electronics

MDPI

Review

A Survey of Recent Advances in Quantum Generative Adversarial Networks

Tuan A. Ngo, Tuyen Nguyen and Truong Cong Thang *

Department of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu 965-8580, Japan
* Correspondence: thang@u-aizu.ac.jp

Abstract: Quantum mechanics studies nature and its behavior at the scale of atoms and subatomic particles. By applying quantum mechanics, a lot of problems can be solved in a more convenient way thanks to its special quantum properties, such as superposition and entanglement. In the current noisy intermediate-scale quantum era, quantum mechanics finds its use in various fields of life. Following this trend, researchers seek to augment machine learning in a quantum way. The generative adversarial network (GAN), an important machine learning invention that excellently solves generative tasks, has also been extended with quantum versions. Since the first publication of a quantum GAN (QuGAN) in 2018, many QuGAN proposals have been suggested. A QuGAN may have a fully quantum or a hybrid quantum–classical architecture, which may need additional data processing in the quantum–classical interface. Similarly to classical GANs, QuGANs are trained using a loss function in the form of max likelihood, Wasserstein distance, or total variation. The gradients of the loss function can be calculated by applying the parameter-shift method or a linear combination of unitaries in order to update the parameters of the networks. In this paper, we review recent advances in quantum GANs. We discuss the structures, optimization, and network evaluation strategies of QuGANs. Different variants of quantum GANs are presented in detail.

Keywords: quantum machine learning; generative adversarial networks; quantum GAN; hybrid quantum–classical system

Citation: Ngo, T.A.; Nguyen, T.; Thang, T.C. A Survey of Recent Advances in Quantum Generative Adversarial Networks. *Electronics* **2023**, *12*, 856. https://doi.org/10.3390/electronics12040856

Academic Editors: Juan M. Corchado, Byung-Gyu Kim, Carlos A. Iglesias, In Lee and Rashid Mehmood

Received: 16 January 2023
Revised: 3 February 2023
Accepted: 5 February 2023
Published: 8 February 2023

1. Introduction

Machine learning (ML) [1] has been a hot topic for researchers for a long time. Early ML works, such as artificial neurons [2], the perceptron [3], support-vector machines [4], recurrent neural networks [5], and convolutional neural networks [6] were developed and applied in all parts of the human society, such as education [7–9], agriculture [10,11], finance [12,13], and health care [14–16]. One of the important goals of ML is generative tasks, where ML programs have to generate new data based on some existing data. Through excellent empirical results, generative adversarial networks (GANs) [17] have been found a brilliant candidate to fulfill this task. In a GAN, a generator and a discriminator play an adversarial game against each other. The generator tries to generate new data that resemble some real data in order to fool the discriminator, while the discriminator aims to distinguish the generated data from the real data.

There has been a sharp increase in the number of GAN variants in the past few years [18]. Since their invention, GANs have been widely used in both semi-supervised and unsupervised learning. However, this type of network bears some issues, such as training instability, mode collapse, and non-convergence [19]. Researchers have been trying their best to improve the efficiencies of the networks, to overcome their shortcomings, and to expand their applications.

In recent years, quantum computing [20] has emerged and drawn a lot of attention from researchers. This new paradigm of computing can accomplish difficult tasks that traditional computing cannot solve. With the development of near-term quantum devices, quantum computing can make use of special properties, including superposition

and entanglement, to even perform those tasks exponentially faster [21]. For example, quantum computers can search an unsorted dataset of N entries in time $O(\sqrt{N})$, whereas it takes time $O(N)$ for classical computers to do the same task [22]. Thus, researchers seek to combine quantum computing with machine learning to develop quantum ML. The quantum counterparts of ML models, such as quantum reinforcement learning [23], quantum support vector machines [24], and quantum variational autoencoders [25], were in turn invented.

The generative tasks of ML are getting more and more complicated. Therefore, quantum GANs (QuGANs) were also developed. Basically, QuGANs may have different architectures with various components, in either classical or quantum forms. Instead of neural networks, the quantum parts of a QuGAN consist of variational quantum circuits which also depend on a set of parameters [26]. A certain loss or cost function can be formed, and its gradients with respect to the parameters are calculated for updating the parameters themselves. When an optimal status is reached, the network is able to generate a specific type of data that mimics the true data. The performance of the network can be evaluated using some metrics measuring the distances between these generated data and the real data [27–30].

To the best of our knowledge, there has been only one survey about QuGAN [31], which generally discusses the structures, the loss functions, the applications, and the challenges of GANs and QuGANs. Recently, there have been many new QuGAN methods proposing new structures, objective functions, gradient calculation, optimization, and evaluation strategies. In this paper, we deeply analyze quantum GANs in a systematic way. Variants of QuGAN are discussed in detail.

The structure of the survey is as follows. Firstly, we review some background related to quantum GANs in Section 2. The structures of quantum GANs are discussed in Section 3. Section 4 describes some optimization techniques of QuGAN, including loss function choices, gradient calculation, network optimization, and evaluation strategies. Quantum GAN variants are presented in detail in Section 5. Finally, the paper is concluded in Section 6.

2. Background

2.1. Generative Adversarial Networks (GANs)

The idea of quantum GANs originates from classical GANs, which were first proposed by Goodfellow et al. [17]. A GAN aims at learning a target distribution, which can be a series of text or a collection of images or audio, and generating samples that have similar characteristics to the samples in the training distribution. The architecture of a typical GAN is depicted in Figure 1. A GAN typically consists of a generator G and a discriminator D. The generator and the discriminator are made from neural networks and are parameterized by θ_G and θ_D, respectively. The mission of G is to take noise vectors z from a noise source and produce data x' that mimic the realistic data x as much as possible in order to fool D, which is supposed to distinguish whether a sample is taken from the real distribution or is generated by G. The distinguishing results y of D are then used to train both G and D iteratively; in other words, θ_G and θ_D are updated alternately until the network reaches its optimality. Ideally, this optimality, or Nash equilibrium [17], occurs when G can generate identical samples to those in the real distribution, and D is not able to determine which source the data belong to.

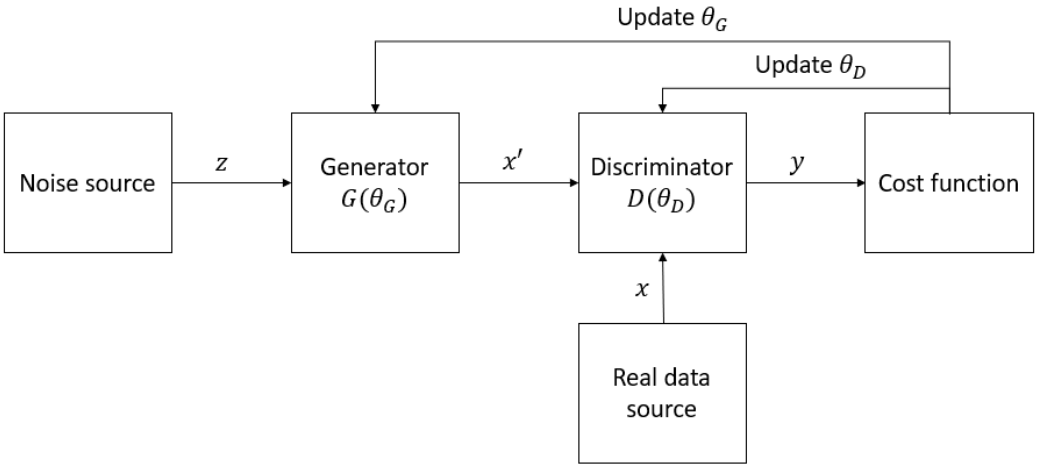


Figure 1. The architecture of a typical GAN.

GANs are the most potential and popular types of networks used for generative tasks. They vary in terms of network architectures, optimization strategies, and purposes of use. Some notable variants of GANs include deep convolutional GAN [32], conditional GAN [33], Wasserstein GAN [34], boundary equilibrium GAN [35], Big GAN [36], Laplacian GAN [37], and information maximizing GAN [38]. Many GANs have been the basis of outstanding achievements. For example, StyleGAN3 [39] obtained a Fréchet inception distance of 3.07 on the FFHQ dataset, 4.40 on the AFHQv2 dataset, and 4.57 on the Beaches dataset. These networks have found their roles in image domains such as image synthesis [36,37,40,41], image inpainting [42,43], image blending [44,45], image superresolution [46,47], and image-to-image translation [48–50]; audio domains such as speech synthesis [51,52] and music composing [53,54]; and other fields such as autonomous driving [55], weather forecasting [56,57], and data augmentation [58,59]. GANs' effectiveness and varied applications have been described in previous surveys of them [18,19,60].

Consider two probability measures P and Q of a random variable x defined on a measurable space $(\Omega, \mathcal{F})$. Denote $E[\cdot]$ the expected value of $[\cdot]$. Typical ways to measure the distance between these two distributions are as follows.

- Total variation (TV) [61]:

$$TV(P, Q) = \sup_{A \in \mathcal{F}} \|P(A) - Q(A)\|. \tag{1}$$

- Kullback–Leibler divergence (KLD) [62]:

$$KLD(P\|Q) = E_{P(x)}\left[\log \frac{P(x)}{Q(x)}\right]. \tag{2}$$

- Jensen–Shannon divergence (JSD) [63]:

$$JSD(P\|Q) = \frac{1}{2}KLD(P\|M) + \frac{1}{2}KLD(Q\|M), \tag{3}$$

  where $M = \frac{P+Q}{2}$.

