Quantum generative adversarial networks

Pierre-Luc Dallaire-Demers* and Nathan Killoran
Xanadu, 372 Richmond Street W, Toronto, Ontario M5V 1X6, Canada
(Dated: May 2, 2018)

Quantum machine learning is expected to be one of the first potential general-purpose applications of near-term quantum devices. A major recent breakthrough in classical machine learning is the notion of generative adversarial training, where the gradients of a discriminator model are used to train a separate generative model. In this work and a companion paper, we extend adversarial training to the quantum domain and show how to construct generative adversarial networks using quantum circuits. Furthermore, we also show how to compute gradients – a key element in generative adversarial network training – using another quantum circuit. We give an example of a simple practical circuit ansatz to parametrize quantum machine learning models and perform a simple numerical experiment to demonstrate that quantum generative adversarial networks can be trained successfully.

I. INTRODUCTION

Deep learning [1, 2] is currently transforming the way we process large-scale complex data with computers. Deep neural networks are now able to perform image and speech recognition with accuracies at a similar level to humans [3]. One of the most exciting recent developments in deep learning is *generative adversarial networks* (GANs) [4]. These are a class of deep neural networks which have shown great promise for the task of *generative* machine learning, that is, learning to generate realistic data samples. Despite the initial difficulties of training these models [5], GANs have quickly found applications in many fields [6], including image generation [7], super-resolution [8], image-to-image translation [9], generation of 3D objects [10], text generation [11], and the generation of synthetic data for chemistry [12], biology [13], and physics [14].

The goal of GANs is to simultaneously train two functions: a *generator* G, and a *discriminator* D, through an *adversarial learning* strategy. The goal for the generator is to generate new sample data from some specific domain, such as images, text, or audio. The outputs from the generator should not be completely unstructured; rather, they should be plausible samples that reflect the properties of real-world data (e.g., realistic images or natural language). The goal of the discriminator is to distinguish fake data samples which were created by the generator from those which are real.

The training strategy for GANs is anchored in game theory and is analogous to the competition between counterfeiters who have to produce fake currencies and the police who have to design methods to distinguish increasingly more convincing counterfeits from the real ones. This game has a Nash equilibrium where the fake coins become indistinguishable from the real ones and the authorities can no longer devise a method to discriminate the real currencies from the generated ones [4] . Interestingly, theoretical proofs regarding the optimal points of adversarial training assume that the generator and discriminator have infinite capacity [4], i.e., they can encode arbitrary functions or probability distributions. Yet it is widely believed that classical computers cannot efficiently solve certain hard problems, so these optimal points may be intrinsically out of reach of classical models in many cases of interest.

Quantum computers [15, 16] have the potential to solve problems believed to be beyond the reach of classical computers, such as factoring large integers [17]. Realistic near-term quantum devices [18] may be able to speed up difficult optimization and sampling problems, even if the full power of fault-tolerant devices may not be available for several years. For instance, variational quantum algorithms [19–23], such as the variational quantum eigensolver (VQE), have been demonstrated with great success in the field of quantum chemistry. Currently, these ideas and algorithms are being extended to the domain of quantum machine learning [24–36], which could also benefit from a quantum advantage. Since many machine learning algorithms are naturally robust to noise, this direction is a promising application for near-term imperfect quantum devices.

In this paper, we introduce *QuGANs*, the quantum version of generative adversarial networks. The paper has the following structure. In Section II A, we generalize the model structure of classical generative adversarial networks [4] to define the quantum mechanical equivalent – QuGANs – and provide the cost function for training. A key ingredient for GANs is that the discriminator provides a gradient which the generator can use for gradient-based learning. In Section II B, we present a general formalism for computing exact gradients of quantum optimization and machine learning problems using quantum circuits. We then show how these gradients can be combined with a classical optimization routine to train QGANs in Section II C. Finally, we provide an example quantum circuit for both the generator and discriminator in Section II D and show that QuGANs can be trained in practice with a simple proof-of-principle numerical experiment in Section II E.

We will explore the practical issues of QuGANs by

* pierre-luc@xanadu.ai

explicitly constructing quantum circuits for the generator and discriminator and proposing quantum methods for computing the gradients of these circuits. A more in-depth theoretical exploration of quantum adversarial learning can be found in the companion paper [37].

II. TRAINING QUGANS

A. The structure of GANs and QuGANs

1. *Classical GANs*

We first provide a high-level overview of the GAN architecture [4]. We suppose that the real-world data comes from some fixed distribution $p_R(x)$, generated by some (potentially complex and unknown) process R. The generator – parameterized by a vector of real-valued parameters $\vec{\theta}_G$ – takes as input an unstructured random variable z (typically drawn from a normal or uniform distribution). G transforms this noise source into data samples $x = G(\vec{\theta}_G, z)$, creating the generator distribution $p_G(x)$. In the ideal case of a perfectly trained generator G, the discriminator would not be able to decide whether a given sample x came from $p_G(x)$ or from $p_R(x)$. Therefore, the task of training G corresponds to the task of maximizing the probability that D misclassifies a generated sample as an element of the real data. On the other hand, the discriminator – parameterized by a vector of real-valued parameters $\vec{\theta}_D$ – takes as input either real data examples $x \sim p_R(x)$ or fake data samples $x \sim p_G(x)$. D's goal is to discriminate between these two classes, outputting a binary random variable. Training D thus corresponds to maximizing the probability of successfully classifying real data, while minimizing the probability of misclassifying fake data.

We will formalize QuGANs as a quantum generalization of conditional GANs [38]. Conditional GANs generate samples from a *conditional* distribution $p(x|\lambda)$ (conditioned on labels λ), rather than the unconditional distribution $p(x)$ of vanilla GANs. Conditional GANs reduce to vanilla GANs in the case where the label is uninformative about the data, i.e., $p(x|\lambda) = p(x)$ for all x and λ. A possible motivation for using the conditional approach comes from performing quantum chemistry calculations on quantum computers. For example, one could have a list of VQE state preparations for molecules, labeled by their physical properties. A well-trained QuGAN could produce new molecular states which also have the same properties but were not in the original dataset. In another context, a QuGAN could be used to compress time evolution gate sequences [39] for different time steps to use in larger quantum simulations.

