD cost function can be written in a kernelised form[41,43], similarly to the MMD:

$$\mathcal{L}_{\mathrm{SD}}(p_{\boldsymbol{\theta}}, \pi) := \mathbb{E}_{\mathbf{x},\mathbf{y}\sim p_{\boldsymbol{\theta}}}[\kappa_\pi(\mathbf{x},\mathbf{y})], \tag{20}$$

$$\begin{aligned} \kappa_\pi(\mathbf{x},\mathbf{y}) := {} & s_\pi(\mathbf{x})^T \kappa(\mathbf{x},\mathbf{y}) s_\pi(\mathbf{y}) - s_\pi(\mathbf{x})^T \Delta_{\mathbf{y}}^{*}\kappa(\mathbf{x},\mathbf{y}) \\ & -\Delta_{\mathbf{x}}^{*}\kappa(\mathbf{x},\mathbf{y})^T s_\pi(\mathbf{y}) + \mathrm{tr}(\Delta_{\mathbf{x},\mathbf{y}}^{*}\kappa(\mathbf{x},\mathbf{y})), \end{aligned} \tag{21}$$

where κ_π is the Stein kernel and κ is a usual positive semi-definite kernel. $\Delta_{\mathbf{x}}^{*}$ is a conjugate version of the operator $\Delta_{\mathbf{x}}$, but for our purposes, the behaviour of both $\Delta_{\mathbf{x}}^{*}$ and $\Delta_{\mathbf{x}}$ are identical. For completeness, we define it in generality in Supplementary Material Section III.

Just as above, the gradient (derived in an identical fashion to the MMD gradient Eq. (15) as is detailed in Supplementary Material Section III) of $\mathcal{L}_{\mathrm{SD}}$ with respect to the parameter, θ_k, is given by:

$$\frac{\partial \mathcal{L}_{\mathrm{SD}}}{\partial \theta_k} = \mathop{\mathbb{E}}_{\substack{\mathrm{x}\sim p_\theta^{-} \\ \mathrm{y}\sim p_\theta}}[\kappa_\pi(\mathrm{x},\mathrm{y})] - \mathop{\mathbb{E}}_{\substack{\mathrm{x}\sim p_\theta^{+} \\ \mathrm{y}\sim p_\theta}}[\kappa_\pi(\mathrm{x},\mathrm{y})] - \mathop{\mathbb{E}}_{\substack{\mathrm{x}\sim p_\theta \\ \mathrm{y}\sim p_\theta^{-}}}[\kappa_\pi(\mathrm{x},\mathrm{y})] - \mathop{\mathbb{E}}_{\substack{\mathrm{x}\sim p_\theta \\ \mathrm{y}\sim p_\theta^{+}}}[\kappa_\pi(\mathrm{x},\mathrm{y})]. \tag{22}$$

We show that almost every term in Eqs. (20) and (22) can be computed efficiently, even when the quantum kernel κ_{Q} from Eq. (13) is used in Eq. (21), that is, with the exception of the score function s_π with respect to the data distribution. The score contains an explicit dependence on the data distribution, π. If we are given oracle access to the probabilities, $\pi(\mathbf{y})$, then there is no issue and SD will be computable. Unfortunately, in any practical application this will not be the case.

To deal with such a scenario, we give two approaches to approximate the score via samples from π. The first of these we call the 'Identity' method since it inverts Stein's identity[45] from Eq. (18). We refer to the second as the 'Spectral' method since it uses a spectral decomposition[46] of a kernel to approximate the score. The latter approach uses the Nyström method[47], which is a technique used to approximately solve integral equations. We will only use the Spectral method in training the QCIBM in the numerical results in Fig. 3, since the Identity method does not give an immediate out-of-sample method to compute the score. Details of these methods can be found in Supplementary Material Section III.

Notice that, even with the difficulty in computing the score, the SD is still more suitable for training these models than the KL divergence as the latter requires computing the circuit probabilities, $p_{\boldsymbol{\theta}}(\mathbf{x})$, which is in general intractable, and so could not be computed for any data set.

SHD training

The second cost function we consider is the so-called SHD. This is a relatively new method to compare probability distributions[48–50], defined by the following:

$$\mathcal{L}_{\mathrm{SHD}}^{\epsilon}(p_{\boldsymbol{\theta}}, \pi) := \mathrm{OT}_\epsilon^c(p_{\boldsymbol{\theta}}, \pi) - \frac{1}{2}\mathrm{OT}_\epsilon^c(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}}) - \frac{1}{2}\mathrm{OT}_\epsilon^c(\pi, \pi), \tag{23}$$

$$\mathrm{OT}_\epsilon^c(p_{\boldsymbol{\theta}}, \pi) := \min_{U \in \mathcal{U}(p_{\boldsymbol{\theta}}, \pi)} \left(\sum_{\substack{(\mathbf{x},\mathbf{y}) \\ \in \mathcal{X}^d \times \mathcal{Y}^d}} c(\mathbf{x},\mathbf{y}) U(\mathbf{x},\mathbf{y}) + \epsilon \mathrm{KL}(U | p_\theta \otimes \pi) \right), \tag{24}$$

where $\epsilon \geq 0$ is a regularisation parameter, $c(x, y)$ is a Lipschitz 'cost' function, and $\mathcal{U}(p, \pi)$ is the set of all couplings between $p_{\boldsymbol{\theta}}$ and π, i.e. the set of all joint distributions, whose marginals with respect to $\mathbf{x}$ and $\mathbf{y}$ are $p_{\boldsymbol{\theta}}(\mathbf{x})$ and $\pi(\mathbf{y})$, respectively. The above cost function, $\mathcal{L}_{\mathrm{SHD}}^{\epsilon}$, is particularly favourable as a candidate because of its relationship to the theory of optimal transport[51] (OT), a method to compare probability distributions. It has become a major tool used to train models in the classical domain, for example with GANs[52] through a restriction of OT called the Wasserstein metric, which is derived from OT, when the cost ($c(\mathbf{x}, \mathbf{y})$) is chosen to be a metric on the space of $\mathcal{X}^n$.