- Wasserstein distance (WD) [34]:

$$W(P, Q) = \inf_{\gamma \in \prod(P,Q)} E_{(x,y)\sim\gamma}[\|x - y\|], \tag{4}$$

where $\prod(P, Q)$ is the set of all joint distributions $\gamma(x; y)$ whose marginals are P and Q. However, normally, the infimum is difficult to keep track of, so according to Kantorovich–Rubinstein duality, the formula can be written as

$$W(P, Q) = \sup_{\|f\|_L \leq 1} E_{x \sim P}[f(x)] - E_{x \sim Q}[f(x)]. \tag{5}$$

The problem with some of these distances is that they may not be continuous in some cases. When the two distributions have supports lying on low-dimensional manifolds, their TV, JSD, and KLD will be discontinuous at some points [34]. The WD metric is defined by the minimum cost of transporting mass to transform from one distribution to another. When applied in a GAN, the discriminator becomes a critic, which aims to estimate a function $f(x)$ such that substituting that function into the WD formula will produce the exact or approximate WD of the real and the generated distributions. The Lipschitz constraint, $\|f\|_L \leq 1$, can be satisfied by clipping the weights of the critic [34], penalizing the model if the gradient norm moves away from one [64,65], or adding a regularization term to the formula to make sure the norm of the function is less than one [66].

Although classical GANs have shown their effectiveness in various fields, they still face drawbacks that prevent them from tackling all generative problems. Sometimes, they cost a lot of computational resources and have rather long runtimes [36]. They also have difficulties in generating discrete data because it is not easy to represent the gradient update of a discrete value to the parameters of the network [17]. In addition, training a classical GAN usually encounters unstable convergence [67,68], vanishing gradients, or mode collapse [69].

2.2. Quantum–Classical Interface

As quantum machines only work with data stored in quantum states, classical data must be encoded into this form of data. Therefore, data encoding (or embedding) has also become an interesting field of research. There are various data encoding techniques that are used in quantum machine learning algorithms, but in quantum GANs, basis encoding, amplitude encoding, and angle encoding are the most popular [70].

Basis encoding is the simplest way to embed classical data into a quantum state. The inputs must be in binary form, and each of them corresponds with a computational basis of the qubit system. In other words, an n-bit binary string $x = \{0, 1\}^n$ will be encoded into a quantum state $|x\rangle$, and exactly n qubits are required to contain the information of x. Thanks to the superposition property of quantum computing, one can encode multiple inputs into a single quantum state. Suppose a collection of inputs D consists of M binary strings x^m of length n. Then, all inputs in D can be embedded in the quantum state:

$$|D\rangle = \frac{1}{\sqrt{M}} \sum_{m=1}^{M} |x^m\rangle . \tag{6}$$

Another data-encoding method is amplitude encoding, which is applied in [29,71]. This encoding method can help embed classical N-dimensional data points into a quantum state. Each dimension of the data point becomes the amplitude of a corresponding computational basis state. More specifically, a data point $x = (x_k)$ for $k = 1, \ldots, N$ is encoded as

$$|x\rangle = \frac{1}{\sqrt{\sum_{k=1}^{N} x_k^2}} \sum_{k=1}^{N} x_k |i_k\rangle , \tag{7}$$

where $|i_k\rangle$ is the k-th computational basis state. To encode an input with N dimensions, at least $\log_2(N)$ qubits are needed. M inputs with N dimensions can be encoded together by concatenating them to form a new data point with MN dimensions.

Sometimes, for simplicity, angle encoding is applied. Here, each dimension of a data point $x = (x_k)$ for $k = 1, \ldots, N$ becomes the angle of a rotation gate $R(\cdot)$. The encoded quantum state is

$$|x\rangle = \bigotimes_{k=1}^{N} R(x_k) |0\rangle . \tag{8}$$

This method typically requires N qubits for N dimensions of x. For example, in Ref. [28], Stein et al. normalized the range of each dimension of the dataset between 0 and 1, and then transformed a data point x into an angle $\theta = 2 \arcsin \sqrt{x}$, which would then go through R_y gates. This lets the quantum data return to the classical form after being generated. In [72,73], various encoding types for classical data were evaluated. It should be noted that for high-dimensional data, different ways of scanning data at the quantum–classical interface may also impact the performance of quantum machine learning [74].

When the data are generated by a quantum generator, to let the traditional discriminator interpret and distinguish between the real and the fake ones, or simply to enable us to visualize them, they must be converted into a classical form. This can be done by carrying out some quantum measurements using positive operator-valued measurement [75]. Suppose each outcome m that can occur after the measurement corresponds with a measurement operator M_m, and the state right before the measurement is $|\psi\rangle$. Then, the probability that m occurs is $p(m) = \langle \psi | M_m^\dagger M_m | \psi \rangle$. After the measurement, the quantum system immediately changes its state to $\frac{M_m |\psi\rangle}{\sqrt{\langle \psi | M_m^\dagger M_m | \psi \rangle}}$ and can be no more manipulated. Therefore, in practice, the quantum circuits are executed multiple times (or shots), and the probability of each possible outcome will be estimated based on its occurrences.

3. Structures of QuGAN

A QuGAN can be fully quantum, hybrid, or based on tensor networks. A diagram of quantum GAN categorization in terms of network architecture is sketched in Figure 2.

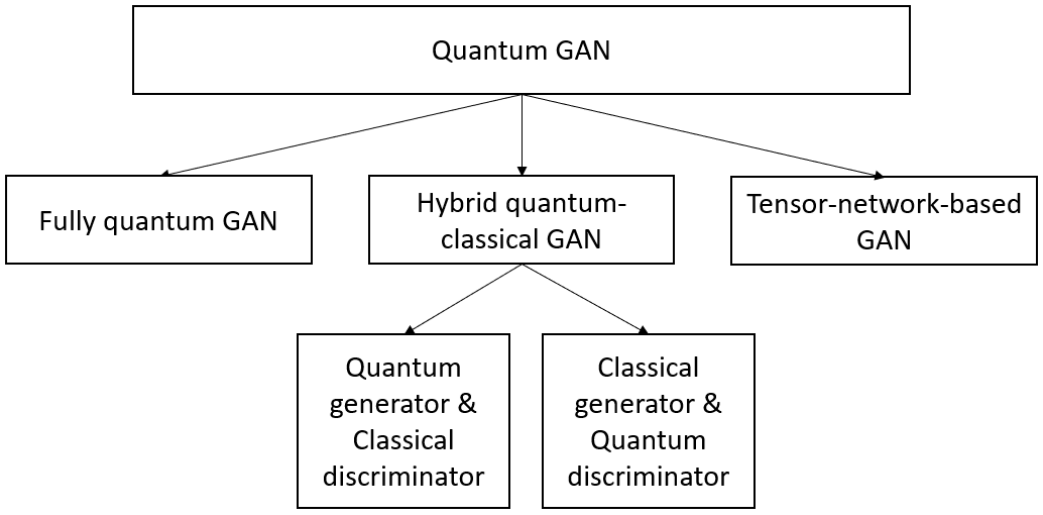


Figure 2. Quantum GAN structures.

In the fully quantum (or quantum-quantum) case, both the generator and the discriminator are quantum. Suppose the quantum generator produces fake data with a density matrix ρ, and the quantum discriminator must discriminate those fake data with true data described by a density matrix σ with a measurement operator D. The outcomes of D can be T (i.e., true, or the data are from the real data source), or F (i.e., false, or the data are generated by the generator). Since T and F are positive operators with a 1-norm less than or equal to 1, the set of them is convex, which means there must exist a minimum error measurement [26]. To reach this optimal measurement, the discriminator aims to maximize

the probability that it correctly categorizes the data as real or fake by following the gradients of the probability with respect to its parameters to adjust its weights. After that, due to the fact that the set of ρ is convex, the generator also manages to adjust its own weights to maximize the probability that the discriminator fails to discriminate the data and produce an optimal density matrix ρ [26]. Similarly to the traditional GANs, after a number of iterations of adjusting the weights of both the generator and the discriminator, a quantum GAN can also approach the Nash equilibrium. Some typical fully quantum GANs include the networks proposed by Dallaire-Demers et al. [27], Huang et al. [29], and Stein et al. [28].

The mechanism is similar when only a part of the classical GAN is replaced by a quantum engine. However, not all hybrid structures are possible. In particular, the generator or the discriminator cannot be classical if the training dataset is generated by a quantum system, which has quantum supremacy. This means the classical generator can never generate the statistics of a dataset that are similar to those of a quantum data source [20]. Therefore, using the same strategy as in a fully quantum network, the quantum discriminator can always find a measurement to distinguish the true and the generated data. As a result, the probability that the discriminator makes wrong predictions will always be less than $\frac{1}{2}$, and the Nash equilibrium will never happen [26]. On the contrary, in the case where the discriminator is classical, both the real data and the generated data which are fed into the discriminator are quantum.

When the target data are classical, either the generator or the discriminator can be classical. However, although possessing quantum supremacy, the quantum systems only act on data as quantum states and are unable to work with classical data directly. If the discriminator is quantum, both the training data and the fake data generated by the classical generator need to be encoded before being fed into the discriminator. On the contrary, if the generator is quantum, the generated data must be measured using some computational bases to produce classical generated samples [29,30].

Recently, Huggins et.al. [76] introduced a new genre of QuGAN architectures based on tensor networks (TN). It is based on a high-rank tensor that describes the quantum wavefunction of a multi-partite system as a factorization over low-rank tensors. This design offers a robust and reliable foundation for developing quantum-assisted machine learning models. Differently from the fully quantum or hybrid architectures mentioned above, the models based on tensor networks provide a unified framework in which quantum computing and classical computing can benefit from the same technical developments. This means the same models are trained in a classical environment and then can be transferred to a quantum environment for further optimization without any modification. Therefore, this design is suitable for both classical and quantum data.

An important issue in GANs is latent space [77]. As for QuGANs, all proposals up to this time use a quantum generator, which requires the latent space to be a collection of quantum states. The preferred method to prepare the latent space is randomly drawing from a certain distribution. For example, in [30], the authors tested the network in the cases where the latent quantum state is sampled from a uniform, a normal, or a random distribution. There has been only one work where a quantum circuit-based generative model was used to learn and generate the latent space of the generator in a classical GAN [78].

4. Optimization of QuGAN

4.1. Loss Function

Just like the traditional GANs, the quantum version needs a function whose gradients it can trace along to adjust its weight so as to reach the Nash equilibrium. This function may involve different quantities. Some QuGANs make use of the quantum states of the generator and the discriminator, and some others involve a certain function estimated by the discriminator (in this case, it is called the critic). However, in most QuGANs, the discriminating results of the discriminator are used to determine the loss.