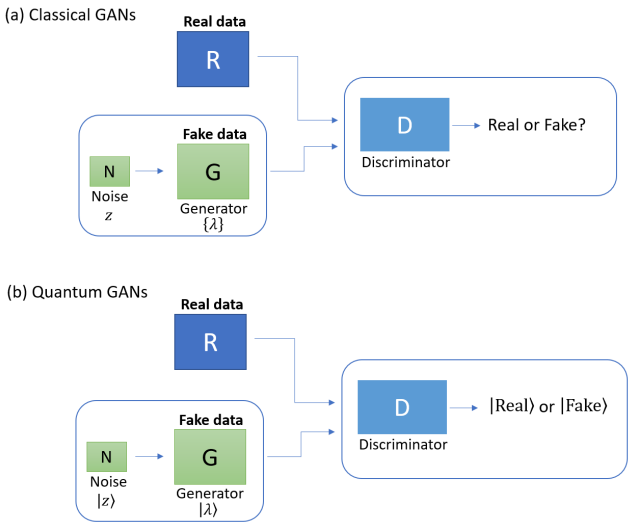

Figure 1. In (a), we show the building blocks of classical GANs. A discriminator must determine whether the samples it is given are produced by a real source R or a generator $G(z)$ equipped with a source of noise z. In (b), a quantum discriminator must decide whether the quantum state it receives at its input come from a real quantum data source R or a quantum generator $G(|z\rangle)$ with a quantum noise vector $|z\rangle$. The discriminator yields its output as a quantum state $|\text{Real}\rangle$ or $|\text{Fake}\rangle$.

2. *Quantum GANs*

We will now generalize these ideas to the quantum setting. In Figure 1, we highlight the structural similarities of classical and quantum GANs. For the quantum case, suppose we are given a data source R which, given a label $|\lambda\rangle$, outputs a density matrix ρ_λ^R into a register containing n subsystems, i.e.,

$$R(|\lambda\rangle) = \rho_\lambda^R. \tag{1}$$

The general aim of training a GAN is to find a generator G which mimics the real data source R. In the quantum case, we define G to be a variational quantum circuit whose gates are parametrized by a vector $\vec{\theta}_G$. The generator takes as input the label $|\lambda\rangle$ and an additional state $|z\rangle$, and produces a quantum state,

$$G(\vec{\theta}_G, |\lambda, z\rangle) = \rho_\lambda^G(\vec{\theta}_G, z), \tag{2}$$

where ρ_λ^G is output on a register containing n subsystems, similar to the real data.

The role of the extra input state $|z\rangle$ is two-fold. On one hand, it can be seen as a source of unstructured noise which provides entropy within the distribution of generated data. For instance, we could have a generator which is unitary, producing a fixed state $\rho_\lambda^G(\vec{\theta}_G, z) = |\psi_\lambda(z)\rangle\langle\psi_\lambda(z)|$ for each λ and $|z\rangle$. By allowing the input $|z\rangle$ to randomly fluctuate, we can create more than one output state for each label. On the other hand, the

variable $|z\rangle$ can serve as a control for the generator. By tuning $|z\rangle$, we can transform the output state prepared by the generator, varying properties of the generated data which are not captured by the labels λ. During training, the generator should learn to encode the most important intra-label factors of variation with $|z\rangle$. While the first role could have been accomplished via coupling the generator to a bath, the second role requires $|z\rangle$ to be under our control, even if we endow it with no explicit structure during training.

As in the classical case, the training signal of the generator is provided by a discriminator D, made up of separate quantum circuit parametrized by a vector $\vec{\theta}_D$. The task of D is to determine whether a given input state was created by the real data source R or the generator G, whereas the task of G is to fool D into accepting its output as being real. If the input was created by R, then D should output $|\text{real}\rangle$ in its output register, otherwise it should output $|\text{fake}\rangle$. The discriminator is also allowed to do operations on an internal workspace. In order to force G to respect the supplied labels, the discriminator is also given an unaltered copy of the label $|\lambda\rangle$.

The optimization objective for QuGAN training can be formalized as the adversarial task $\min_{\vec{\theta}_G} \max_{\vec{\theta}_D} V(\vec{\theta}_D, \vec{\theta}_G)$, or:

$$\min_{\vec{\theta}_G} \max_{\vec{\theta}_D} \frac{1}{\Lambda} \sum_{\lambda=1}^{\Lambda} \Pr\Big(\Big(D(\vec{\theta}_D, |\lambda\rangle, R(|\lambda\rangle)) = |\text{real}\rangle \Big) \cap \Big(D(\vec{\theta}_D, |\lambda\rangle, G(\vec{\theta}_G, |\lambda, z\rangle)) = |\text{fake}\rangle \Big) \Big). \quad (3)$$

For classical GANs, the optimization task is traditionally defined with log-likelihood functions but it is more convenient to define a cost function linear in the output probabilities of D in the quantum case since we want to optimize a function which is linear in some expectation value. Since the logarithmic function is convex, the optimal points are the same. Finally, for simplicity, the formula above assumes that the labels are countable, with cardinality Λ, though this could be relaxed.

The heuristic of the algorithm is illustrated in Figure 2, where the quantum circuit is divided into 6 operationally defined registers. The real source R and the generator G are given a label $|\lambda\rangle$ in the s-subsystem register **Label R|G**, an initial blank state $|0\rangle^{\otimes n}$ on the n-subsystem register **Out R|G** and a noise vector $|z\rangle$ on the m-subsystem register **Bath R|G**. In this work, we assume that R is a purified unitary operation on $s+n+m$ subsystems. In general, the real source may be a physical device entangled with an unknown number of environmental degrees of freedom m', with $m' \neq m$. With no loss of generality, we can assume that the **Bath R|G** register is initialized in the reference state $|0\rangle^{\otimes m}$ when the source is R as the entropy can be provided by the environment. We assume that the discriminator does not have access to the **Bath R|G** register.

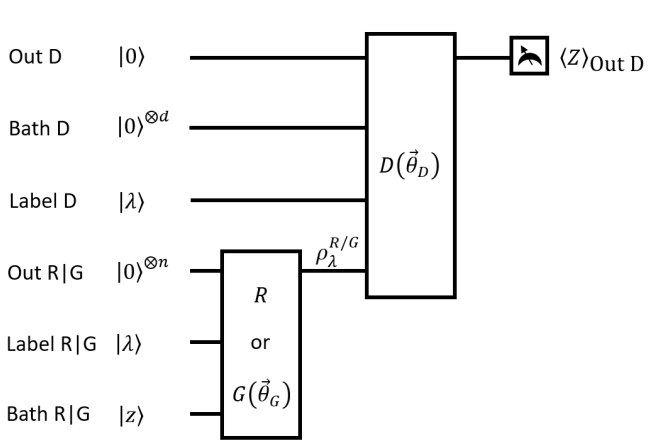

Figure 2. The general structure of QuGANs. The real source R or the parametrized generator $G(\vec{\theta}_G)$ is applied on an initial state $|0, \lambda, z\rangle$ respectively defined on the **Label R|G**, **Out R|G** and **Bath R|G** registers. The discriminator $D(\vec{\theta}_D)$ uses the information $\rho_\lambda^{R/G}$ from the source and an initial resource state $|0, 0, \lambda\rangle$ defined on the **Out D**, **Bath D** and **Label D** registers. D outputs its answer $|\text{real}\rangle$ or $|\text{fake}\rangle$ in the **Out D** register. The expectation value $\langle Z\rangle_{\text{Out D}}$ is proportional to the probability that D outputs $|\text{real}\rangle$.