We would like to use OT itself to train generative models, due to its metric properties. Unfortunately, OT has high computational cost and exponential sample complexity[53]. For this reason, the SHD was proposed in refs [48–50] to interpolate between OT and the MMD as a function of the regularisation parameter ϵ in Eq. (24). In particular, for the two extreme values of ϵ, we recover[48] both unregularised OT and the MMD:

$$\begin{array}{l} \underline{\longrightarrow 0}: \\ \qquad \mathcal{L}_{\mathrm{SHD}}^{0}(p_\theta, \pi) \longrightarrow \mathrm{OT}_0^c(p_\theta, \pi) \end{array}. \tag{25}$$

$$\begin{array}{l} \underline{\epsilon \longrightarrow \infty}: \\ \qquad \mathcal{L}_{\mathrm{SHD}}^{\epsilon}(p_{\boldsymbol{\theta}}, \pi) \longrightarrow \mathrm{MMD}(p_{\boldsymbol{\theta}}, \pi) : \kappa(\mathbf{x},\mathbf{y}) = -c(\mathbf{x},\mathbf{y}) \end{array}. \tag{26}$$

As before, we need a gradient of the $\mathcal{L}_{\mathrm{SHD}}^{\epsilon}$ with respect to the parameters, which is given by:

$$\frac{\partial \mathcal{L}_{\mathrm{SHD}}^{\epsilon}(p_{\boldsymbol{\theta}}, \pi)}{\partial \theta_k} = \mathop{\mathbb{E}}_{x\sim p_{\theta_k^{-}}}[\varphi(\mathbf{x})] - \mathop{\mathbb{E}}_{\mathbf{x}\sim p_{\theta_k^{+}}}[\varphi(\mathbf{x})], \tag{27}$$

where $\varphi(\mathbf{x})$ is a function that depends on the optimal solutions found to the regularised OT problem in Eq. (24). See Supplementary Material Section IV for more details on the SHD and its gradient.

Sinkhorn complexity

The sample complexity of the SHD is of great interest to us as we claim that the TV and the KL are not suitable to be directly used as cost functions. This is due to the difficulty of computing the outcome probabilities of quantum circuits efficiently. We now analyse why the MMD is a weak cost function and why the SHD should be used as an alternative. This will depend critically on the regularisation parameter ϵ, which allows a smooth interpolation between the OT metric and the MMD.

First, we address the computability of $\mathcal{L}^{\epsilon}_{\text{SHD}}$ and we find, due to the results of ref. [54], a somewhat 'optimal' value for ϵ, for which the sample complexity of $\mathcal{L}_{\text{SHD}}$ becomes efficient. Specifically, the mean error between $\mathcal{L}_{\text{SHD}}$ and its approximation $\hat{\mathcal{L}}^{\epsilon}_{\text{SHD}}$ for n qubits, computed using M samples, scales as:

$$\mathbb{E}|\mathcal{L}^{\epsilon}_{\text{SHD}} - \hat{\mathcal{L}}^{\epsilon}_{\text{SHD}}| = \mathcal{O}\left(\frac{1}{\sqrt{M}}\left(1 + \mathrm{e}^{\left(2\frac{n^2+n}{\epsilon}\right)}\right)\left(1 + \frac{1}{\epsilon^{\lfloor n/2 \rfloor}}\right)\right). \tag{28}$$

We show in Supplementary Material Section IV.1 that by choosing $\epsilon = \mathcal{O}(n^2)$, we get:

$$\mathbb{E}|\mathcal{L}^{\mathcal{O}(n^2)}_{\text{SHD}} - \hat{\mathcal{L}}^{\mathcal{O}(n^2)}_{\text{SHD}}| = \mathcal{O}\left(\frac{1}{\sqrt{M}}\right), \tag{29}$$

which is the same sample complexity as the MMD[55] but exponentially better than that of unregularised OT, which scales as $\mathcal{O}\left(1/M^{1/n}\right)$[53].

A similar result can be derived using a concentration bound[54], such that, with probability $1 - \delta$,

$$|\mathcal{L}^{\mathcal{O}(n^2)}_{\text{SHD}} - \hat{\mathcal{L}}^{\mathcal{O}(n^2)}_{\text{SHD}}| = \mathcal{O}\left(\frac{n}{\sqrt{M}}\log(1/\delta)^{1/2}\right), \tag{30}$$

where we have chosen the same scaling for ϵ as in Eq. (29). Therefore, we can choose an optimal theoretical value for the regularisation, such that $\mathcal{L}_{\text{SHD}}$ is sufficiently far from OT to be efficiently computable but perhaps still retains some of its favourable properties. It is likely in practice, however, that a much lower value of ϵ could be chosen without a blow up in sample complexity[49,54]. See Supplementary Material Section IV for derivations of the above results.

Second, we can relate the $\mathcal{L}_{\text{SHD}}$ to unregularised OT and TV via a sequence of inequalities. We have mentioned that the MMD is weak, meaning it provides a lower bound on TV in the following way[55]:

$$\text{TV}(p_{\boldsymbol{\theta}}, \pi) \geq \frac{\sqrt{\text{MMD}(p_{\boldsymbol{\theta}}, \pi)}}{\sqrt{C}}, \tag{31}$$

if $C := \sup_{\mathbf{x} \in \mathcal{X}^n} \kappa(\mathbf{x}, \mathbf{x}) < \infty$.

Note that for the two kernels introduced earlier:

$$\kappa_{\text{G}}(\mathbf{x}, \mathbf{x}) = \frac{1}{c}\sum_{c} e^{-\frac{1}{2\sigma_i}|\mathbf{x}-\mathbf{x}|^2} = \frac{1}{c}(c) = 1, \tag{32}$$

$$\kappa_{\text{Q}}(\mathbf{x}, \mathbf{x}) = |\langle\phi(\mathbf{x})|\phi(\mathbf{x})\rangle|^2 = |\langle 0|0\rangle^{\otimes n}|^2 = 1, \tag{33}$$

hence $C = 1$ and the lower bound is immediate.

In contrast, as is seen from the inequality on a discrete sample space in Eq. (34)[56], the Wasserstein metric (unregularised OT) provides an upper bound on TV, and hence we would expect it to be stronger than the MMD.

$$d_{\min}\text{TV}(p_{\boldsymbol{\theta}}, \pi) \leq \text{OT}^d_0(p_{\boldsymbol{\theta}}, \pi) \leq \text{diam}(\mathcal{X})\text{TV}(p_{\boldsymbol{\theta}}, \pi), \tag{34}$$

where $\text{diam}(\mathcal{X}^n) = \max\{d(\mathbf{x}, \mathbf{y}), \mathbf{x}, \mathbf{y} \in \mathcal{X}^n\}$, $d_{\min} = \min_{x \neq y} d(\mathbf{x}, \mathbf{y})$, and $d(\mathbf{x}, \mathbf{y})$ is the metric on the space, $\mathcal{X}^n$. This arises by choosing $c = d$ and $\epsilon = 0$ in Eq. (24). If, for instance, we were to choose $d(\mathbf{x}, \mathbf{y})$ to be the ℓ_1 metric between the binary vectors of length n (a.k.a. the Hamming distance), then we get that $d_{\min} = 1, \text{diam}(\mathcal{X}) = n$, and so:

$$\text{TV}(p_{\boldsymbol{\theta}}, \pi) \leq \text{OT}^{\ell_1}_0(p_{\boldsymbol{\theta}}, \pi) \leq n\,\text{TV}(p_{\boldsymbol{\theta}}, \pi). \tag{35}$$