4.1.1. Maximum Likelihood

Inspired by the classical GANs, the generator and the discriminator also play a minimax game. Suppose x is a data point drawn from the real data distribution and x' is a sample generated by the generator. The discriminator should take x or x' as input, label 1 if the input is a real sample, and label 0 otherwise. Denote $D(\cdot)$ as the probability that the data $(\cdot)$ are from the real distribution. The goal of the discriminator is to make $D(x)$ as near 1, and $D(x')$ as near 0 as possible. Meanwhile, the generator aims to fool the discriminator by maximizing $D(x')$. Therefore, the cost of training a quantum GAN can be written as a value function:

$$V(G, D) = E_{x\sim\sigma}[\log D(x)] + E_{x'\sim\rho}[\log(1 - D(x'))], \tag{9}$$

where $E[\cdot]$ is the expectation value, σ is the real data distribution, and ρ is the fake data distribution generated by the generator. Training a quantum GAN means solving a minimax problem: $\max_D \min_G V(G, D)$. This loss function is popular and used in most quantum GAN works [28–30,79–81].

However, just like in the case of traditional GANs, in the beginning, the generated and the real distributions may hardly overlap due to the poorly trained generator, which enables the discriminator to easily classify the data with very high accuracy. This results in the quick convergence of the discriminator and provides insufficient gradients for the generator to learn because its loss function saturates. Therefore, in practice, instead of minimizing $E_{x'\sim\rho}[\log(1 - D(x'))]$, the generator should attempt to maximize the function:

$$V(G) = E_{x'\sim\rho}[\log(D(x'))]. \tag{10}$$

This function helps provide gradients with more information during the early stage of learning while still resulting in the same optimal point [17]. Nonetheless, the discriminator and the generator are trained simultaneously without considering each other. Therefore, the gradients have large variances and fluctuate, which makes it uncertain that the training process will end up with the convergence of the network.

In addition, the maximum likelihood cost function does not help the network overcome mode collapse, and the generator produces an especially plausible output and it keeps generating output similar or even identical to that one. This phenomenon happens when the discriminator is stuck in a local minimum and facilitates the generator to overoptimize for that particular discriminator. The generator will then rotate around a small space of output with little variance [82]. In addition, optimizing the maximum likelihood cost function is equivalent to minimizing the KLD between the real and the generated distribution [83]. In principle, if we consider a distance between two distributions as a cost function, minimizing this function will make one distribution closer to the other, and this only holds if the distance is continuous. Otherwise, at the point of discontinuity, the gradients become useless [34].

4.1.2. Wasserstein Distance

From the classical WD, Chakrabarti et al. made a reference to the quantum terms and constructed a quantum Wasserstein semi-metric [84]. Denote the set of quantum states over space X as $D(X)$ and the trace of a matrix as $Tr(\cdot)$. The quantum Wasserstein semi-metric between two states $\mathcal{P} \in D(X)$ and $\mathcal{Q} \in D(Y)$ is

$$qW(\mathcal{P}, \mathcal{Q}) := \min_{\pi} Tr(\pi C)$$

$$s.t. \qquad Tr_Y(\pi) = \mathcal{P}, Tr_X(\pi) = \mathcal{Q}, \pi \in D(X \otimes Y), \tag{11}$$

where π is the diagonal matrix representing the coupling distribution $\pi(x, y)$ satisfying $Tr_Y(\pi) = \mathcal{P}$ and $Tr_X(\pi) = \mathcal{Q}$ in the diagonal, and C is a Hermitian matrix. In [84], the authors chose $C = \frac{1}{2}(I_{X\otimes Y}) - \text{SWAP}$ to leverage the concept of symmetric subspace

in quantum information. Here, $I_{X\otimes Y}$ is the identity operator over $X \otimes Y$, and SWAP is an operator that swaps $\overrightarrow{x} \otimes \overrightarrow{y}$ into $\overrightarrow{y} \otimes \overrightarrow{x}$ for all $\overrightarrow{x} \in X$ and $\overrightarrow{y} \in Y$.

The dual form of the quantum Wasserstein semi-metric is

$$\max_{\phi,\psi} \quad Tr(\mathcal{Q}\psi) - Tr(\mathcal{P}\phi)$$

$$s.t. \qquad I_X \otimes \psi - \phi \otimes I_Y \preceq C, \phi \in \mathcal{H}(X), \psi \in \mathcal{H}(Y), \tag{12}$$

where $\mathcal{H}(X)$ and $\mathcal{H}(Y)$ are the sets of Hermitian matrices over space X and Y, respectively, [84].

Similarly to the Lipschitz constraint, the quantum Wasserstein semi-metric also has a condition of $I_X \otimes \psi - \phi \otimes I_Y \preceq C$; i.e., $C = (I_X \otimes \psi - \phi \otimes I_Y)$ is a positive semidefinite matrix. To satisfy this condition, Chakrabarti et al. added a quantum-relative-entropy-based regularizer with a tunable coefficient λ to the semi-metric:

$$\min_{\pi} \quad Tr(\pi C) + \lambda Tr(\pi \log(\pi) - \pi \log(\mathcal{P} \otimes \mathcal{Q}))$$

$$s.t. \qquad Tr_Y(\pi) = \mathcal{P}, Tr_X(\pi) = \mathcal{Q}, \pi = D(X \otimes Y). \tag{13}$$

Accordingly, Chakrabarti et al. constructed an approximation of the semi-metric in the dual form:

$$\max_{\phi,\psi} \quad E_{\mathcal{Q}}[\psi] - E_{\mathcal{P}}[\phi] - E_{\mathcal{P}\otimes\mathcal{Q}}[\xi_R] \quad s.t. \ \phi \in \mathcal{H}(X), \psi \in \mathcal{H}(Y), \tag{14}$$

where

$$\xi_R = \frac{\lambda}{e} \exp\left(\frac{-C - \phi \otimes I_Y + I_X \otimes \psi}{\lambda}\right). \tag{15}$$

Optimizing the quantum Wasserstein semi-metric as the cost function is equivalent to finding out the optimal parameters ψ and ϕ of the discriminator to approximate the exact Wasserstein semi-metric between the real and the generated quantum states.

4.1.3. Total Variation

In the classical scenario, total variation is not a favorite type of cost function for GANs. It is normally used as a metric to evaluate the performance of a network. Only a few works have used total variation as a cost function, such as [85]. When it comes to quantum terms, the total variation can be written as the trace distance of the target quantum state ρ_R and the generated quantum state ρ_G:

$$D(\rho_R, \rho_G) = \frac{1}{2} Tr(\rho_R - \rho_G). \tag{16}$$

In the case where the network converges, this distance reaches 0. At that time, the generated state is identical to the target state. We can see this type of cost function applied in [86,87].

In [27], we can see that the first proposed quantum GAN also used total variation as a cost function. Although classical GANs which inspire quantum GANs mostly use log-likelihood as the cost function, Dallaire-Demers et al. [27] defined such a linear cost function for simplicity and convenience. When the source of data is fairly selected, the final form of the cost function is

$$\frac{1}{2} + \frac{1}{4\Lambda} \sum_{\lambda=1}^{\Lambda} Tr((\rho_R - \rho_G)Z), \tag{17}$$

where λ is the data label; Λ is the total number of labels; ρ_R and ρ_G are the quantum states of the system after applying the discriminator in the cases of real data and generated data,

respectively; and the expectation value of the operator $Z = |\text{real}\rangle \langle\text{real}| - |\text{fake}\rangle \langle\text{fake}|$ is proportional to the probability that the discriminator yields $|\text{real}\rangle$.

4.2. Gradient Computation

In quantum GANs, there are parametrized quantum circuits that build one or more parts of the network. That part can be the generator, or the discriminator, or both, or even the supplement components to assist the network in working more efficiently. These variational circuits consist of quantum gates or unitaries with tunable continuous parameters. Similarly to the classical counterparts, during the training process, the parameters are updated using a certain optimizing algorithm, such as gradient descent [88], Adam [89], or Adagrad [90]. All these approaches to adjusting the circuits to optimality require calculating the gradients of the cost function, including the partial derivatives with respect to the parameters of the circuits.

Consider a gate $U_i(\theta_i)$ in a variational circuit consisting of a gate sequence $U(\theta) = U_N(\theta_N)U_{N-1}(\theta_{N-1})...U_i(\theta_i)...U_0(\theta_0)$ that depends on a set of parameters θ. Suppose the state of the circuit before being applied $U_i(\theta_i)$ is $|\psi\rangle$. After going through $U_i(\theta_i)$, the circuit state can be measured by an observable $\hat{Q}$. Then, the circuit can be represented as a function $f(\theta)$:

$$f(\theta) := \langle\psi| U_i^\dagger(\theta_i)\hat{Q}U_i(\theta_i) |\psi\rangle . \tag{18}$$

The partial derivative of the function f with respect to the parameter θ_i of the gate $U_i(\theta_i)$ is

$$\frac{\partial f}{\partial \theta_i} = \langle\psi| \frac{\partial U_i^\dagger(\theta_i)\hat{Q}U_i(\theta_i)}{\partial \theta_i} |\psi\rangle . \tag{19}$$

Similarly to other quantum machine learning fields of research, there are two main methods to calculate $\frac{\partial f}{\partial \theta_i}$, namely, the parameter-shift rule and linear combination of unitaries.

4.2.1. Parameter-Shift Method

The parameter-shift method was first proposed by Mitarai et al. [91], and then was expanded by Schuld et al. [92]. In many cases, the partial derivative of a unitary conjugation $U_i^\dagger(\theta_i)\hat{Q}U_i(\theta_i)$ with respect to θ_i can be expressed by that unitary conjugation itself but with a shift s in θ_i:

$$\frac{\partial U_i^\dagger(\theta_i)\hat{Q}U_i(\theta_i)}{\partial \theta_i} = r[U_i^\dagger(\theta_i + s)\hat{Q}U_i(\theta_i + s) - U_i^\dagger(\theta_i - s)\hat{Q}U_i(\theta_i - s)], \tag{20}$$

where r is the positive eigenvalue of G, and $s = \frac{\pi}{4r}$. For example, if U_i is generated by a Pauli product P_i, i.e., $G = \frac{P_i}{2}$, then $r = \frac{1}{2}$ and $s = \frac{\pi}{2}$.