D outputs its answer $|\text{real}\rangle$ or $|\text{fake}\rangle$ on the register **Out D**. It is given the state of the source through register **Out R|G**. The workspace of the discriminator is defined on the d-subsystem register **Bath D** and a reference copy of the label $|\lambda\rangle$ is fed through the s-subsystem register **Label D**. Finally, the expectation value of the operator

$$Z \equiv |\text{real}\rangle\langle\text{real}| - |\text{fake}\rangle\langle\text{fake}| \quad (4)$$

on the **Out D** register is proportional to $\Pr\Big(D(\vec{\theta}_D, |\lambda\rangle, R(|\lambda\rangle)) = |\text{real}\rangle\Big)$ and can be used to define the optimization problem (3) in a fully quantum mechanical setting.

3. *The quantum cost function*

We will follow the flow of the training process as illustrated in Figure 3 to rewrite and analyze the quantum version of the cost function (3). At the beginning of the algorithm, the discriminator and the generator are respectively initialized by the (arbitrary) parameters $(\vec{\theta}_D^0, \vec{\theta}_G^0)$. The quantum computer of Figure 2 is initialized in the state

$$\rho_\lambda^0(z) = (|0\rangle\langle 0|)^{\otimes d+1} \otimes |\lambda\rangle\langle\lambda| \otimes (|0\rangle\langle 0|)^{\otimes n} \otimes |\lambda\rangle\langle\lambda| \otimes |z\rangle\langle z|. \quad (5)$$

If only either R or G were systematically fed into D, the optimal strategy of the latter to maximize the cost function (3) would be to trivially output a constant answer, which is not desirable. In order to make sure that D cannot rely on the statistics of the choice of the source to

determine its answer, the choice of R or G can be made by the toss of a fair coin. The unitary operations corresponding to the sources R and $G(\vec{\theta}_G)$ acting on the whole quantum computer have the respective form

$$\begin{aligned} U_R &= I^{\otimes(1+d+s)} \otimes R, \\ U_G(\vec{\theta}_G) &= I^{\otimes(1+d+s)} \otimes G(\vec{\theta}_G). \end{aligned} \tag{6}$$

After the chosen source has been applied, the quantum computer is in the corresponding state

$$\begin{aligned} \rho_\lambda^R &= U_R \rho_\lambda^0 (0) U_R^\dagger, \\ \rho_\lambda^G(\vec{\theta}_G, z) &= U_G(\vec{\theta}_G) \rho_\lambda^0 (z) U_G^\dagger(\vec{\theta}_G). \end{aligned} \tag{7}$$

The unitary operation defining the discriminator $D(\vec{\theta}_D)$ has the form

$$U_D(\vec{\theta}_D) = D(\vec{\theta}_D) \otimes I^{\otimes m} \tag{8}$$

such that the state of the quantum computer when $U_D(\vec{\theta}_D)$ follows U_R is given by

$$\rho_\lambda^{DR}(\vec{\theta}_D) = U_D(\vec{\theta}_D) \rho_\lambda^R U_D^\dagger(\vec{\theta}_D) \tag{9}$$

and the state when $U_D(\vec{\theta}_D)$ is applied after $U_G(\vec{\theta}_G)$ is

$$\rho_\lambda^{DG}(\vec{\theta}_D, \vec{\theta}_G, z) = U_D(\vec{\theta}_D) \rho_\lambda^G(\vec{\theta}_G, z) U_D^\dagger(\vec{\theta}_D). \tag{10}$$

The cost function (3) can then be written in the quantum formalism as

$$\begin{aligned} V(\vec{\theta}_D, \vec{\theta}_G) = \frac{1}{2} + \frac{1}{2\Lambda} \sum_{\lambda=1}^{\Lambda} \Bigg(& \cos^2(\phi) \operatorname{tr}\left(Z \rho_\lambda^{DR}(\vec{\theta}_D) \right) \\ & - \sin^2(\phi) \operatorname{tr}\left(Z \rho_\lambda^{DG}(\vec{\theta}_D, \vec{\theta}_G, z) \right) \Bigg) \end{aligned} \tag{11}$$

where both parts depend on $\vec{\theta}_D$ and only the second part depends on $\vec{\theta}_G$, as in the classical case [4]. Here the angle ϕ parametrizes the bias of the coin used in Figure 3 since the probability that R or G is used as a source is not explicitly constrained in (3). Assuming a fair coin $\phi = \frac{\pi}{4}$, the quantum optimization problem has the final form

$$\min_{\vec{\theta}_G} \max_{\vec{\theta}_D} \frac{1}{2} + \frac{1}{4\Lambda} \sum_{\lambda=1}^{\Lambda} \operatorname{tr}\left(\left(\rho_\lambda^{DR}(\vec{\theta}_D) - \rho_\lambda^{DG}(\vec{\theta}_D, \vec{\theta}_G, z) \right) Z \right). \tag{12}$$

It is possible to train the circuit of Figure 2 using gradient descent methods [5]. Depending on whether $D(\vec{\theta}_D^k)$ or $G(\vec{\theta}_G^k)$ is being trained at a specific step k, the update rule of the parameters are given by

$$\begin{aligned} \vec{\theta}_D^{k+1} &= \vec{\theta}_D^k + \chi_D^k \nabla_{\vec{\theta}_D} V(\vec{\theta}_D^k, \vec{\theta}_G^k) \\ \vec{\theta}_G^{k+1} &= \vec{\theta}_G^k - \chi_G^k \nabla_{\vec{\theta}_G} V(\vec{\theta}_D^k, \vec{\theta}_G^k), \end{aligned} \tag{13}$$

where χ_D^k and χ_G^k are learning rates which can depend on k in general.