Finally, we can examine the relationship induced by the regularisation parameter through the following inequality; Theorem 1 in ref. [54]:

$$0 \leq \text{OT}^c_\epsilon(p_{\boldsymbol{\theta}}, \pi) - \text{OT}^c_0(p_{\boldsymbol{\theta}}, \pi) \leq 2\epsilon\log\left(\frac{\mathrm{e}^2 LD}{n\epsilon}\right) \sim_{\epsilon\to 0} 2\epsilon\log(1/\epsilon), \tag{36}$$

where the size of the sample space is bounded by D, as measured by the metric, and L is the Lipschitz constant of the cost c. As detailed in Supplementary Material Section IV.1, we can choose $D = n$ and $L = n$:

$$0 \leq \text{OT}^{\ell_1}_\epsilon(p_{\boldsymbol{\theta}}, \pi) - \text{OT}^{\ell_1}_0(p_{\boldsymbol{\theta}}, \pi) \leq 2\epsilon\log\left(\frac{\mathrm{e}^2 n}{\epsilon}\right). \tag{37}$$

The log term will be positive as long as $\epsilon \leq n\mathrm{e}^2$, in which case regularised OT will give an upper bound for the Wasserstein metric, and hence the TV through Eq. (34), so we arrive at:

$$\text{TV}(p_{\boldsymbol{\theta}}, \pi) \leq \text{OT}^{\ell_1}_0(p_{\boldsymbol{\theta}}, \pi) \leq \text{OT}^{\ell_1}_{\epsilon \leq n\mathrm{e}^2}. \tag{38}$$

Unfortunately, comparing this with Eqs. (29) and (30), we can see that, with this scaling of ϵ, the sample complexity would pick up an exponential dependence on the dimension, n, so it would not be efficiently computable. We comment further on this point later.

Numerical performance

In Figs 2–4, we illustrate the superior performance of our alternative training methods, as measured by the TV distance. A lower TV indicates that the model is able to learn parameters which fit the true data more closely. TV was chosen as an objective benchmark for several reasons. First, it is typically the notion of distance that is required by quantum supremacy experiments where one wants to prove hardness of classical simulation. Second, we use it in the definitions of QLS. Finally, it is one of the strongest notions of convergence in probability one can ask for, so it follows that a training procedure that can more effectively minimise TV, in an efficient way, should be better for generative modelling.

We train the model on Rigetti's Forest platform[57] using both a simulator and the real quantum hardware, the Aspen QPU. Figure 2 illustrates the training of the model using the Gaussian (Eq. (12)) vs. the quantum kernel (Eq. (13)) for 4 qubits, and we see that the quantum kernel offers no significant advantage vs. training with a purely classical one. Figure 2a shows the TV as trained for 200 epochs, using both the classical and quantum kernels with various learning rates. Figure 2b shows the learned probabilities outputted after training with each kernel, and Fig. 2c shows the difference in the actual $\mathcal{L}_{\text{MMD}}$ itself while training with both methods. Interestingly, the latter behaviour is quite different for both kernels, with the quantum kernel initialising with much higher values of $\mathcal{L}_{\text{MMD}}$, whereas they both minimise TV in qualitatively the same way. This indicates that hardness of classical simulation (of computing the kernel) does not imply an advantage in learning.

On the other hand, a noticeable outperformance is observed for the SHD and the SD relative to training with the MMD (using a Gaussian kernel), as measured by TV in Fig. 3. Furthermore, we observed that the gap (highlighted in the inset in Fig. 3a) which separates the SHD and SD (red and blue lines) from the MMD (green, yellow, and cyan lines) grows as the number of qubits grows. Unfortunately, the Spectral method to approximate the

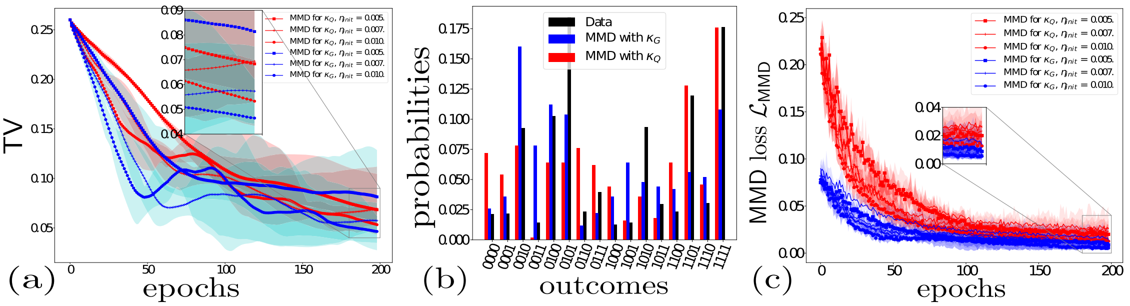


Fig. 2 The performance of the quantum kernel κ_Q (red) vs. the Gaussian kernel, κ_G (blue) (with $\eta_{\text{init}} = 0.1$) for 4 qubits. During training, we sample from the QCIBM and the data 500 times and use a minibatch size of 250. One epoch is one complete update of all parameters according to gradient descent. Error bars represent maximum, minimum, and mean values achieved over five independent training runs, with the same initial conditions on the same data samples. **a** TV difference achieved with both kernel methods during training. No observable or obvious advantage is seen in using the quantum kernel over the Gaussian one; in contrast, the Gaussian kernel seems to perform better on average. **b** Final learned probabilities with $\eta_{\text{init}} = 0.01$ using the Adam optimiser. **c** MMD computed using 400 samples as training points and 100 as test points (seen as the thin lines without markers), independent of the training data.

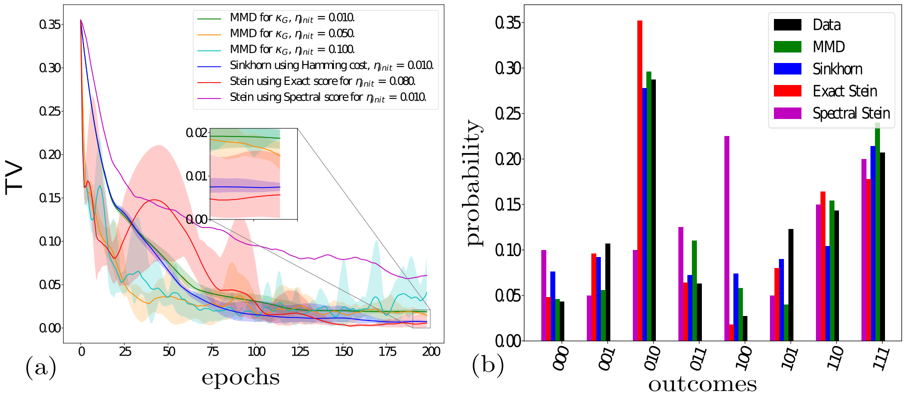


Fig. 3 MMD (cyan, yellow, green) vs. Sinkhorn (blue) and Stein training with Exact score function (red) and Spectral score method (magenta) for 3 qubits with fully connected topology on the Rigetti 3q−qvm, **trained on the data, Eq. (44).** Five hundred data points are used for training, with 400 used as a training set and 100 used as a test set. Plots show mean, maximum, and minimum values achieved over five independent training runs on the same data set. a TV difference between training methods, with regularisation parameter for SHD and 3 eigenvectors for Spectral Stein method. Both Sinkhorn divergence and Stein discrepancy are able to achieve a lower TV than the MMD. Inset shows region of outperformance on the order of ~0.01 in TV. We observe that the Spectral score method was not able to minimise TV as well as the exact Stein discrepancy, potentially indicating the need for better approximation methods. b Final learned probabilities of each training method. See Supplementary Material Section V for behaviour of corresponding cost functions.