This method can only be applied to the cases where the gate is in the form of $U_i(\theta_i) = exp(-i\theta_i G)$ and the Hermitian operator G has at most two distinct eigenvalues. Many of the works about quantum GANs use variational circuits consisting of this kind of quantum gates, so it is used in a variety of quantum GAN optimization algorithms [28–30,80,81,84,87,93]. Making use of the parameter-shift rule, one can easily calculate the gradient of the cost function without impacting the network or constructing another circuit by executing the same variational circuit again with only small modifications to the considered parameters.

4.2.2. Linear Combination of Unitaries

As for the quantum gates that do not belong to the types of gates above, their gradients can always be evaluated by using their linear combinations of unitaries. Schuld et al. first suggested the idea of decomposing the gates into linear unitary combinations in [94], and then generalized the idea in [92].

The partial derivative of the function f with respect to θ_i can be further decomposed as

$$\frac{\partial f}{\partial \theta_i} = \sum_{k=1}^{K} \alpha_k (\langle \psi | U_i^\dagger(\theta_i) \hat{Q} A_k | \psi \rangle + h.c.), \tag{21}$$

where α_k is real and A_k is a unitary matrix. For each individual term $\alpha_k(\langle \psi | U_i^\dagger(\theta_i)\hat{Q}A_k | \psi \rangle + h.c.)$ in the sum, a corresponding quantum circuit, as in Figure 3, is implemented. An ancilla bit in state $|0\rangle$ is added and applied to a Hadamard gate, followed by the application of the unitary U_i conditioned on the ancilla's state $|0\rangle$, and the unitary A_k conditioned on the ancilla's state $|1\rangle$. Finally, the ancilla is applied to another Hadamard gate. Measuring the ancilla bit and the observable $\hat{Q}$ for the final state $|\psi'\rangle$ gives $\tilde{E_0}$, the expectations of $\hat{Q}$ being when $c = |0\rangle$ with a probability of p_0 and $\tilde{E_1}$ when $c = |1\rangle$ with a probability of p_1. The individual term of the sum in the target derivative can then be easily derived:

$$\alpha_k(\langle \psi | U_i^\dagger(\theta_i)\hat{Q}A_k | \psi \rangle + h.c.) = 2\alpha_k(p_0\tilde{E_0} - p_1\tilde{E_1}). \tag{22}$$

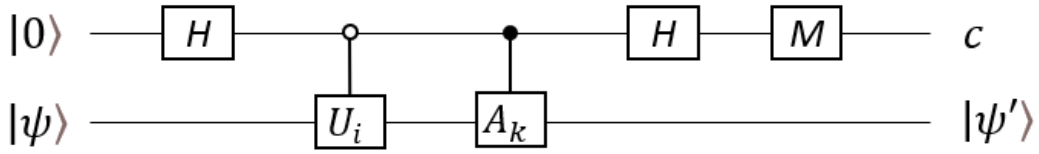


Figure 3. The implemented circuit for computing an individual term in the target derivative using gate decomposition.

Although this method can be applied for any kind of gate, it requires an additional qubit as the ancilla and applying the gates and their decompositions conditioned on the ancilla. In addition, one must execute the circuit several times to estimate the probabilities and the expected values for each decomposition component of the gate, and there are in total K such components, which is resource-consuming. In addition, how to decompose the gate must also be determined. However, for the gates that are not in the form of $exp(-i\theta_i G)$, or when the generator G of the gates has more than two eigenvalues, this is the only way to compute the gradient. Some quantum GAN architectures using this method are [79,84]. In the first-ever suggested quantum GAN architecture [27], although the derivatives are computed directly, the method can also be categorized as linear combinations of unitaries with a special $K = 1$.

4.3. Optimization and Evaluation Strategies

A good strategy to optimize a quantum GAN is also an important research direction with a lot of interesting questions. For example, what is the order of training the generator and the discriminator that results in the most efficient optimization process? How much should one train the generator in comparison with the discriminator? How many steps of gradients should the network go through? How can the parameters of a network be initialized for the best performance? Additionally, how can one set or even adjust the learning rate to adapt to the change during the training process?

These problems matter when training a classical GAN, and they are certainly carried through to the quantum counterparts. With the exponential computational power and the more complicated algorithms, researchers should carefully set up the hyperparameters and determine the optimization strategy before conducting the training process.

The solutions to these issues, however, are rather indiscriminate, and due to the different structures of the networks and the elusiveness of quantum nature, there has been no rule to find out the best training strategy that can be applied for every quantum network in general, and for every quantum GAN in particular. To tackle the learning rate and the number of measurement shots, Huang et al. [29] simply performed a grid search to find the optimal hyperparameters. In the same manner, some researchers also set the

hyperparameters and adjust them manually during the training process. For example, Dallaire-Demers et al. [27] set the number of gradient steps to 10,000 and the initial learning rate of the discriminator to 10; the latter was made to decrease exponentially to 0.1 during the first 4000 steps and then remain constant. The learning rate of the generator was 5 times as much as that of the discriminator. In [28], Stein et al. trained the network 25 times with the constant learning rate of 0.01, and the initial weights of the network were drawn randomly from $(0, \pi)$. This was also the case for a quantum WGAN [84], where Chakrabarti et al. set a fixed learning rate for both the generator and the discriminator. Many others have seen the benefits of preconstructed optimizing algorithms, so they directly use these for training their networks. For instance, Zoufal et al. [30] made use of the AMSGRAD algorithm in [95] to optimize their quantum GAN.

Jointly training the generator and the discriminator is a more difficult issue to determine. In comparison with the discriminator, if the generator is trained too little, it may not get enough information to defeat the discriminator, which results in instability in the loss of the network. On the contrary, if the generator is trained too much, it will likely lead to mode collapse, and there will be a waste of computational resources. This also varies and depends on each network architecture and the authors' purposes. Many approaches to handle this problem have been applied. To make the data source choice for the discriminator truly fair, Dallaire-Demers et al. [27] tossed a coin to determine the input of the discriminator to be real data or generated data. During the training process, the generator is updated once for every 100 gradient steps of the discriminator. Meanwhile, Huang et al. [29] updated the generator and the discriminator iteratively for a preset number of epochs. Alternatively, in [28], for each iteration, Stein et al. trained the discriminator on all real data once, then on I generated samples, and finally trained the generator for R times. For the quantum supports that produce data prior, as in [78], the quantum part is updated after a certain number of times of training the main network. However, for the quantum WGAN family, thanks to the role of the critics in finding the best function to estimate the Wasserstein distance, not discriminating between real and fake samples, they can be trained to optimality before the generators are trained using the optimal distance.

Like their classical counterparts, quantum GANs require some methods to assess their performances. There are many useful metrics used for classical networks, but those for quantum networks are still an open question. Even in the classical cases, finding suitable metrics that embrace as many aspects as possible, such as network running time, resources needed, and the similarity between generated and real data, is complicated.

To evaluate the performance of the classical GANs, there are many metrics that have been invented and applied. These metrics can be used directly to assess the efficiency of a quantum GAN in the case where the network generates classical data. In [30], Zoufal et al. calculated the relative entropy and the Kolmogorov–Smirnov statistic between the generator's output and the real distribution. Huang et al. [29] used Fréchet distance to measure the similarity of the generated and the target data. The Hellinger distance between the fake dataset and the real dataset is another metric, and it was used in Stein et al.'s work [28]. On the other hand, Rudolph et al. [78] made use of the Inception-v3 network to calculate the inception score to evaluate the proposed network's performance.

The classical metrics can also be used in some cases of quantum data. When proposing the first quantum GAN, Dallaire-Demers et al. [27] applied the KLD method to the trace of the density matrix:

$$S(\rho_R \| \rho_G) = Tr(\rho_R(\log \rho_R - \log \rho_G)). \tag{23}$$

Many of the quantum GANs working with quantum data use quantum fidelity and trace distance to attain the similarity of the generated states and the target states [84,86,87,93,96]. The trace distance of two states ρ_R and ρ_G is defined as

$$D(\rho_R, \rho_G) = \frac{1}{2} Tr(\rho_R - \rho_G). \tag{24}$$

The fidelity of two states ρ_R and ρ_G is calculated using

$$F(\rho_R, \rho_G) = (Tr(\sqrt{\sqrt{\rho_R}\rho_G\sqrt{\rho_R}}))^2. \quad (25)$$

When the calculated values of these metrics approach 0, it means that the generated distribution is near the target distribution, or the evaluated network has a good performance. In addition, in some computer vision tasks, to judge how effectively a network works, one may simply display the generated data and make a perceptual comparison with the training data, as in [79].

5. Quantum GAN Variants

In this section, a wide variety of QuGANs are discussed in detail. A summary of typical quantum GAN variants and their characteristics is provided in Table 1.

5.1. Fully Quantum GANs

With fully quantum GANs, both generators and discriminators are constructed by quantum circuits, connected directly with the other, and they together apply to a system of qubits [27–29,79,86,87,93]. The target distributions can be quantum, which can be fed directly into the network, or classical, which must be encoded to some quantum states before being input into the network. The workflow of fully quantum GANs is depicted in Figure 4. The noise from latent space puts the quantum system in the state $|z\rangle$. After being applied by the generator, the system is in state $|\psi\rangle$. At this time, in the case the real data are used, the real quantum data source outputs state $|x\rangle$ from the quantum system. The discriminator is then applied, and it changes the state to $|y\rangle$. The states $|y\rangle$ in the cases where the discriminator's input is real or generated will then be used for the cost function and updating the parameters in the circuits.