4. *Limit cases of the training*

The probability that $D(\vec{\theta}_D)$ successfully assigns the correct label to R and $G(\vec{\theta}_G)$ is given by the cost function $V(\vec{\theta}_D, \vec{\theta}_G)$. In what follows we will refer to this probability as $\Pr\left(\text{Success } D(\vec{\theta}_D) | \vec{\theta}_G\right)$. In the ideal case where $G(\vec{\theta}_G^*) = R$, G perfectly reproduces the statistics of the data source, D cannot distinguish [40, 41] between R and $G(\vec{\theta}_G^*)$, and $\Pr\left(\text{Success } D(\vec{\theta}_D) | \vec{\theta}_G\right) = \frac{1}{2}$. At this point the training is finished as D cannot improve its strategy and all gradients vanish:

$$\begin{aligned} \nabla_{\vec{\theta}_D} V(\vec{\theta}_D, \vec{\theta}_G^*) &= 0, \\ \nabla_{\vec{\theta}_G} V(\vec{\theta}_D, \vec{\theta}_G^*) &= 0. \end{aligned} \tag{14}$$

During the training, $\Pr\left(\text{Success } D(\vec{\theta}_D) | \vec{\theta}_G\right)$ is bounded by the purity function

$$C(\vec{\theta}_G) \equiv \operatorname{tr}\left(\rho^R \rho^G(\vec{\theta}_G) \right), \tag{15}$$

such that the performance of the discriminator is

$$\frac{1}{2} C(\vec{\theta}_G) \leq \Pr\left(\text{Success } D(\vec{\theta}_D) | \vec{\theta}_G\right) \leq 1 - \frac{1}{2} C(\vec{\theta}_G). \tag{16}$$

The purity function $C(\vec{\theta}_G)$ is itself bounded by the nature of R. If we define $r_{\min}$ as being the minimal eigenvalue of ρ^R, then

$$r_{\min} \leq C(\vec{\theta}_G) \leq \operatorname{tr}\left((\rho^R)^2 \right), \tag{17}$$

where the upper bound corresponds to the purity of ρ^R.

It is possible to train the circuit of Figure 2 by evaluating gradients from a numerical finite difference method. This requires sampling many points around each $(\vec{\theta}_D, \vec{\theta}_G)$ to estimate the gradient of (12). In the following section, we will show how gradients can be evaluated directly on a quantum computer and explicitly construct the circuits to optimize (12).

B. Quantum gradients

A key element of GANs is that the generator can be optimized by using gradient signals obtained from the discriminator. Thus, in addition to quantum circuits for G and D, we would also like to have quantum circuits which can compute the required gradients. Given access to these quantum gradients, model parameters can be updated via gradient descent on a classical computer. We introduce some notation useful to define gradient extraction on a quantum computer [27, 32, 33, 35, 42, 43]. In order to present a specific circuit setup, from here onwards we fix that the subsystems of our quantum computer are qubits. We also note that, in addition to the particular

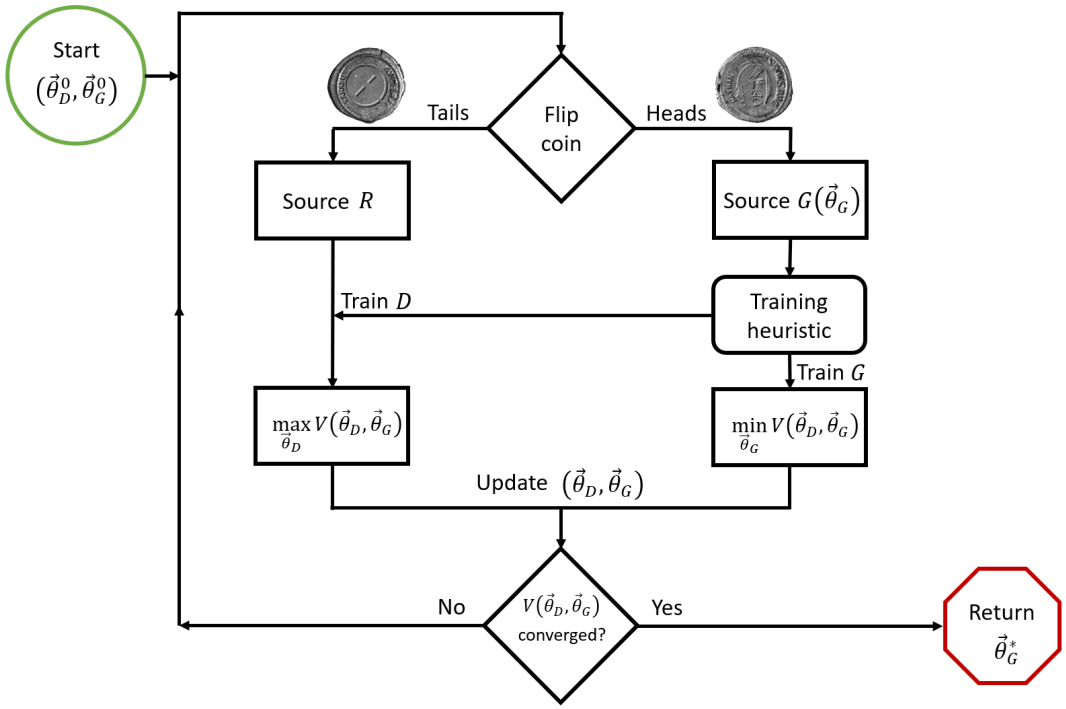

Figure 3. We illustrate the algorithmic flow of the training of a QuGAN (see text for details).

setup we use here, there can be other approaches for using a quantum computer to compute gradients of quantum circuits. A unitary transformation U parametrized by a vector $\vec{\theta}$ with N components is denoted

$$\begin{aligned} U(\vec{\theta}) &\equiv U_N(\theta_N) U_{N-1}(\theta_{N-1}) \dots U_2(\theta_2) U_1(\theta_1) \\ &= \mathcal{T} \prod_{j=1}^{N} U_j(\theta_j), \end{aligned} \tag{18}$$

where $\mathcal{T}$ is the time-ordering operator. It is convenient to introduce the ordered notation [44]

$$\begin{aligned} U_{k:l} &\equiv U_k(\theta_k) U_{k-1}(\theta_{k-1}) \dots U_{l+1}(\theta_{l+1}) U_l(\theta_l) \\ &= \mathcal{T} \prod_{j=l}^{k} U_j(\theta_j), \end{aligned} \tag{19}$$

which can also be represented in a quantum circuit notation as shown in Figure 4. In the same fashion, the anti-ordered notation has the form

$$\begin{aligned} U_{l:k}^\dagger &\equiv U_l^\dagger(\theta_l) U_{l+1}^\dagger(\theta_{l+1}) \dots U_{k-1}^\dagger(\theta_{k-1}) U_k^\dagger(\theta_k) \\ &= \bar{\mathcal{T}} \prod_{j=l}^{k} U_j^\dagger(\theta_j), \end{aligned} \tag{20}$$

where $\bar{\mathcal{T}}$ is the anti-time ordering operator. It follows that we can generally denote $U(\vec{\theta}) = U_{N:1}$ and $U^\dagger(\vec{\theta}) = U_{1:N}^\dagger$.