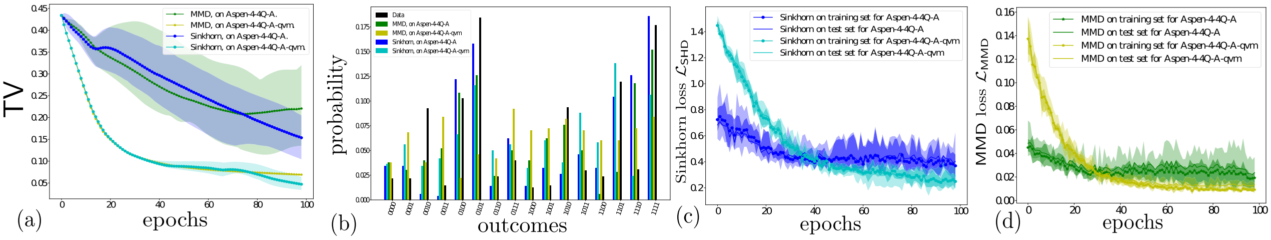


Fig. 4 MMD (green, yellow) vs. Sinkhorn (blue, cyan) for 4 qubits comparing performance on the real QPU (Aspen–4–4Q–A) vs. simulated behaviour on QVM (Aspen–4–4Q–A–qvm) using 500 samples and a batch size of 250, learning target data (black), and a initial learning rate for Adam as $\eta_{\text{init}} = 0.01$. a TV difference between training methods with regularisation parameter $\epsilon = 0.08$. **b** Final learned probabilities (black) indicates the probabilities of a random instance of the data distribution (see 'Methods') chosen. The probabilities given by the other bars are those achieved after training the model with either the MMD or SHD on the simulator or the physical Rigetti chip, on an average run. The probabilities of the model are generated by simulating the entire wavefunction after training. **c** $\mathcal{L}^{0.08}_{\text{SHD}}$ on QVM (cyan) vs. QPU (blue). **d** $\mathcal{L}_{\text{MMD}}$ on QVM (yellow) vs. QPU (green). In both latter cases, trained model performance on 100 test samples is seen as the thin lines without markers. Again it can be seen that the Sinkhorn divergence outperforms the MMD both simulated and on chip, with the deviation apparent towards the end of training. Similar behaviour observed after 100 epochs but not shown due to limited QPU time.

Stein score does not outperform the MMD, despite training successfully. The discrepancy between the true and approximate versions of the Stein score is likely due to the low number of samples used to approximate the score, with the number of samples limited by the computational inefficiency. We leave tuning the hyperparameters of the model in order to get better performance to future work.

This behaviour is shown to persist on the QPU, Fig. 4, where we show training of the model with both the MMD and SHD relative to TV, (Fig. 4a), the learned probabilities of both methods on, and off, the QPU (Fig. 4b), and the behaviour of the cost functions associated with both methods (Fig. 4c, d). This reinforces our theoretical argument that the SHD is able to better minimise TV to achieve superior results.

Given the performance noted above, we would recommend the SHD as the primary candidate for future training of these models, due to its simplicity and competitive performance. One should also note that we do not attempt to learn these data distributions exactly since we use a shallow fixed circuit structure for training (i.e. a QAOA circuit), which we do not alter. Better fits to the data could likely be achieved with deeper circuits with more parameters.

For extra numerical result demonstrating the performance of the learning algorithms, see Supplementary Material Section V, including a comparison between the quantum and Gaussian kernels for two qubits, similar to Fig. 2; the behaviour of the corresponding cost functions themselves associated with Fig. 3; the performance of the model for 4 qubits, similar to Fig. 3; and the results using a 3 qubit device, the Aspen–4–3Q–A. In all cases, the performance was qualitatively similar to that reported in the main text.

Hardness and quantum advantage

It is crucially important, not just for our purposes but for the design of QML algorithms in general, that the algorithm itself is providing some advantage over any classical one for the same task. This is the case for so-called coherent algorithms, like the HHL linear equation solver[3], which is BQP-complete, and therefore unlikely to be fully dequantised. However, such a proven advantage for near term QML algorithms is yet out of reach. We attempt to address such a question in two steps.

1. We show that, for a large number of parameter values, $\boldsymbol{\theta}$, our QCIBM circuits are 'hard'. That is to say, it cannot be efficiently simulated classically up to a multiplicative error, in the worst case. We also show that this holds for the auxiliary quantum circuits used for the gradient estimation, and hence the model may remain hard during training (although we do not know for sure).
2. We provide formal definitions for QLS, the ability of a quantum model to provably outperform all classical models in a certain task, and a potential pathway to prove such a thing.

The intuition behind point 2 is the following. If our QCIBM model could learn a target distribution π, which demonstrates quantum supremacy, by providing a quantum circuit C close enough to π (i.e. below a threshold error in TV), then the model would have demonstrated something that is classically infeasible. Else there would exist an efficient classical algorithm that can get close to π, which contradicts hardness.

Point 1 does not completely fit that intuition. For one thing, hardness is not known to hold for the required notion of additive error (i.e. TV distance) but only for multiplicative error. Also, even though the model is more expressive than any classical model[16], this does not imply that it could actually learn a hard distribution. On the other hand, it is easy to see why the converse would be true, if the QCIBM could learn a distribution that is hard to sample from classically, the underlying circuit must have, at some point, reached a circuit configuration for which the output distribution is hard to classically sample.

We can address point 1 informally (see Supplementary Material Section VI for the formal statements and proof) in three steps:

- If the parameters of the model are initialised randomly in $\{\boldsymbol{\alpha}\} = \{J_{ij}, b_k\} \in \{0, \frac{\pi}{8}, \dots, \frac{7\pi}{8}\}$ and final measurement angles are chosen such that $U_f(\boldsymbol{\Gamma}, \boldsymbol{\Delta}, \boldsymbol{\Sigma}) = H^{\otimes n}$, then the resulting QCIBM circuit class will be hard to simulate up to an additive error of 1/384 in TV distance, subject to a conjecture relating to the hardness of computing the Ising partition function[6].
- If certain configurations of the parameters are chosen to be either of the form, $(2l+1)\pi/kd$, where l and d are integers and k is a number that depends on the circuit family, or in the form $2\pi\nu$, where ν is irrational, then the resulting class of circuits will be hard to sample from classically, up to a multiplicative error, in the worst case.
- The circuits produced at each epoch as a result of the gradient updates will each result in a hard circuit class as long as the gradient updates are not chosen carelessly. In each epoch, if the update step is constrained in a way that the new value of the parameter $\theta_k^{d+1} = \theta_k^d - \eta\partial_{\theta_k}\mathcal{L}_B$ does not become rational, then the updated circuits will also belong to a class that is hard to simulate (a similar result can be shown for the case where the parameters are updated to keep within the form of $(2l+1)\pi/kd$). This is because the updates can simply be absorbed into the original gates, to give a circuit which has the same form. This holds also for the gradient-shifted circuits in Eq. (16) since these correspond to circuits whose parameters are updated as follows: $\theta_k^{d,\pm} \leftarrow \theta_k^d \pm \pi/2$.