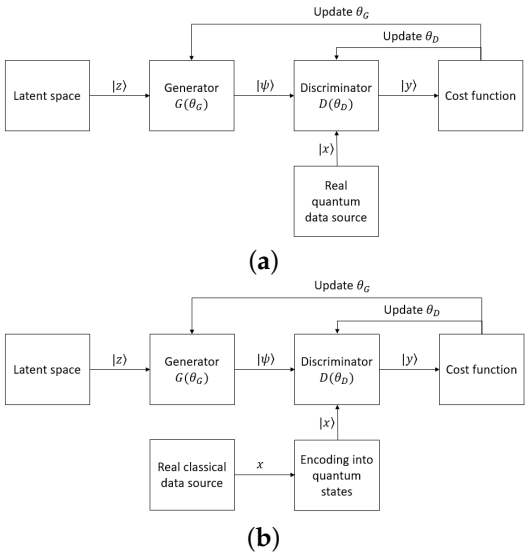


Figure 4. The workflow of a fully quantum network in the cases where the target data are (**a**) quantum and (**b**) classical.

Table 1. A summary of quantum GAN variants and their characteristics.

Year	Name	Author	Data Type	Encoding Method	Conditional	Quantum G	Quantum D	Loss Function	Gradient	Network Evaluation	Application
2018	Quantum generative adversarial networks [27]	Dallaire-Demers et al.	Quantum	None	Yes	Yes	Yes	Total variation	Directly using a separate register	Cross entropy	Generate pure quantum state with 2 labels; Approximate quantum circuits
2019	Quantum Generative Adversarial Networks for learning and loading random distributions [30]	Zoufal et al.	Classical	None	No	Yes	No	Log-likelihood	Parameter-shift	Relative entropy; Kolmogorov–Smirnovm statistic	Learn and load random probability distribution into quantum states; Financial derivative pricing
2019	Quantum Wasserstein Generative Adversarial Networks [84]	Chakrabarti et al.	Quantum	None	No	Yes	Yes	Quantum Wasserstein distance with regularizer	Directly using a separate quantum circuit; Parameter-shift	State fidelity	Approximate quantum circuits
2019	OpticalGAN: Generative Adversarial Networks for Continuous Variable Quantum Computation [79]	Shrivastava et al.	Quantum	None	No	Yes	Yes	Log-likelihood	Gate decomposition	Visualization	Generate energy eigenstates and coherent states
2019	Quantum generative adversarial learning in a superconducting quantum circuit [86]	Hu et al.	Quantum	None	No	Yes	Yes	Total variation	By definition	State fidelity	Replicate a quantum state from a quantum channel simulator
2019	Efficient Online Quantum Generative Adversarial Learning Algorithms with Applications [93]	Du et al.	Quantum	None	No	Yes	Yes	Total variation	Zero-order differential method; Parameter-shift	State fidelity	Entanglement test for a bipartite pure state; Approximate a pure state
2019	Adversarial quantum circuit learning for pure state approximation [87]	Benedetti et al.	Quantum	None	No	Yes	Yes	Total variation	Parameter-shift	Trace distance	Approximate pure states
2020	Quantum generative adversarial network for generating discrete distribution [80]	Situ et al.	Classical	None	No	Yes	No	Log like-lihood	Parameter-shift	Number of valid generated samples; KLD	Generate discrete distribution such as BAS
2021	Experimental Quantum Generative Adversarial Networks for Image Generation [29]	Huang et al.	Classical	Amplitude encoding	No	Yes	Yes	Log-likelihood	Parameter-shift	2-Wasserstein distance (Fréchet distance)	Generate handwritten digit images of 0 and 1; Generate a data set of images (gray scale bar)
	Quantum batch GAN										
	Quantum patch GAN			None			No			Visualization; Fréchet distance	
2021	QuGAN: A Quantum State Fidelity based Generative Adversarial Network [28]	Stein et al.	Classical	Angle encoding	No	Yes	Yes	Log-likelihood	Parameter-shift	Hellinger distance	Generate images of handwritten digits from the MNIST dataset
2021	A hybrid quantum–classical conditional generative adversarial network algorithm for human-centered paradigm in cloud [81]	Liu et al.	Classical	None	Yes	Yes	No	Log-likelihood	Parameter-shift	Time complexity; Number of bits used; Visualization	Generate discrete distribution such as BAS with human orientation on the generated data
2022	Learning quantum data with the quantum earth mover's distance [96]	Kiani et al.	Quantum	None	No	Yes	Yes	Quantum Wasserstein distance	Directly using a linear program; Parameter-shift	State fidelity	Estimate quantum states and approximate quantum circuits

In this variant of quantum GANs, both the generator and the discriminator are boosted by quantum supremacy, and they can benefit from two facts. First, the computational power is leveraged at an exponential rate in comparison with classical neural networks. Second, they can manipulate, learn from, and generate certain types of data that classical GANs cannot [26]. For example, in [29], to achieve the same performance with the quantum batch GAN using totally 21 parameters, the classical GANs must have at least 106 parameters.

5.2. Hybrid Quantum–Classical GANs

A GAN could be a combination of a quantum and a classical module. In practice, a GAN with a classical generator and a quantum discriminator is never used. As stated in [26], if the target distribution is quantum, it is impossible for the classical generator to estimate and learn such a distribution. On the other hand, if the target data are classical, the generator is able to generate such data, but it can always be beaten by the quantum discriminator. In this way, the generator never reaches its convergence. Hence, a hybrid quantum–classical GAN has a quantum generator and a classical discriminator. The architecture of a hybrid quantum–classical GAN is illustrated in Figure 5. This variant of quantum GANs is used to generate classical data with extraordinary performance in comparison with traditional GANs. Since the discriminator is also classical, the target data do not need encoding. However, the states of the quantum system after the generator must be measured to be transformed into classical statistics, which are readable for the discriminator. The classical discriminator can be made of a fully connected neural network [29,30,80], with its output being 0 or 1, corresponding with real and fake examples, respectively. This reduces the problem to a two-class classification.

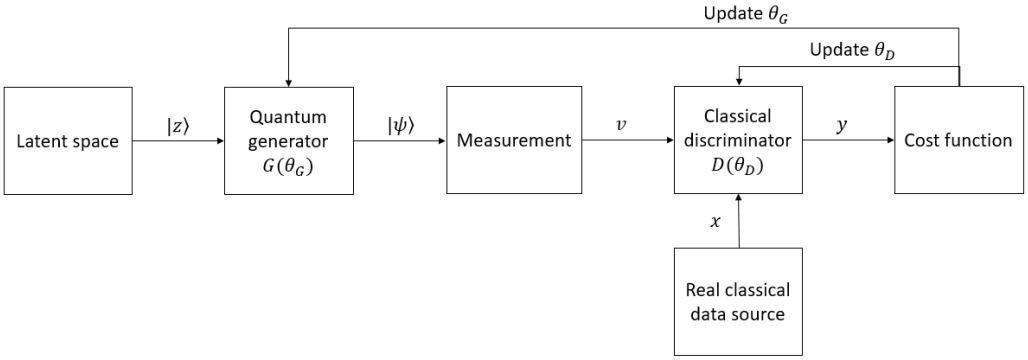


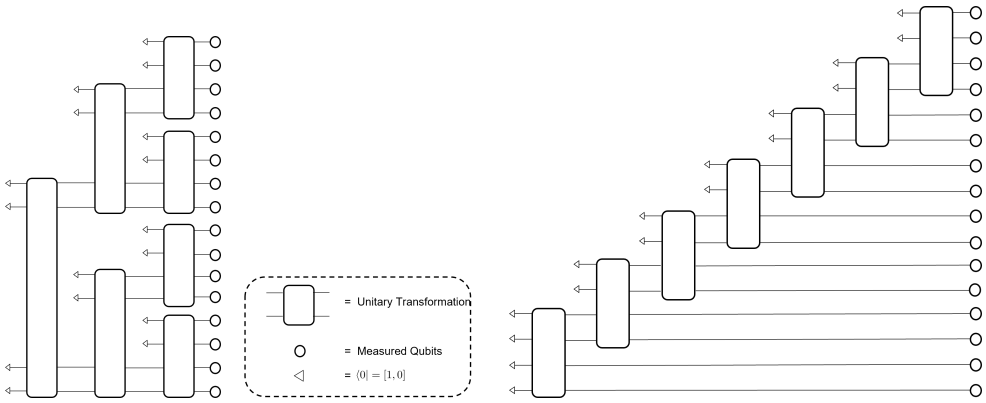
Figure 5. The workflow of a hybrid quantum–classical GAN.

Although only the generator has quantum power, it is enough for hybrid GANs to outperform traditional ones. In [80], a GAN with a quantum generator and a classical discriminator was able to generate discrete distributions, a difficult task for classical GANs to deal with. Most notably, the proposed hybrid GAN could easily converge after only about 1000 epochs.

5.3. Tensor-Network-Based GANs

Tensor networks have recently been considered as a promising design for many generative learning tasks [76,97,98]. In [76], the authors considered two families of tensor networks, namely, matrix product states and tree tensor networks, to model a generative circuit. Their architectures are shown in Figure 6. Rather than applying the universal unitary operators of all qubits, tensor-network-based generative models consider some specific patterns by choosing a subset of qubits for each unitary transformation. The algorithms begin by initializing 2V qubits in one subset in a reference computational basis state $\langle 0|^{\otimes 2V}$, then transform these qubits by a unitary operator. Another subset of 2V qubits is prepared in $\langle 0|^{\otimes 2V}$, and half of them will be entangled with V qubits from the first subset by another unitary operator. The process continues until the total number of qubits reaches the desired

output dimensionality. Figure 6 shows examples of the designs of generators based on tree tensor network (a) and matrix product states (b) with V = 2.

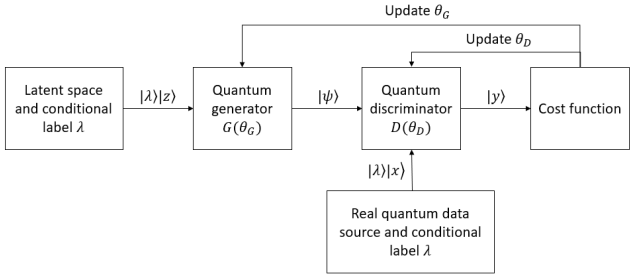


(**a**) Tree Tensor Network (**b**) Matrix Product State

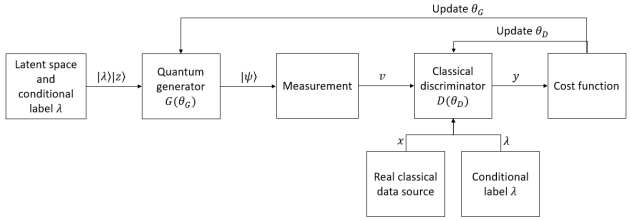
Figure 6. Generative models with tensor networks.