Assuming each element is generated by a Hamiltonian $h_j = h_j^\dagger$, an individual gate has the form

$$U_j(\theta_j) = e^{-\frac{i}{2}\theta_j h_j}, \tag{21}$$

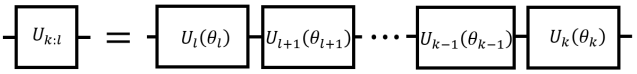

Figure 4. This notation is used to signify the decomposition of a unitary transformation in its elementary parametrized gates.

such that $U_j^\dagger(\theta_j) = e^{\frac{i}{2}\theta_j h_j}$. The derivative of gate j with respect to parameter θ_j is given by

$$\frac{\partial}{\partial \theta_j} U_j(\theta_j) = -\frac{i}{2} h_j U_j(\theta_j). \tag{22}$$

Using the chain rule, we find that

$$\begin{aligned} \tfrac{\partial}{\partial \theta_j} U(\vec{\theta}) &= -\tfrac{i}{2} U_{N:j+1} h_j U_{j:1} \\ \tfrac{\partial}{\partial \theta_j} U^\dagger(\vec{\theta}) &= \tfrac{i}{2} U_{1:j}^\dagger h_j U_{j+1:N}^\dagger. \end{aligned} \tag{23}$$

If we define an initial state on q qubits as ρ_0, the expectation value of an observable P evaluated for parameters $\vec{\theta}$ is given by

$$\left\langle P(\vec{\theta}) \right\rangle = \mathrm{tr}\left(\rho_0 U^\dagger(\vec{\theta}) P U(\vec{\theta}) \right). \tag{24}$$

The gradient with respect to a parameter θ_j is then given by

$$\frac{\partial}{\partial \theta_j} \left\langle P(\vec{\theta}) \right\rangle = -\frac{i}{2} \mathrm{tr}\left(\rho_0 U_{1:j}^\dagger \left[U_{j+1:N}^\dagger P U_{N:j+1}, h_j \right] U_{j:1} \right), \tag{25}$$

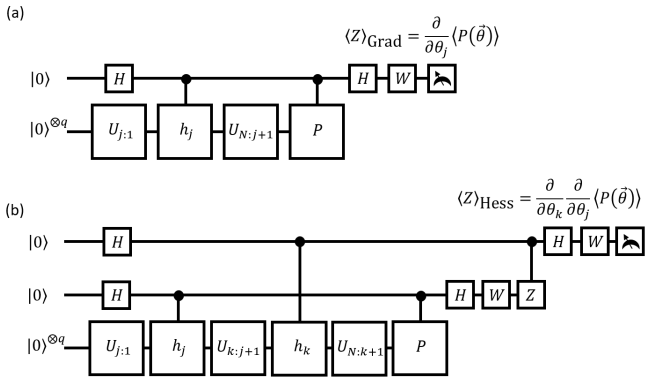

Figure 5. In (a), we show the general structure of quantum gradients and the structure of quantum Hessians is shown in (b).

where $[\cdot,\cdot]$ is the commutator.

At this point it is convenient to introduce some canonical quantum gates [16]. Specifically, the Hadamard gate is defined as $H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, the NOT gate as $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ and $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$. It is also useful to define the single-qubit W gate as

$$W \equiv e^{-i\frac{\pi}{4}X} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix}. \tag{26}$$

As shown in Figure 5 (a), the gradient of a parametrized quantum circuit can be sampled from the $\langle Z\rangle_{\text{Grad}}$ expectation value of an ancillary qubit such that

$$\begin{aligned} \langle Z\rangle_{\text{Grad}} &= \Pr\left(|x_{\text{Grad}}\rangle = |0\rangle\right) - \Pr\left(|x_{\text{Grad}}\rangle = |1\rangle\right) \\ &= \tfrac{\partial}{\partial\theta_j}\left\langle P(\vec{\theta})\right\rangle. \end{aligned} \tag{27}$$

Note that this requires the ability to perform control gates for the Hamiltonians h_j and measurement operator P. Similarly, using the fact that the Hessian is the gradient of a gradient, we show how the Hessian can be measured in Figure 5 (b), such that the output is

$$\langle Z\rangle_{\text{Hess}} = \frac{\partial^2}{\partial\theta_k\partial\theta_j}\left\langle P(\vec{\theta})\right\rangle. \tag{28}$$

C. Using quantum gradients to train QuGANs

We now have all the elements required to evaluate the gradients of (13) directly on a quantum computer. The operator P from Section II B corresponds to the Z operator of (4) when computing gradients. The parametrized discriminator D and generator G can be decomposed into respectively N_D and N_G gates, such that

$$\begin{aligned} D(\vec{\theta}_D) &= D_{N_D:1}, \\ G(\vec{\theta}_G) &= G_{N_G:1}. \end{aligned} \tag{29}$$

In order to measure gradients, we introduce a single-qubit register **Grad**. It follows that all elements $\frac{\partial}{\partial\theta_{Dj}}V(\vec{\theta}_D,\vec{\theta}_G) = \frac{1}{4\Lambda}\langle Z\rangle_{\text{Grad}}$ of the gradient of the discriminator

$$\begin{aligned} \frac{\partial}{\partial\theta_{Dj}}V(\vec{\theta}_D,\vec{\theta}_G) = -\frac{i}{8\Lambda}\sum_{\lambda=1}^{\Lambda}\text{tr}\Bigg(\left(\rho_\lambda^R - \rho_\lambda^G(\vec{\theta}_G,z)\right) \\ \times U_{D,1:j}^\dagger\left[U_{D,j+1:N_D}^\dagger Z U_{D,N_D:j+1}, h_j^D\right]U_{D,j:1}\Bigg), \end{aligned} \tag{30}$$

can be evaluated for each label and sources R and $G(\vec{\theta}_G)$ by the quantum circuit of Figure 6 (a) with an appropriate X gate to account for the sign of the cost function. In the later case, an X gate is applied on the **Out D** register after the discriminator to get the correct sign of the gradient. The circuit that yields the gradient