We now provide definitions to meet the requirements of point 2, adapting definitions from distribution learning theory[25] for this purpose. Specifically, we say that a generative QML algorithm, $\mathcal{A} \in \mathsf{BQP}$ (with a small abuse of notation) has demonstrated QLS if there exists a class of probability distributions $\mathcal{D}_n$ over $\mathcal{X}^n$ (bit vectors of length n), for which there exists a metric d and a fixed ϵ such that $\mathcal{D}_n$ is $(d, \epsilon, \mathsf{BQP})$-learnable via $\mathcal{A}$ but not $(d, \epsilon, \mathsf{BPP})$-learnable (i.e. learnable by a purely classical algorithm). The task of the learning algorithm $\mathcal{A}$ is, given a target distribution $D \in \mathcal{D}_n$, to output, with high probability, a Generator, $\mathrm{GEN}_{D'}$, for a distribution D', such that D' is close to $D \in \mathcal{D}_n$ with respect to the metric d. For the precise definitions of learnability we employ, see Supplementary Material Section VII.

This framework is very similar to that of, and inspired by, probably approximately correct (PAC) learning, which has been well studied in the quantum case[58] but it applies more closely to the task of generative modelling. It is known that, in certain cases, the use of quantum computers can be beneficial to PAC learning but not generically[59]. Based on this, it is possible that there exist some classes of distributions that cannot be efficiently learned by classical computers (BPP algorithms) but that could be learned by quantum devices (BPQ algorithms). The motivation for this is exactly rooted in the question of quantum supremacy and illustrated crudely in Fig. 5b.

An initial attempt at QLS is as follows. As mentioned above, if random IQP circuits could be classically simulated to within a TV error of $\epsilon = 1/384$[6] in the worst case (with high probability over the choice of circuit), this would imply unlikely consequences for complexity theory. Now, if a generative quantum model was able to achieve a closeness in TV less than this constant value, perhaps by minimising one of the upper bounds in Eq. (38), then we could claim that this model had achieved something classically intractable. For example, if we make the following assumptions,

1. QCIBM could achieve a TV $< \delta$ to a target IQP distribution.
2. A classical probabilistic algorithm, C, could output a distribution q in polynomial time which was γ close in TV to the QCIBM, i.e. it could simulate it efficiently.

Then

$$\mathrm{TV}(p_{\mathrm{IQP}}, q) = \frac{1}{2}\sum_{\mathbf{x}} |p_{\mathrm{IQP}}(\mathbf{x}) - q(\mathbf{x})| \tag{39}$$

$$= \frac{1}{2}\sum_{\mathbf{x}} |p_{\mathrm{IQP}}(\mathbf{x}) - p_{\boldsymbol{\theta}}(\mathbf{x}) + p_{\boldsymbol{\theta}}(\mathbf{x}) - q(\mathbf{x})| \tag{40}$$

$$\leq \frac{1}{2}\sum_{\mathbf{x}} |p_{\mathrm{IQP}}(\mathbf{x}) - p_{\boldsymbol{\theta}}(\mathbf{x})| + \frac{1}{2}\sum_{\mathbf{x}} |p_{\boldsymbol{\theta}}(\mathbf{x}) - q(\mathbf{x})| \tag{41}$$

$$\leq \delta + \gamma \equiv \epsilon, \tag{42}$$

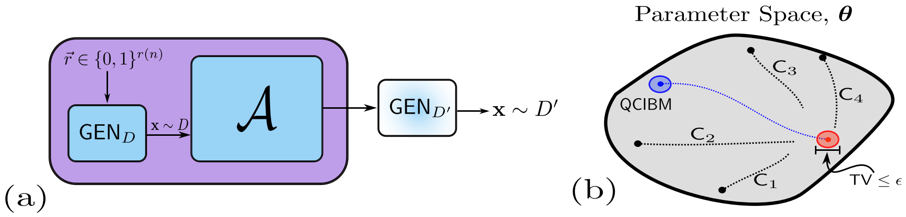


Fig. 5 Quantum learning supremacy in distribution learning. a Illustration of a learning procedure using a Generator. The algorithm $\mathcal{A}$ is given access to GEN_D, which provides samples, $\mathbf{x}\sim D$, and must output a Generator for a distribution that is close to the original. We allow the target generator to be classical, hence it may take as input a string of random bits of size polynomial in n, $r(n)$, if not able to generate its own randomness. **b** Crude illustration of quantum learning supremacy. No classical algorithm, C_i, should be able to achieve the required closeness in total variation to the target distribution, but the QCIBM (or similar) should be able to, for *some* class of target distributions. There should be some path in the parameter space of the QCIBM, $\boldsymbol{\theta}$, which achieves this.

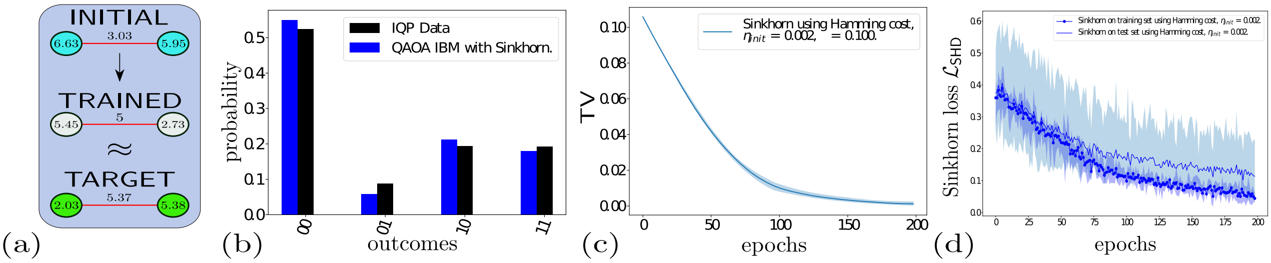


Fig. 6 Automatic compilation of IQP circuit to a $p=1$ QAOA circuit with two qubits using $\mathcal{L}^{\epsilon}_{\mathrm{SHD}}$ with $\epsilon=0.1$. Five hundred data samples were used with 400 used for a training set and 100 used as a test set. QCIBM circuit is able to mimic the target distribution well, even though actual parameter values and circuit families are different. Error bars represent mean, maximum, and minimum values achieved over five independent training runs on the same data set. **a** Initial (cyan) and trained (grey) QAOA circuit parameters for two qubits. Target IQP circuit parameters (green). Parameter values scaled by a factor of 10 for readability. **b** Final learned probabilities of QCIBM (QAOA) (blue) circuit vs. 'data' probabilities (IQP) (black). **c** Total variation distance and **d** Sinkhorn divergence for 400 training samples and 100 test samples, using a Hamming optimal transport cost.

where the third line follows from the triangle inequality. Therefore, C could simulate an IQP distribution also, and we arrive at a contradiction.