5.4. Quantum Conditional GANs

In classical generative tasks, the input of the generator is random, so one has no control over the generated output. To force the network to produce the examples with desired classes, one conditional constraint about the label is added [33]. Both the generator and the discriminator are aware of this constraint. In addition to discriminating whether the sample comes from a real or generated distribution, the discriminator has to evaluate whether the sample has the characteristics corresponding to the right label or not. This is the same approach when it comes to the quantum scenario. The conditional label λ is also encoded in the form of quantum states. The schematic of quantum conditional GANs is shown in Figure 7.



(**a**)



(**b**)

Figure 7. The workflow of (**a**) a quantum conditional GAN generating quantum data and (**b**) that of a hybrid conditional GAN generating classical data.

In [27], Dallaire-Demers et al. delicately constructed the quantum conditional GAN version based on the structural similarities between classical and quantum mechanics. The real data source R is used to provide real data, and the generator $G(\theta_G)$ and the discriminator $D(\theta_D)$ are parametrized quantum circuits. During the training process, a coin is tossed to determine the source of data. If the outcome is tails, the real data source

is used to train the discriminator. If it is heads, the generator takes the latent noise and produces the fake data. Only in the heads case are both the generator and the discriminator trained. This keeps the network from having the generator trained too much. The cost function used for parameter update is the total variation or the trace distance between the quantum states of the system when the input data of the discriminator is real or generated. The network was tested on generating quantum states $\rho_A^R = |0\rangle\langle 0|$ with label A, and $\rho_B^R = |1\rangle\langle 1|$ with label B. After about 7000 steps of the discriminator, the KLD of the real and generated data reaches zero.

Liu et al. [81] also proposed a hybrid quantum–classical conditional GAN that utilizes human orientation on the generated data. The conditional variable in this network is not a label of data, but it contains the probability distribution of samples in the training set. This classical conditional constraint is encoded as a quantum state as $|\lambda\rangle = \sum_{j=1}^{m} \frac{1}{\alpha_j} |\lambda_j\rangle$, where $\frac{1}{\alpha_j} = (p(x|\lambda_j))^{-1/2}$, λ_j is the j-th label and m is the total number of classes. Of course, $\frac{1}{\alpha_j}$ must meet the condition of amplitudes of a quantum state. Normally, the classical label of an example is in the form of one-hot coding, so it is convenient to express $|\lambda_j\rangle$ as the corresponding basis states. The network uses the log-likelihood cost function and parameter-shift method [92] to calculate the gradients. After being trained for 100 epochs, each epoch with 150 network training iterations, the network could generate a distribution similar to BAS, a discrete distribution, with less time complexity and controllability, using the generated data. Thanks to this human-centered algorithm, quantum GANs of this type have been noticed for their huge potential for the human-centered paradigm in the cloud.

5.5. Quantum Wasserstein GANs

Quantum Wasserstein GANs are the quantum GANs that use Wasserstein distance, or Earth mover's distance (EMD), as their cost functions. Differently from other quantum GANs variants, the mission of the discriminator in Wasserstein GANs (WGANs) is not to distinguish between the real and the generated data. Due to the fact that Wasserstein distance contains a function that satisfies the Lipschitz condition, the discriminator acts as an estimator that finds out the optimal function for calculating the distance. After the distance is computed, it is used to update the parameters in the generator. The workflow of quantum WGANs is illustrated in Figure 8.

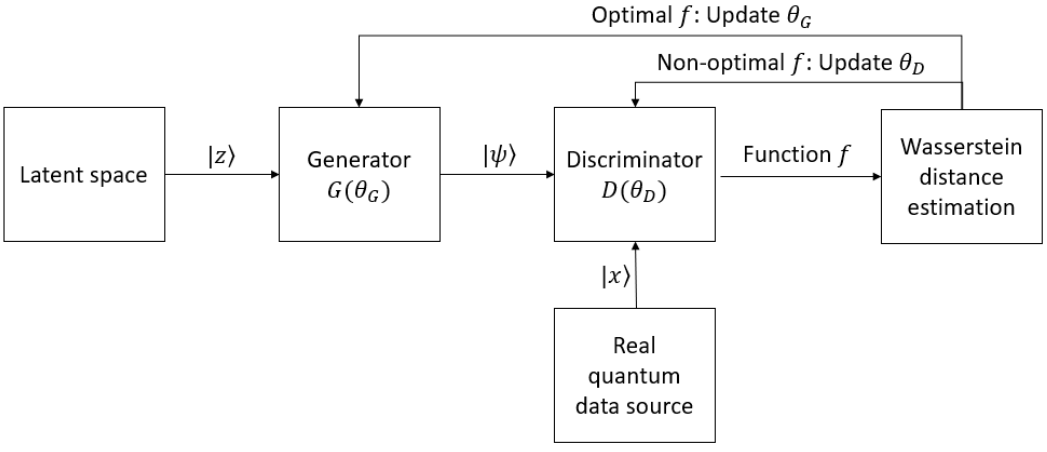


Figure 8. The general workflow of a quantum WGAN.

The first quantum Wasserstein GAN was proposed by Chakrabarti et al. [84]. The generator consists of ensembles of unnitaries $\{(p_1, U_1), ..., (p_r, U_r)\}$, parameterized by the set of parameters θ. Each tuple (p_i, U_i) means applying the unitary U_i, a 1-qubit or 2-qubit Pauli rotation quantum gate, with probability p_i. The discriminator contains linear combinations of tensor products of Pauli matrices that are parameterized by two sets of parameters, α and β, corresponding to ϕ and ψ, respectively, in the formula in Section 4.1.2. The gradients

of the cost function with respect to p_i, α and β are computed directly, since the cost function is a linear function of those parameters. The gradients of the cost function with respect to θ are estimated by the parameter-shift method [92].

However, Kiani et al. realized that this formula of quantum Wasserstein distance is unitarily invariant, so they developed another formula and applied it to their quantum WGAN [96]. The Wasserstein distance between two states, ρ and σ, is

$$D_{EM}(\rho,\sigma) = \max Tr[(\rho-\sigma)H] \qquad s.t. \qquad H \in \mathcal{O}_n, \|H\|_L \leq 1, \tag{26}$$

where $\mathcal{O}_n$ is the set of n-qubit observables. The quantum Lipschitz constant of the observable H must be at most one. The discriminator contains the parameterized sum of strings of Pauli operators and is supposed to estimate the quantum EMD. The generator in this network has the same architecture as that in Chakrabarti et al.'s work [84]. The weights of the discriminator are updated by executing a linear program, the probabilities of the unitaries in the generator are updated by directly computing the gradients, and gradients of the cost function with respect to the unitary parameters are calculated via parameter-shift rules [92].

Quantum Wasserstein GANs can help overcome the disadvantages of quantum GANs using other metrics for network updating, such as discontinuous distances between real and generated distributions, mode collapse, and vanishing gradients. They can successfully estimate GHZ state [96], up to 8 qubit pure states, up to 3 qubit mixed states, and even 4 qubit pure states with noise [84]. They can also be used to approximate quantum circuits such as the 1D 3-qubit Heisenberg model circuit [99] with an average output fidelity of 0.9999 [84] and 8 qubit teacher circuits using student circuits of depth 4 with the fidelity of approximately 1 [96].

5.6. Quantum Patch GANs Using Multiple Sub-Generators

For a classical dataset with M dimensions, whichever encoding method is used, each sample requires at least $N = \log M$ qubits to be represented. To deal with the case there are limited quantum resources, i.e., the number of available qubits, Huang et al. suggested a quantum patch GAN [29]. This network consists of T quantum sub-generators and a classical discriminator. The sub-generators are identical, and each is responsible for a portion of the high-dimensional data. The outputs of the sub-generators are measured and concatenated together to form a classical vector, which then can be fed into the discriminator. Thanks to dividing the data into small parts for each sub-generator, the training can be carried out on distributed quantum devices parallelly or on a single quantum device sequentially.

5.7. Quantum GANs Using Quantum Fidelity for a Cost Function

This fully quantum GAN variant was suggested by Stein et al. [28]. The architectures of the generator and the discriminator, the gradient calculation method, and the training strategy stay the same as other fully quantum GANs, but there is a modification in the cost function. The outcomes of the discriminator when the input is real (x) and fake (x'), i.e., $D(x)$ and $D(x')$, are alternated by the fidelities of the state generated by encoding the real samples (ξ) and the state after applying by the generator (γ), respectively, with the state after applying the discriminator (δ). In other words, the original value function

$$V(G,D) = E_{x\sim\sigma}[\log D(x)] + E_{x'\sim\rho}[\log(1-D(x'))] \tag{27}$$

becomes

$$V(G,D) = E[\log D(|\langle\xi,\delta\rangle|^2)] + E[\log(1-D(|\langle\gamma,\delta\rangle|^2))]. \tag{28}$$

The fidelities of the states are obtained using an ancilla qubit in initial state $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ and then applying a controlled swap gate and a Hadamard gate. The probability that the measurement at the ancilla bit yields 0 is equal to half the needed fidelity plus $\frac{1}{2}$.

6. Conclusions

Quantum GAN is a new and potential field of research in quantum machine learning. This kind of quantum generative network is inspired by classical GANs, which have already proved their effectiveness and wide applications. In addition to the outstanding nature of GANs, quantum GANs even perform with higher efficiency due to their unique quantum properties and exponential computing power.

In this work, we have reviewed recent advances in quantum GANs in terms of their architectures, input encodings, loss functions, gradient calculation methods, and network training strategies. Different variants of QuGANs were discussed in detail. Although benefiting from quantum supremacy, quantum GANs still have problems, such as instability in the training process, vanishing gradients, and mode collapse. In addition, the choices of latent space, parameter initialization, and circuit architecture also have crucial impacts on the performance of a network. In future work, we will carry out extensive evaluations of QuGAN proposals and architecture-related options.