$$\begin{aligned} \frac{\partial}{\partial\theta_{Gj}}V(\vec{\theta}_D,\vec{\theta}_G) = \frac{i}{8\Lambda}\sum_{\lambda=1}^{\Lambda}\text{tr}\Bigg(\rho_\lambda^0(z) \\ \times U_{G,1:j}^\dagger\left[U_{G,j+1:N_G}^\dagger U_D^\dagger(\vec{\theta}_D) Z U_D(\vec{\theta}_D) U_{G,N_G:j+1}, h_j^G\right]U_{G,j:1}\Bigg) \end{aligned} \tag{31}$$

of the generator $-\frac{\partial}{\partial\theta_{Gj}}V(\vec{\theta}_D,\vec{\theta}_G) = \frac{1}{4\Lambda}\langle Z\rangle_{\text{Grad}}$ for each label is shown in Figure 6 (b). We note that the sign is meant to be the same as the one in (13), such that the generator improves its capability to fool the discriminator. More advanced methods to update the parameters could also leverage the use of quantum Hessians (28).

1. *Improved training heuristics*

Training GANs is equivalent to finding the Nash equilibrium of a two-player game. This problem is known to be in the complexity class PPAD which is not expected to be contained in BQP [45, 46]. Advanced heuristics have been developed to improve the training of classical GANs [5]. Namely, it should be straightforward to implement semi-supervised learning in the quantum context by increasing the number of labels to $\Lambda + 1$ and supplying some labeled examples of generated data. Feature matching should also be possible by truncating the decomposition of $D(\vec{\theta}_D)$ when evaluating the gradients of $G(\vec{\theta}_G)$ with the circuit of Figure 6 (b). We also assumed that the expectation value of each gradient is evaluated from ensemble averaging; it may also be possible to use Bayesian methods to update the parameters after single-shot measurements [47].

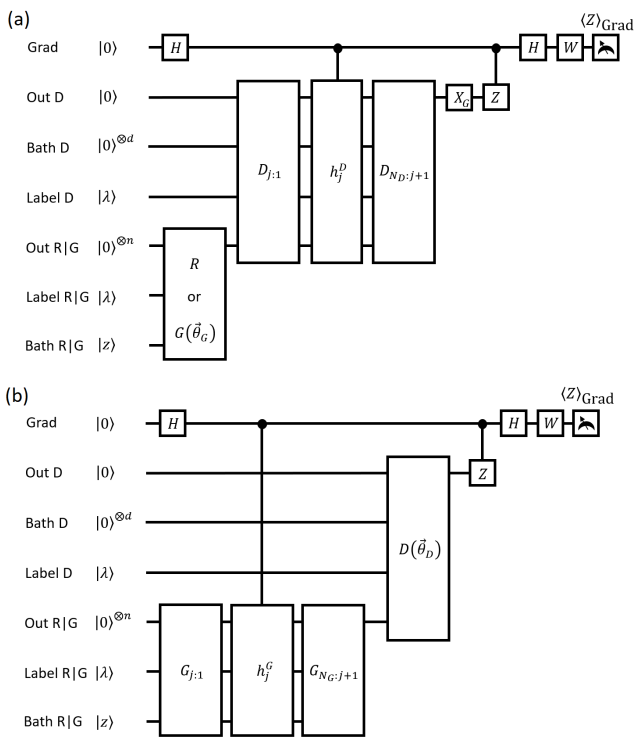

Figure 6. In (a), we show the quantum circuit used to measure gradient of the discriminator with real data (using $X_G = I$ and fixing $|z = 0\rangle$) and with generated data (using $X_G = X$). The circuit to measure the gradient of the generator is shown in (b).

D. A practical ansatz

A potentially useful ansatz to parametrize $D(\vec{\theta}_D)$ and $G(\vec{\theta}_G)$ is shown in Figure 7. It is universal for quantum computing in the limit of an infinite number of layers τ. Since the generators of those gates are all simple Pauli operators, it is easy to implement the conditional h_j's with CNOTs, CPHASEs and CZZs where the ZZs are between nearest-neighbor qubits. Other types of ansatz may be used depending on the context [19, 20, 23, 33, 34].

E. Numerics

We numerically tested ideas in this paper with a simple example involving two labels A and B. We chose a source R such that $\rho_A^R = |0\rangle\langle 0|$ and $\rho_B^R = |1\rangle\langle 1|$. The labels can be encoded in a 1-qubit **Label R|G** register and the **Out R|G** register only requires 1 qubit. Since the labeled distributions ρ_λ^R are pure we don't need a **Bath R|G** register to generate entropy. The expected solution is that G should be able to generate a CNOT gate conditioned on the label register. We find that this can be achieved with 2 layers of the ansatz previously introduced. This corresponds to 10 variational parameters in $\vec{\theta}_G$.

The discriminator requires at least 1 qubit for its output **Out D**, 1 qubit for **Label D**, and it also operates on the 1-qubit **Out R|G** register. We find that a **Bath D** register did not appear to improve convergence of our numerical experiments. Therefore, D operates on 3 qubits, and we found that 4 layers of the ansatz of section II D were sufficient to train the QuGAN. This yields 32 parameters in $\vec{\theta}_D$ for a total of 42. With the qubit of register **Grad**, the algorithm operates on a total of 5 qubits.

Training GANs is a delicate art. To keep this proof-of-principle simple we chose not to use any advanced training heuristic. We trained the QuGAN for 10,000 gradient steps of the update rule (13). The learning rate χ_D^k exponentially decreases from 10 to $\frac{1}{10}$ for the first 4,000 steps and remains constant at the latter value for the remaining 6,000 steps. The generator G is only updated once for every 100 steps of D with a learning rate $\chi_G^k = 5\chi_D^k$.