The major open question left by this work is whether QLS is possible at all; can a quantum model outperform all classical ones in generative learning? This idea motivated our search for metrics that upper bound TV but yet were efficiently computable and therefore could be minimised to efficiently learn distributions to a sufficiently small value of TV. Unfortunately, we can see from the exponential scaling observed in Eq. (38), which gives the upper bound on TV by regularised OT, that SHD will not provably achieve this particular task, despite achieving our primary goal of being stronger than the MMD for generative modelling. We briefly discuss avenues of future research in the 'Discussion' section, which could provide alternative routes to QLS.

Quantum compiling

As a concrete application of such a model outside the scope of classical generative modelling, we can use the QCIBM training to perform a type of 'weak' quantum circuit compilation. There are potentially other areas that could be studied using these tools or by applying techniques in generative modelling to other quantum information processing tasks, but this is beyond the scope of this work.

The major objective in this area is to compile a given target unitary, U, into one that consists exclusively of operations available to the native hardware of the quantum computer in question. For example, in the case of Rigetti's Aspen QPU, the native gates are $\{R_x(\pm\pi/2), R_z(\theta), CZ\}$[57,60], and any unitary which a user wishes to implement must be compiled into a unitary V that contains only these ingredients.

Potential solutions to this problem[60,61] involve approximating the target unitary by assuming that V is a parametric circuit built from the native gates, which can be trained by some optimisation strategy. We adopt a similar view here, but we do not require any extra quantum resources to perform the compilation. With this limitation, we make a trade-off in that we are not guaranteed to apply the same target unitary, only that the output distribution will be close to that produced by the target. Clearly this is a much weaker constraint than the task of direct compilation, since many unitaries may give rise to the same distribution, but it is much closer to the capabilities of near term devices. To illustrate this application, we train an QCIBM to learn the output distribution of a random IQP circuit when restricted to a QAOA architecture itself using $\mathcal{L}_{\mathrm{SHD}}$ as a cost function. The process is illustrated in Eq. (43), where we try to determine suitable QAOA parameters, $\{J_{ij}^{\mathrm{QAOA}}, b_k^{\mathrm{QAOA}}\}$, which reproduce the distribution observed from a set of random IQP parameters, $\{J_{ij}^{\mathrm{IQP}}, b_k^{\mathrm{IQP}}\}$.

$$\begin{aligned}&\mathrm{QCIBM}\left(\left\{J_{ij}^{\mathrm{QAOA}}, b_k^{\mathrm{QAOA}}\right\}, \{\Gamma_k = \tfrac{\pi}{4}\}, 0, 0\right)\\&\overset{Compile}{\longrightarrow} \mathrm{QCIBM}\left(\left\{J_{ij}^{\mathrm{IQP}}, b_k^{\mathrm{IQP}}\right\}, \left\{\Gamma_k = \tfrac{\pi}{2\sqrt{2}}\right\}, 0, \left\{\Sigma_k = \tfrac{\pi}{2\sqrt{2}}\right\}\right).\end{aligned} \tag{43}$$

The measurement unitary at the end of the circuit makes this process non-trivial, since this will give rise to significantly different distributions, even given the same parameters in U_z. We illustrate this in Fig. 6 using the Rigetti 2q–qvm and for three qubits in Supplementary Material Section V. We find that, even though the learned parameter values are different from the target, the resulting distributions are quite similar, as expected.

DISCUSSION

Providing provable guarantees of the superior performance of near term quantum computers relative to any classical device for some particular non-trivial application is an important milestone of the field. We have shown one potential route towards this goal by combining complexity–theoretic arguments[4,7,9], with an application in generative machine learning[14–16,62], and improved training methods of generative models. Specifically, we introduced the Ising Born machine, a restricted form of a quantum circuit Born machine. These models utilise the Born rule of quantum mechanics to train a parameterised quantum circuit as a generative machine learning model, in a hybrid manner.

We proved that the model cannot be simulated efficiently by any classical algorithm up to a multiplicative error in the output probabilities, which holds for many circuit families that may be encountered during gradient-based training. As such, this type of model is a good candidate for a provable quantum advantage in QML using NISQ devices. To formalise this intuition, we defined a notion of QLS to rigorously define what such an advantage would look like, in the context of machine learning.

We adapted novel training methods for generative modelling in two ways. First, by introducing quantum kernels to be evaluated on the quantum hardware and, second, by proposing and adapting new cost functions. In the case of SHD, we discussed its sample complexity and used this to define a somewhat optimal cost function through a judicious choice of the regularisation parameter. It is possible to choose this parameter such that the cost is efficiently computable even as the number of qubits grows. We showed numerically that these methods have the ability to outperform previous methods in the random data set we used as a test case.

Finally, we demonstrated an application of the model as a heuristic compiler to compile one quantum circuit into another via classical optimisation techniques, which has the advantage of requiring minimal quantum overhead. These techniques could potentially be adapted into methods to benchmark and verify near term quantum devices.

The major question that this work raises is whether or not a provable notion of quantum learning could be achievable for a particular data set, thereby solidifying a use case for quantum computers in the near term with provable advantage. The best prospect for this is the quantum supremacy distributions we know of (for example, IQP), but they are not efficiently testable[63]. Owing to this, they are also likely to not be efficiently learnable either, given the close relationship between distribution testing and learning[64]. Furthermore, we can see from the exponential scaling required in Eq. (38) for regularised OT to upper bound TV that other techniques are necessary to achieve QLS, since the methods we present here are not suited to this particular task, despite achieving our goal of being stronger than the MMD for generative modelling. However, this assumes that we have access only to classical samples from the distribution, and the possibility of gaining an advantage using quantum samples[38,58] is unexplored in the context of distribution learning.

METHODS

In this section, we detail the methods used to train the QCIBM to reproduce a given probability distribution. The target distribution is the one given by Eq. (44), which is used in both refs [65,66] to train versions of the quantum Boltzmann machine:

$$\pi(\mathbf{y}) := \frac{1}{T}\sum_{k=1}^{T} p^{n-d_{\mathrm{H}}(s_k,\mathbf{y})}(1-p)^{d_{\mathrm{H}}(s_k,\mathbf{y})}. \tag{44}$$

To generate this data, T binary strings of length n, written s_k and called ‘modes’, are chosen randomly. A sample $\mathbf{y}$ is then produced with a probability that depends on its Hamming distance $d_{\mathrm{H}}(s_k, \mathbf{y})$ to each mode. In all of the above, the Adam[67] optimiser was applied, using the suggested hyperparameters, i.e. $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1\times 10^{-8}$, and initial learning rate, η_{init}. This was chosen since it was found to be more robust to sampling noise[14].