Author Contributions: Conceptualization, T.C.T. and T.A.N.; methodology, T.A.N., T.N. and T.C.T.; formal analysis, T.A.N. and T.C.T.; investigation, T.A.N., T.N. and T.C.T.; writing—original draft preparation, T.A.N., T.N. and T.C.T.; writing—review and editing, T.A.N. and T.C.T.; supervision, T.C.T.; project administration, T.C.T.; funding acquisition, T.C.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the competitive fund of the University of Aizu.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mitchell, T.; Buchanan, B.; DeJong, G.; Dietterich, T.; Rosenbloom, P.; Waibel, A. Machine Learning. *Annu. Rev. Comput. Sci.* **1990**, *4*, 417–433. [CrossRef]
2. McCulloch, W.S.; Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* **1943**, *5*, 115–133. [CrossRef]
3. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **1958**, *65 6*, 386–408. [CrossRef] [PubMed]
4. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
5. Hopfield, J.J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci.* **1982**, *79*, 2554–2558. [CrossRef]
6. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
7. Alam, M.M.; Mohiuddin, K.; Das, A.K.; Islam, M.K.; Kaonain, M.S.; Ali, M.H. A Reduced Feature Based Neural Network Approach to Classify the Category of Students. In Proceedings of the 2nd International Conference on Innovation in Artificial Intelligence, Shanghai, China, 9–12 March 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 28–32. [CrossRef]
8. Luckin, R.; Holmes, W.; Griffiths, M.; Forcier, L.B. *Intelligence Unleashed: An Argument for AI in Education*; Pearson Education: London, UK, 2016.
9. Djambic, G.; Krajcar, M.; Bele, D. Machine learning model for early detection of higher education students that need additional attention in introductory programming courses. *Int. J. Digit. Technol. Econ.* **2016**, *1*, 1–11.
10. Amatya, S.; Karkee, M.; Gongal, A.; Zhang, Q.; Whiting, M.D. Detection of cherry tree branches with full foliage in planar architecture for automated sweet-cherry harvesting. *Biosyst. Eng.* **2016**, *146*, 3–15. [CrossRef]
11. Pantazi, X.E.; Moshou, D.; Bravo, C. Active learning system for weed species recognition based on hyperspectral sensing. *Biosyst. Eng.* **2016**, *146*, 193–202. [CrossRef]
12. Bouri, E.; Gkillas, K.; Gupta, R.; Pierdzioch, C. Forecasting Realized Volatility of Bitcoin: The Role of the Trade War. *Comput. Econ.* **2021**, *57*, 29–53. [CrossRef]
13. Lussange, J.; Lazarevich, I.; Bourgeois-Gironde, S.; Palminteri, S.; Gutkin, B. Modelling Stock Markets by Multi-agent Reinforcement Learning. *Comput. Econ.* **2021**, *57*, 113–147. [CrossRef]
14. Sughasiny, M.; Rajeshwari, J. Application of Machine Learning Techniques, Big Data Analytics in Health Care Sector—A Literature Survey. In Proceedings of the 2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 30–31 August 2018; pp. 741–749. [CrossRef]
15. Hazra, A.; Kumar, S.; Gupta, A. Study and Analysis of Breast Cancer Cell Detection using Naïve Bayes, SVM and Ensemble Algorithms. *Int. J. Comput. Appl.* **2016**, *145*, 39–45. [CrossRef]

16. Otoom, A.; Abdallah, E.; Kilani, Y.; Kefaye, A.; Ashour, M. Effective diagnosis and monitoring of heart disease. *Int. J. Softw. Eng. Its Appl.* **2015**, *9*, 143–156. [CrossRef]
17. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K., Eds.; Curran Associates: New York, NY, USA, 2014; Volume 27.
18. Farajzadeh-Zanjani, M.; Razavi-Far, R.; Saif, M.; Palade, V., Generative Adversarial Networks: A Survey on Training, Variants, and Applications. In *Generative Adversarial Learning: Architectures and Applications*; Razavi-Far, R., Ruiz-Garcia, A., Palade, V., Schmidhuber, J., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 7–29. [CrossRef]
19. Pradhyumna, P.; Mohana. A Survey of Modern Deep Learning based Generative Adversarial Networks (GANs). In Proceedings of the 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 29–31 March 2022; pp. 1146–1152. [CrossRef]
20. Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. [CrossRef]
21. Harrow, A.W.; Montanaro, A. Quantum computational supremacy. *Nature* **2017**, *549*, 203–209. [CrossRef]
22. Biamonte, J.; Wittek, P.; Pancotti, N.; Rebentrost, P.; Wiebe, N.; Lloyd, S. Quantum machine learning. *Nature* **2017**, *549*, 195–202. [CrossRef]
23. Dong, D.; Chen, C.; Li, H.; Tarn, T.J. Quantum Reinforcement Learning. *IEEE Trans. Syst. Man Cybern. Part B* **2008**, *38*, 1207–1220. [CrossRef]
24. Rebentrost, P.; Mohseni, M.; Lloyd, S. Quantum Support Vector Machine for Big Data Classification. *Phys. Rev. Lett.* **2014**, *113*, 130503. [CrossRef]
25. Khoshaman, A.; Vinci, W.; Denis, B.; Andriyash, E.; Sadeghi, H.; Amin, M.H. Quantum variational autoencoder. *Quantum Sci. Technol.* **2018**, *4*, 14001. [CrossRef]
26. Lloyd, S.; Weedbrook, C. Quantum Generative Adversarial Learning. *Phys. Rev. Lett.* **2018**, *121*, 40502. [CrossRef]
27. Dallaire-Demers, P.L.; Killoran, N. Quantum generative adversarial networks. *Phys. Rev. A* **2018**, *98*, 12324. [CrossRef]
28. Stein, S.A.; Baheri, B.; Chen, D.; Mao, Y.; Guan, Q.; Li, A.; Fang, B.; Xu, S. QuGAN: A Quantum State Fidelity based Generative Adversarial Network. In Proceedings of the 2021 IEEE International Conference on Quantum Computing and Engineering (QCE), Broomfield, CO, USA, 17–22 October 2021; IEEE: Piscataway, NJ, USA, 2021. [CrossRef]
29. Huang, H.L.; Du, Y.; Gong, M.; Zhao, Y.; Wu, Y.; Wang, C.; Li, S.; Liang, F.; Lin, J.; Xu, Y.; et al. Experimental Quantum Generative Adversarial Networks for Image Generation. *Phys. Rev. Appl.* **2021**, *16*, 24051. [CrossRef]
30. Zoufal, C.; Lucchi, A.; Woerner, S. Quantum Generative Adversarial Networks for learning and loading random distributions. *NPJ Quantum Inf.* **2019**, *5*, 103. [CrossRef]
31. Li, T.; Zhang, S.; Xia, J. Quantum Generative Adversarial Network: A Survey. *Comput. Mater. Contin.* **2020**, *64*, 401–438. [CrossRef]
32. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*; Pereira, F., Burges, C., Bottou, L., Weinberger, K., Eds.; Curran Associates: New York, NY, USA, 2012; Volume 25.
33. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arXiv:1411.1784.
34. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein GAN. *arXiv* **2017**, arXiv:1701.07875.
35. Berthelot, D.; Schumm, T.; Metz, L. BEGAN: Boundary Equilibrium Generative Adversarial Networks. *arXiv* **2017**, arXiv:1703.10717.
36. Brock, A.; Donahue, J.; Simonyan, K. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
37. Denton, E.; Chintala, S.; Szlam, A.; Fergus, R. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. *arXiv* **2015**, arXiv:1506.05751.
38. Chen, X.; Duan, Y.; Houthooft, R.; Schulman, J.; Sutskever, I.; Abbeel, P. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. *arXiv* **2016**, arXiv:1606.03657.
39. Karras, T.; Aittala, M.; Laine, S.; Härkönen, E.; Hellsten, J.; Lehtinen, J.; Aila, T. Alias-Free Generative Adversarial Networks. *arXiv* **2021**, arXiv:2106.12423.
40. Tang, X.; Wang, Z.; Luo, W.; Gao, S. Face Aging with Identity-Preserved Conditional Generative Adversarial Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7939–7947. [CrossRef]
41. Wu, X.; Xu, K.; Hall, P. A survey of image synthesis and editing with generative adversarial networks. *Tsinghua Sci. Technol.* **2017**, *22*, 660–674. [CrossRef]
42. Dolhansky, B.; Ferrer, C.C. Eye In-Painting with Exemplar Generative Adversarial Networks. *arXiv* **2017**, arXiv:1712.03999.
43. Demir, U.; Unal, G. Patch-Based Image Inpainting with Generative Adversarial Networks. *arXiv* **2018**, arXiv:1803.07422.
44. Wu, H.; Zheng, S.; Zhang, J.; Huang, K. GP-GAN: Towards Realistic High-Resolution Image Blending. *arXiv* **2017**, arXiv:1703.07195.
45. Chen, B.C.; Kae, A. Toward Realistic Image Compositing With Adversarial Learning. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 8407–8416. [CrossRef]