As shown in Figure 8, the generator has been properly trained at the end of the algorithm, as the cross-entropy

$$S\left(\rho_\lambda^R \| \rho_\lambda^G\right) = \mathrm{tr}\left(\rho_\lambda^R \left(\log_2 \rho_\lambda^R - \log_2 \rho_\lambda^G\right)\right) \qquad (32)$$

quickly converges to zero. We also plotted the components of the cost function V defined as

$$\begin{aligned} V^{DR}(\vec{\theta}_D) &= \tfrac{1}{4\Lambda}\textstyle\sum_{\lambda=1}^{\Lambda} \mathrm{tr}\left(\rho_\lambda^{DR}(\vec{\theta}_D) Z\right) \\ V^{DG}(\vec{\theta}_D, \vec{\theta}_G) &= -\tfrac{1}{4\Lambda}\textstyle\sum_{\lambda=1}^{\Lambda} \mathrm{tr}\left(\rho_\lambda^{DG}(\vec{\theta}_D, \vec{\theta}_G, z) Z\right), \end{aligned} \qquad (33)$$

such that $V = \frac{1}{2} + V^{DR} + V^{DG}$. At the beginning of the training sequence, the parameters are chosen randomly. In order to provide a reliable training signal for G, the gradients are amplified by a large learning rate to quickly train D. The generator initially produces a decent state for the A label but fails to produce a good state for the B label. The training of the discriminator appears successful since V is typically larger than $\frac{1}{2}$ and learns to differentiate the data produced by R from the data produced by G. Updating the generator less often than the discriminator provides a trade-off between a fast training of G and a good training signal. After a few tens of training steps of G (which corresponds to a few thousand training cycles of D) the cross-entropy between the real and the generated data starts to converge to zero as the generator creates better samples. In this case, D cannot differentiate R and G as V approaches its equilibrium value of $\frac{1}{2}$. The final strategy of D is to designate all data as real, yielding $V^{DR} \approx \frac{1}{2}$ and $V^{DG} \approx -\frac{1}{2}$.

III. CONCLUSION

Quantum machine learning is likely to be one of the first general-purpose applications of near-term quantum devices. Here we showed how generative models can be

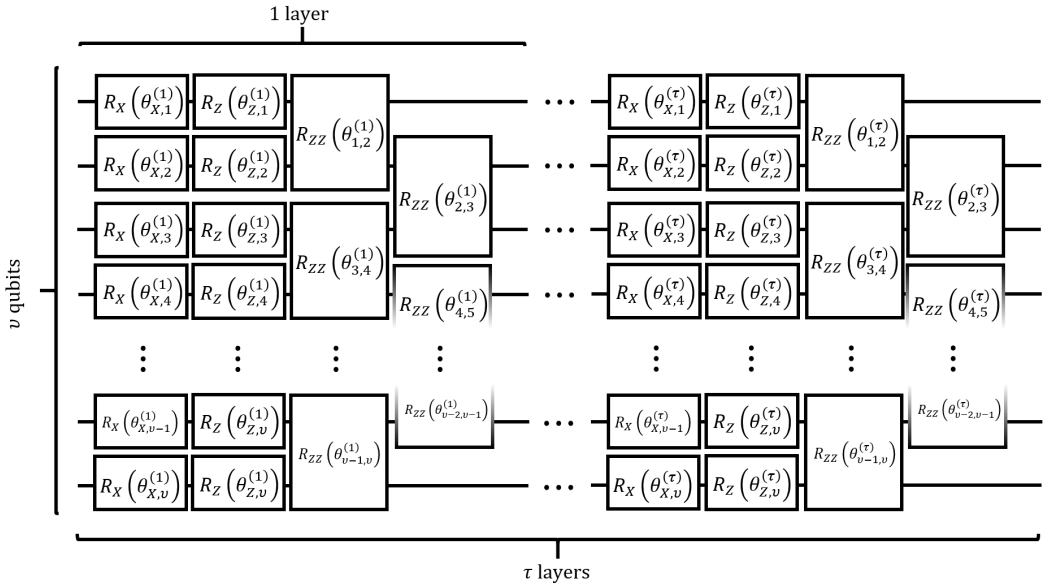

Figure 7. A practical circuit ansatz for the generator G and the discriminator D composed of τ layers acting on ν qubits. Each layer t is composed of single-qubit X rotations parametrized by angles $\vec{\theta}_X^{(t)} = \left\{\theta_{X,1}^{(t)}, \ldots, \theta_{X,\nu}^{(t)}\right\}$ followed by Z rotations parametrized by $\vec{\theta}_Z^{(t)} = \left\{\theta_{Z,1}^{(t)}, \ldots, \theta_{Z,\nu}^{(t)}\right\}$. A layer of two staggered sets of nearest-neighbor ZZ rotations parametrized by $\vec{\theta}_{ZZ}^{(t)} = \left\{\theta_{1,2}^{(t)}, \ldots, \theta_{\nu-1,\nu}^{(t)}\right\}$ follows the single-qubit rotations. The ansatz is universal for quantum computing in the limit of an infinite number of layers since it can generate arbitrary single-qubit gates as well as entangling two-qubit gates.

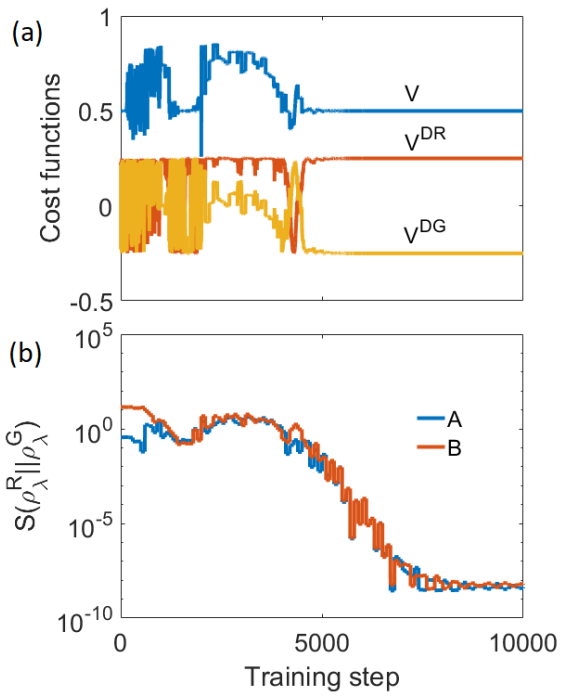

Figure 8. A source R produces two labeled states $|A, 0\rangle$ and $|B, 1\rangle$. In (a) we have the values of the cost functions as a function of the training step. In (b) we show the cross-entropy $S\left(\rho_\lambda^R \| \rho_\lambda^G\right)$ for each labeled distribution as a function of the training step.

trained on quantum computers. We have reformulated the optimization problem of GANs in the quantum formalism, yielding QuGANs. We have shown how the cost function can be optimized by directly evaluating the gradients with a quantum processor. We provided a simple universal qubit ansatz which constrains the set of additional quantum resources required to evaluate the gradients. Finally, we showed that QuGANs can be trained in practice by performing a simple numerical experiment.