In all of the numerical results, we used a QAOA structure as the underlying circuit in the QCIBM. Specifically, the parameters in U_f were chosen such that $\forall k$, $\Gamma_k = \pi/4$, $\Delta_k = 0$, and $\Sigma_k = 0$. The Ising parameters $\{J_{ij}, b_k\}$ were initialised randomly.

For the SD, we used 3 Nyström eigenvectors to approximate the Spectral score in Fig. 3 for 3 qubits, and 6 eigenvectors for 4 qubits. In all cases when using the MMD with a Gaussian kernel, we chose the bandwidth parameters, $\sigma = [0.25,\ 10,\ 1000]$[14]. Note that this article was previously published as a preprint[68].

DATA AVAILABILITY

Data and simulations presented in this work are available from the corresponding author upon request. Code used in this work is available from the corresponding author upon request or on Github[69].

Received: 29 August 2019; Accepted: 12 May 2020;
Published online: 08 July 2020

REFERENCES

1. Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum* **2**, 79 (2018).
2. Shor, P. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* **26**, 1484–1509 (1997).
3. Harrow, A. W., Hassidim, A. & Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **103**, 150502 (2009).
4. Bremner, M. J., Jozsa, R. & Shepherd, D. J. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proc. R. Soc. Lond. A* **467**, 459–472 (2011).
5. Gao, X., Wang, S.-T. & Duan, L.-M. Quantum supremacy for simulating a translation-invariant Ising spin model. *Phys. Rev. Lett.* **118**, 040502 (2017).
6. Bremner, M. J., Montanaro, A. & Shepherd, D. J. Average-case complexity versus approximate simulation of commuting quantum computations. *Phys. Rev. Lett.* **117**, 080501 (2016).
7. Aaronson, S. & Arkhipov, A. The computational complexity of linear optics. *Theory Comput.* **9**, 143–252 (2013).
8. Farhi, E. & Harrow, A. W. Quantum supremacy through the quantum approximate optimization algorithm. Preprint at http://arxiv.org/abs/1602.07674 (2016).
9. Boixo, S. et al. Characterizing quantum supremacy in near-term devices. *Nat. Phys.* **14**, 595–600 (2018).
10. Arute, F. et al. Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
11. Maron, M. E. Automatic indexing: an experimental inquiry. *J. ACM* **8**, 404–417 (1961).
12. Goodfellow, I. J. et al. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27* (eds Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D. & Weinberger, K. Q.), pp. 2672–2680 (Curran Associates, Inc., 2014).
13. Cheng, S., Chen, J. & Wang, L. Information perspective to probabilistic modeling: Boltzmann machines versus born machines. *Entropy* **20**, 583 (2018).
14. Liu, J.-G. & Wang, L. Differentiable learning of quantum circuit Born machines. *Phys. Rev. A* **98**, 062324 (2018).
15. Benedetti, M. et al. A generative modeling approach for benchmarking and training shallow quantum circuits. *npj Quantum Inf.* **5**, 1–9 (2019).
16. Du, Y., Hsieh, M.-H., Liu, T. & Tao, D. The expressive power of parameterized quantum circuits. Preprint at http://arxiv.org/abs/1810.11922 (2018).
17. Zeng, J., Wu, Y., Liu, J.-G., Wang, L. & Hu, J. Learning and inference on generative adversarial quantum circuits. *Phys. Rev. A* **99**, 052306 (2019).
18. Romero, J. & Aspuru-Guzik, A. Variational quantum generators: generative adversarial quantum machine learning for continuous distributions. Preprint at http://arxiv.org/abs/1901.00848 (2019).
19. Benedetti, M., Lloyd, E., Sack, S. & Fiorentini, M. Parameterized quantum circuits as machine learning models. *Quantum Sci. Technol.* **4**, 043001 (2019).
20. Tang, E. Quantum-inspired classical algorithms for principal component analysis and supervised clustering. Preprint at http://arxiv.org/abs/1811.00414 (2018).
21. Tang, E. A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, 217–228 (2019).
22. Andoni, A., Krauthgamer, R. & Pogrow, Y. On solving linear systems in sublinear time. Preprint at http://arxiv.org/abs/1809.02995 (2018).
23. Chia, N.-H., Lin, H.-H. & Wang, C. Quantum-inspired sublinear classical algorithms for solving low-rank linear systems. Preprint at http://arxiv.org/abs/1811.04852 (2018).
24. Gilyén, A., Lloyd, S. & Tang, E. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension. Preprint at http://arxiv.org/abs/1811.04909 (2018).