46. Wang, X.; Yu, K.; Wu, S.; Gu, J.; Liu, Y.; Dong, C.; Loy, C.C.; Qiao, Y.; Tang, X. ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. *arXiv* **2018**, arXiv:1809.00219.
47. Ding, Z.; Liu, X.Y.; Yin, M.; Kong, L. TGAN: Deep Tensor Generative Adversarial Nets for Large Image Generation. *arXiv* **2019**, arXiv:1901.09953.
48. Wang, C.; Xu, C.; Wang, C.; Tao, D. Perceptual Adversarial Networks for Image-to-Image Transformation. *IEEE Trans. Image Process.* **2018**, *27*, 4066–4079. [CrossRef]
49. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *arXiv* **2017**, arXiv:1703.10593.
50. Liu, M.Y.; Breuel, T.; Kautz, J. Unsupervised Image-to-Image Translation Networks. *arXiv* **2017**, arXiv:1703.00848.
51. Kong, J.; Kim, J.; Bae, J. HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis. *arXiv* **2020**, arXiv:2010.05646.
52. Oord, A.v.d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; Kavukcuoglu, K. WaveNet: A Generative Model for Raw Audio. *arXiv* **2016**, arXiv:1609.03499.
53. Dong, H.W.; Hsiao, W.Y.; Yang, L.C.; Yang, Y.H. MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. *arXiv* **2017**, arXiv:1709.06298.
54. Engel, J.; Agrawal, K.K.; Chen, S.; Gulrajani, I.; Donahue, C.; Roberts, A. GANSynth: Adversarial Neural Audio Synthesis. *arXiv* **2019**, arXiv:1902.08710.
55. Uř ičář, M.; Křížek, P.; Hurych, D.; Sobh, I.; Yogamani, S.; Denny, P. Yes, we GAN: Applying adversarial techniques for autonomous driving. *Electron. Imaging* **2019**, *2019*, 48-1–48-17. [CrossRef]
56. Jeong, C.H.; Yi, M.Y. Correcting rainfall forecasts of a numerical weather prediction model using generative adversarial networks. *J. Supercomput.* **2022**, *79*, 1289–1317. [CrossRef]
57. Besombes, C.; Pannekoucke, O.; Lapeyre, C.; Sanderson, B.; Thual, O. Producing realistic climate data with generative adversarial networks. *Nonlinear Process. Geophys.* **2021**, *28*, 347–370. [CrossRef]
58. Sandfort, V.; Yan, K.; Pickhardt, P.J.; Summers, R.M. Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks. *Sci. Rep.* **2019**, *9*, 16884. [CrossRef]
59. Frid-Adar, M.; Diamant, I.; Klang, E.; Amitai, M.; Goldberger, J.; Greenspan, H. GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. *Neurocomputing* **2018**, *321*, 321–331. [CrossRef]
60. Cheng, J.; Yang, Y.; Tang, X.; Xiong, N.; Zhang, Y.; Lei, F. Generative Adversarial Networks: A Literature Review. *KSII Trans. Internet Inf. Syst.* **2020**, *14*, 4625–4647. [CrossRef]
61. Verdú, S. Total variation distance and the distribution of relative information. In Proceedings of the 2014 Information Theory and Applications Workshop (ITA), San Diego, CA, USA, 9–14 February 2014; pp. 1–3. [CrossRef]
62. Joyce, J.M. Kullback–Leibler Divergence. In *International Encyclopedia of Statistical Science*; Lovric, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 720–722. [CrossRef]
63. Fuglede, B.; Topsoe, F. Jensen-Shannon divergence and Hilbert space embedding. In Proceedings of the International Symposium on Information Theory, Chicago, IL, USA, 27 June–2 July 2004. [CrossRef]
64. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A. Improved Training of Wasserstein GANs. *arXiv* **2017**, arXiv:1704.00028.
65. Petzka, H.; Fischer, A.; Lukovnicov, D. On the regularization of Wasserstein GANs. *arXiv* **2017**, arXiv:1709.08894.
66. Sanjabi, M.; Ba, J.; Razaviyayn, M.; Lee, J.D. On the Convergence and Robustness of Training GANs with Regularized Optimal Transport. *arXiv* **2018**, arXiv:1802.08249.
67. Becker, E.; Pandit, P.; Rangan, S.; Fletcher, A.K. Instability and Local Minima in GAN Training with Kernel Discriminators. *arXiv* **2022**, arXiv:2208.09938.
68. Neyshabur, B.; Bhojanapalli, S.; Chakrabarti, A. Stabilizing GAN Training with Multiple Random Projections. *arXiv* **2017**, arXiv:1705.07831.
69. Ding, Z.; Jiang, S.; Zhao, J. Take a close look at mode collapse and vanishing gradient in GAN. In Proceedings of the 2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 27–29 May 2022; pp. 597–602. [CrossRef]
70. Schuld, M. Supervised quantum machine learning models are kernel methods. *arXiv* **2021**, arXiv:2101.11020.
71. Romero, J.; Aspuru-Guzik, A. Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions. *arXiv* **2019**, arXiv:1901.00848.
72. Nguyen, T.; Paik, I.; Watanobe, Y.; Thang, T.C. An Evaluation of Hardware-Efficient Quantum Neural Networks for Image Data Classification. *Electronics* **2022**, *11*, 437. [CrossRef]
73. Nguyen, T.; Paik, I.; Sagawa, H.; Thang, T.C. Quantum machine learning with quantum image representations. In Proceedings of the 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), Broomfield, CO, USA, 18–23 September 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 851–854.
74. Onuoha, C.; Flaherty, J.; Nguyen, T.; Thang, T.C. Data Scanning Methods for Quantum-Classical Interface in Quanvolutional Neural Networks. In Proceedings of the 2022 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Las Vegas, NV, USA, 7–9 January 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–4.
75. Brandt, H.E. Positive operator valued measure in quantum information processing. *Am. J. Phys.* **1999**, *67*, 434–439. [CrossRef]

76. Huggins, W.; Patil, P.; Mitchell, B.; Whaley, K.B.; Stoudenmire, E.M. Towards quantum machine learning with tensor networks. *Quantum Sci. Technol.* **2019**, *4*, 24001. [CrossRef]
77. Cheng, S.; Prentice, I.C.; Huang, Y.; Jin, Y.; Guo, Y.K.; Arcucci, R. Data-driven surrogate model with latent data assimilation: Application to wildfire forecasting. *J. Comput. Phys.* **2022**, *464*, 111302. [CrossRef]
78. Rudolph, M.S.; Toussaint, N.B.; Katabarwa, A.; Johri, S.; Peropadre, B.; Perdomo-Ortiz, A. Generation of High-Resolution Handwritten Digits with an Ion-Trap Quantum Computer. *arXiv* **2020**, arXiv:2012.03924.
79. Shrivastava, N.; Puri, N.; Gupta, P.; Krishnamurthy, B.; Verma, S. OpticalGAN: Generative Adversarial Networks for Continuous Variable Quantum Computation. *arXiv* **2019**, arXiv:1909.07806.
80. Situ, H.; He, Z.; Wang, Y.; Li, L.; Zheng, S. Quantum generative adversarial network for generating discrete distribution. *Inf. Sci.* **2020**, *538*, 193–208. [CrossRef]
81. Liu, W.; Zhang, Y.; Deng, Z.; Zhao, J.; Tong, L. A hybrid quantum-classical conditional generative adversarial network algorithm for human-centered paradigm in cloud. *EURASIP J. Wirel. Commun. Netw.* **2021**, *2021*, 37. [CrossRef]
82. Zhang, K. On Mode Collapse in Generative Adversarial Networks. In *Artificial Neural Networks and Machine Learning—ICANN 2021*; Farkaš, I., Masulli, P., Otte, S., Wermter, S., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 563–574.
83. Arjovsky, M.; Bottou, L. Towards Principled Methods for Training Generative Adversarial Networks. *arXiv* **2019**, arXiv:1701.04862.
84. Chakrabarti, S.; Huang, Y.; Li, T.; Feizi, S.; Wu, X. Quantum Wasserstein Generative Adversarial Networks. *arXiv* **2019**, arXiv:1911.00111.
85. Zhao, J.; Mathieu, M.; LeCun, Y. Energy-based Generative Adversarial Network. *arXiv* **2019**, arXiv:1609.03126.
86. Hu, L.; Wu, S.H.; Cai, W.; Ma, Y.; Mu, X.; Xu, Y.; Wang, H.; Song, Y.; Deng, D.L.; Zou, C.L.; et al. Quantum generative adversarial learning in a superconducting quantum circuit. *Sci. Adv.* **2019**, *5*, eaav2761. [CrossRef]
87. Benedetti, M.; Grant, E.; Wossnig, L.; Severini, S. Adversarial quantum circuit learning for pure state approximation. *New J. Phys.* **2019**, *21*, 43023. [CrossRef]
88. Lemaréchal, C. Cauchy and the gradient method. *Doc. Math. Extra* **2012**, *251*, 10.
89. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2015**, arXiv:1412.6980.
90. Lydia, A.; Francis, S. Adagrad—An optimizer for stochastic gradient descent. *Int. J. Inf. Comput. Sci* **2019**, *6*, 566–568.
91. Mitarai, K.; Negoro, M.; Kitagawa, M.; Fujii, K. Quantum circuit learning. *Phys. Rev. A* **2018**, *98*, 032309. [CrossRef]
92. Schuld, M.; Bergholm, V.; Gogolin, C.; Izaac, J.; Killoran, N. Evaluating analytic gradients on quantum hardware. *Phys. Rev. A* **2019**, *99*. [CrossRef]
93. Du, Y.; Hsieh, M.H.; Tao, D. Efficient Online Quantum Generative Adversarial Learning Algorithms with Applications. *arXiv* **2019**, arXiv:1904.09602.
94. Schuld, M.; Bocharov, A.; Svore, K.M.; Wiebe, N. Circuit-centric quantum classifiers. *Phys. Rev. A* **2020**, *101*, 032308. [CrossRef]
95. Reddi, S.J.; Kale, S.; Kumar, S. On the Convergence of Adam and Beyond. *arXiv* **2019**. [CrossRef]
96. Kiani, B.T.; Palma, G.D.; Marvian, M.; Liu, Z.W.; Lloyd, S. Learning quantum data with the quantum earth mover's distance. *Quantum Sci. Technol.* **2022**, *7*, 045002. [CrossRef]
97. Han, Z.Y.; Wang, J.; Fan, H.; Wang, L.; Zhang, P. Unsupervised Generative Modeling Using Matrix Product States. *Phys. Rev. X* **2018**, *8*, 031012. [CrossRef]
98. Guo, C.; Jie, Z.; Lu, W.; Poletti, D. Matrix product operators for sequence-to-sequence learning. *Phys. Rev. E* **2018**, *98*, 042114. [CrossRef]
99. Childs, A.M.; Maslov, D.; Nam, Y.; Ross, N.J.; Su, Y. Toward the first quantum simulation with quantum speedup. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, 9456–9461. [CrossRef] [PubMed]