It is expected that QuGANs will have a more versatile representation power than their classical counterpart. For example, one can speculate that a large enough QuGAN could learn to generate encrypted data labeled by RSA public encryption keys since quantum computers have the capacity to perform Shor factoring [17] and hence decryption. In that case, the optimal generator would learn a statistical model of the unencrypted data for each key and encrypt with the label. Other classical cryptographic systems (such as elliptic curve) could also be vulnerable to this type of attack. In this work, we have explored the practical issues of QuGANs, namely, explicit quantum circuits for the generator and discriminator, as well as quantum methods for computing the gradients of these circuits. A more general analysis of the theoretical concepts of quantum adversarial learning can be found in the companion paper [37].

ACKNOWLEDGMENTS

We thank Seth Lloyd and Christian Weedbrook for their insightful advices. This works was made possible by ample supplies of Tim Horton's coffee.

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

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning (Adaptive Computation and Machine Learning series)* (The MIT Press, 2016).

[3] L. Deng, Foundations and Trends in Signal Processing **7**, 197 (2014).

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, in *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2* (MIT Press, 2014) pp. 2672–2680.

[5] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, ArXiv e-prints (2016), arXiv:1606.03498 [cs.LG].

[6] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, IEEE Signal Processing Magazine **35**, 53 (2018).

[7] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, in *European Conference on Computer Vision* (Springer, 2016) pp. 597–613.

[8] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, ArXiv e-prints (2016), arXiv:1609.04802 [cs.CV].

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, ArXiv e-prints (2016), arXiv:1611.07004 [cs.CV].

[10] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, in *European Conference on Computer Vision* (Springer, 2016) pp. 628–644.

[11] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, in *Advances in Neural Information Processing Systems* (2017) pp. 5769–5779.

[12] A. Kadurin, S. Nikolenko, K. Khrabrov, A. Aliper, and A. Zhavoronkov, Molecular pharmaceutics **14**, 3098 (2017).

[13] N. Killoran, L. J. Lee, A. Delong, D. Duvenaud, and B. J. Frey, ArXiv e-prints (2017), arXiv:1712.06148 [cs.LG].

[14] L. de Oliveira, M. Paganini, and B. Nachman, Computing and Software for Big Science **1**, 4 (2017).

[15] R. P. Feynman, International Journal of Theoretical Physics **21**, 467 (1982).

[16] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2009).

[17] P. W. Shor, SIAM Journal on Computing **26**, 1484 (1997).

[18] J. Preskill, ArXiv e-prints (2018), arXiv:1801.00862 [quant-ph].

[19] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, Nature Communications **5**, 4213 (2014).

[20] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Nature **549**, 242 (2017).

[21] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, A. Kandala, A. Mezzacapo, P. Müller, W. Riess, G. Salis, J. Smolin, I. Tavernelli, and K. Temme, ArXiv e-prints (2017), arXiv:1710.01022 [quant-ph].

[22] G. Giacomo Guerreschi and M. Smelyanskiy, ArXiv e-prints (2017), arXiv:1701.01450 [quant-ph].

[23] P.-L. Dallaire-Demers, J. Romero, L. Veis, S. Sim, and A. Aspuru-Guzik, ArXiv e-prints (2018), arXiv:1801.01053 [quant-ph].

[24] M. Schuld, I. Sinayskiy, and F. Petruccione, Contemporary Physics **56**, 172 (2014).

[25] M. Arjovsky, A. Shah, and Y. Bengio, ArXiv e-prints (2015), arXiv:1511.06464 [cs.LG].

[26] J. Romero, J. P. Olson, and A. Aspuru-Guzik, Quantum Science and Technology **2**, 045001 (2017).

[27] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. Love, and A. Aspuru-Guzik, ArXiv e-prints (2017), arXiv:1701.02691 [quant-ph].

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

[29] Y. Cao, G. Giacomo Guerreschi, and A. Aspuru-Guzik, ArXiv e-prints (2017), arXiv:1711.11240 [quant-ph].

[30] G. Verdon, M. Broughton, and J. Biamonte, ArXiv e-prints (2017), arXiv:1712.05304 [quant-ph].

[31] J. S. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell, N. Didier, E. Schuyler Fried, S. Hong, P. Karalekas, C. B. Osborn, A. Papageorge, E. C. Peterson, G. Prawiroatmodjo, N. Rubin, C. A. Ryan, D. Scarabelli, M. Scheer, E. A. Sete, P. Sivarajah, R. S. Smith, A. Staley, N. Tezak, W. J. Zeng, A. Hudson, B. R. Johnson, M. Reagor, M. P. da Silva, and C. Rigetti, ArXiv e-prints (2017), arXiv:1712.05771 [quant-ph].

[32] M. Schuld and N. Killoran, ArXiv e-prints (2018), arXiv:1803.07128 [quant-ph].

[33] M. Schuld, A. Bocharov, K. Svore, and N. Wiebe, ArXiv e-prints (2018), arXiv:1804.00633 [quant-ph].

[34] W. Huggins, P. Patel, K. B. Whaley, and E. Miles Stoudenmire, ArXiv e-prints (2018), arXiv:1803.11537 [quant-ph].

[35] E. Farhi and H. Neven, ArXiv e-prints (2018), arXiv:1802.06002 [quant-ph].

[36] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, ArXiv e-prints (2018), arXiv:1803.00745 [quant-ph].

[37] S. Lloyd and C. Weedbrook, ArXiv e-prints (2018), arXiv:1804.09139 [quant-ph].

[38] M. Mirza and S. Osindero, ArXiv e-prints (2014), arXiv:1411.1784 [cs.LG].

[39] I. D. Kivlichan, J. McClean, N. Wiebe, C. Gidney, A. Aspuru-Guzik, G. K.-L. Chan, and R. Babbush, Physical Review Letters **120**, 110501 (2018).

[40] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf, Physical Review Letters **87**, 167902 (2001).
[41] S. Aaronson, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **463**, 3089 (2007).
[42] E. Knill and R. Laflamme, Physical Review Letters **81**, 5672 (1998).
[43] G. Ortiz, J. Gubernatis, E. Knill, and R. Laflamme, Computer Physics Communications **146**, 302 (2002).
[44] S. Machnes, U. Sander, S. J. Glaser, P. de Fouquières, A. Gruslys, S. Schirmer, and T. Schulte-Herbrüggen, Physical Review A **84**, 022305 (2011).
[45] P. V. Fellman and J. V. Post, in *Unifying Themes in Complex Systems* (Springer Berlin Heidelberg, 2010) pp. 454–461.
[46] Y. D. Li, ArXiv e-prints (2011), arXiv:1108.0223 [cs.CC].
[47] M. P. V. Stenberg and F. K. Wilhelm, Physical Review A **94**, 052119 (2016).