25. Kearns, M. et al. On the learnability of discrete distributions. In *Proc. Twenty-sixth Annual ACM Symposium on Theory of Computing* 273–282 (ACM, New York, NY, 1994).
26. Shepherd, D. & Bremner, M. J. Temporally unstructured quantum computation. *Proc. R. Soc. A.* https://doi.org/10.1098/rspa.2008.0443 (2009).
27. Farhi, E., Goldstone, J. & Gutmann, S. A Quantum approximate optimization algorithm. Preprint at http://arxiv.org/abs/1411.4028 (2014).
28. Farhi, E., Goldstone, J., Gutmann, S. & Sipser, M. Quantum computation by adiabatic evolution. Preprint at http://arxiv.org/abs/quant-ph/0001106 (2000).
29. Bremner, M. J., Montanaro, A. & Shepherd, D. J. Achieving quantum supremacy with sparse and noisy commuting quantum computations. *Quantum* **1**, 8 (2017).
30. Fujii, K. & Morimae, T. Commuting quantum circuits and complexity of Ising partition functions. *New J. Phys.* **19**, 033003 (2017).
31. Leyton-Ortega, V., Perdomo-Ortiz, A. & Perdomo, O. Robust implementation of generative modeling with parametrized quantum circuits. Preprint at http://arxiv.org/abs/1901.08047 (2019).
32. Hamilton, K. E., Dumitrescu, E. F. & Pooser, R. C. Generative model benchmarks for superconducting qubits. *Phys. Rev. A* **99**, 062323 (2019).
33. Lloyd, S. & Weedbrook, C. Quantum generative adversarial learning. *Phys. Rev. Lett.* **121**, 040502 (2018).
34. Dallaire-Demers, P.-L. & Killoran, N. Quantum generative adversarial networks. *Phys. Rev. A* **98**, 012324 (2018).
35. Borgwardt, K. M. et al. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics* **22**, e49–e57 (2006).
36. Gretton, A., Borgwardt, K. M., Rasch, M., Schölkopf, B. & Smola, A. J. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems 19* (eds. Schölkopf, B., Platt, J. C. & Hoffman, T.) 513–520 (MIT Press, 2007).
37. Havlíček, V. et al. Supervised learning with quantum-enhanced feature spaces. *Nature* **567**, 209–212 (2019).
38. Schuld, M. & Petruccione, F. *Supervised Learning with Quantum Computers. Quantum Science and Technology* (Springer International Publishing, 2018).
39. Mitarai, K., Negoro, M., Kitagawa, M. & Fujii, K. Quantum circuit learning. *Phys. Rev. A* **98**, 032309 (2018).
40. Schuld, M., Bergholm, V., Gogolin, C., Izaac, J. & Killoran, N. Evaluating analytic gradients on quantum hardware. *Phys. Rev. A* **99**, 032331 (2019).
41. Liu, Q., Lee, J. D. & Jordan, M. A Kernelized Stein Discrepancy for Goodness-of-fit Tests. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, 276–284 (JMLR.org, New York, NY, 2016).
42. Stein, C. A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. In *Proc. Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory.* 583–602 (University of California Press, Berkeley, CA, 1972).
43. Yang, J., Liu, Q., Rao, V. & Neville, J. Goodness-of-fit testing for discrete distributions via Stein discrepancy. In *Proc. 35th International Conference on Machine Learning*, vol. 80 of *Proceedings of Machine Learning Research* (eds Dy, J. & Krause, A.) 5561–5570 (PMLR, Stockholm, 2018).
44. Gorham, J. & Mackey, L. Measuring sample quality with Stein's method. In *Advances in Neural Information Processing Systems 28* (eds Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M. & Garnett, R.) 226–234 (Curran Associates, Inc., 2015).
45. Li, Y. & Turner, R. E. Gradient estimators for implicit models. In *6th International Conference on Learning Representations (ICLR) 2018*, Vancouver, BC, Canada (OpenReview.net, 2018).
46. Shi, J., Sun, S. & Zhu, J. A spectral approach to gradient estimation for implicit distributions. *in Proceedings of the 35th International Conference on Machine Learning, Proceedings of Machine Learning Research*, 4644–4653, (eds Jennifer, D. y. & Andreas, K.), (PMLR, 2018).
47. Nyström, E. J. Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben. *Acta Math.* **54**, 185–204 (1930).
48. Ramdas, A., Trillos, N. G. & Cuturi, M. On wasserstein two-sample testing and related families of nonparametric tests. *Entropy* **19**, 47 (2017).
49. Genevay, A., Peyre, G. & Cuturi, M. Learning generative models with Sinkhorn divergences. In *Proc. Twenty-First International Conference on Artificial Intelligence and Statistics*, Vol. 84 (eds Storkey, A. & Perez-Cruz, F.) 1608–1617 (PMLR, Playa Blanca, 2018).
50. Feydy, J. et al. Interpolating between optimal transport and MMD using Sinkhorn divergences. In *Proc. Machine Learning Research*, Vol. 89 (eds Chaudhuri, K. & Sugiyama, M.) 2681–2690 (PMLR, 2019).
51. Villani, C. *Optimal Transport: Old and New* [Grundlehren der mathematischen Wissenschaften] (Springer, Berlin, 2009).
52. Arjovsky, M., Chintala, S. & Bottou, L. Wasserstein Generative Adversarial Networks. In *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research 70, 214–223 (eds Doina, P. & Yee, W. T.), International Convention Centre, Sydney, Australia, (PMLR, 2017).
53. Dudley, R. M. The speed of mean Glivenko-Cantelli convergence. *Ann. Math. Stat.* **40**, 40–50 (1969).
54. Genevay, A., Chizat, L., Bach, F., Cuturi, M. & Peyré, G. Sample complexity of Sinkhorn divergences. In *Proceedings of Machine Learning Research* 89, 1574–1583 (eds Chaudhuri, K. & Sugiyama, M.), (PMLR, 2019)
55. Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Schölkopf, B. & Lanckriet, G. R. G. On integral probability metrics, phi-divergences and binary classification. Preprint at http://arxiv.org/abs/0901.2698 (2009).
56. Gibbs, A. L. & Su, F. E. On choosing and bounding probability metrics. *Int. Stat. Rev.* **70**, 419–435 (2002).
57. Smith, R. S., Curtis, M. J. & Zeng, W. J. A practical quantum instruction set architecture. Preprint at http://arxiv.org/abs/1608.03355 (2016).
58. Arunachalam, S. & de Wolf, R. Guest column: A survey of quantum learning theory. *ACM SIGACT News* **48**, 41–67 (2017).
59. Arunachalam, S., Grilo, A. B. & Sundaram, A. Quantum hardness of learning shallow classical circuits. Preprint at http://arxiv.org/abs/1903.02840 (2019).
60. Khatri, S. et al. Quantum-assisted quantum compiling. *Quantum* **3**, 140 (2019).
61. Jones, T. & Benjamin, S. C. Quantum compilation and circuit optimisation via energy dissipation. Preprint at http://arxiv.org/abs/1811.03147 (2018).
62. Gao, X., Zhang, Z. & Duan, L. An efficient quantum algorithm for generative machine learning. *Sci. Adv.* 12, https://doi.org/10.1126/sciadv.aat9004 (2018).
63. Hangleiter, D., Kliesch, M., Eisert, J. & Gogolin, C. Sample complexity of device-independently certified "quantum supremacy". *Phys. Rev. Lett.* **122**, 210502 (2019).
64. Goldreich, O., Goldwasser, S. & Ron, D. Property testing and its connection to learning and approximation. *J. ACM* **45**, 653–750 (1998).
65. Amin, M. H., Andriyash, E., Rolfe, J., Kulchytskyy, B. & Melko, R. Quantum Boltzmann machine. *Phys. Rev. X* **8**, 021050 (2018).
66. Verdon, G., Broughton, M. & Biamonte, J. A quantum algorithm to train neural networks using low-depth circuits. Preprint at http://arxiv.org/abs/1712.05304 (2017).
67. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. In 3rd International Conference on Learning Representations, (ICLR) 2015, (eds Yoshua, B. & Yann, L.) (San Diego, CA, USA, 2015)
68. Coyle, B., Mills, D., Danos, V. & Kashefi, E. The Born supremacy: quantum advantage and training of an Ising Born machine. Preprint at http://arxiv.org/abs/1904.02214 (2019).
69. Coyle, B. IsingBornMachine. https://zenodo.org/record/3779865#.XqvfknVKhrk (2020).

ACKNOWLEDGEMENTS

B.C. thanks Andru Gheorghiu for useful discussions and title suggestion. We also thank Jean Feydy, Patric Fulop, Vojtech Havlicek, and Jiasen Yang for clarifying pointers. We thank Atul Mantri for comments on the manuscript. This work was supported by the Engineering and Physical Sciences Research Council (grants EP/L01503X/1, EP/N003829/1), EPSRC Centre for Doctoral Training in Pervasive Parallelism at the University of Edinburgh, and School of Informatics and Entrapping Machines (grant FA9550-17-1-0055). We also thank Rigetti Computing for the use of their quantum compute resources, and views expressed in this paper are those of the authors and do not necessarily reflect the views or policies of Rigetti Computing.

AUTHOR CONTRIBUTIONS

B.C. devised the theoretical aspects of the work and wrote the code for the numerical results with help from D.M.; D.M. contributed to the learning supremacy definitions. E.K. and V.D. supervised the work. All authors contributed to the manuscript writing.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Supplementary information is available for this paper at https://doi.org/10.1038/s41534-020-00288-9.

Correspondence and requests for materials should be addressed to B.C.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2020