IEEE Transactions on Quantum Engineering

Received 20 January 2023; revised 22 September 2023; accepted 22 September 2023; date of publication 26 September 2023; date of current version 30 October 2023.

Digital Object Identifier 10.1109/TQE.2023.3319319

Hybrid Quantum–Classical Generative Adversarial Network for High-Resolution Image Generation

SHU LOK TSANG[1], MAXWELL T. WEST[2], SARAH M. ERFANI[1], AND MUHAMMAD USMAN[2,3]

[1]School of Computing and Information Systems Melbourne School of Engineering, The University of Melbourne, Parkville, VIC 3010, Australia
[2]School of Physics, The University of Melbourne, Parkville, VIC 3010, Australia
[3]Data61, CSIRO, Clayton, VIC 3168, Australia

Corresponding author: Muhammad Usman (e-mail: musman@unimelb.edu.au).

The work of Maxwell T. West was supported by the Australian Government Research Training Program Scholarship. The work of Sarah M. Erfani was supported by the Australian Research Council Discovery Early Career Researcher Award under Grant DE220100680.

ABSTRACT Quantum machine learning (QML) has received increasing attention due to its potential to outperform classical machine learning methods in problems, such as classification and identification tasks. A subclass of QML methods is quantum generative adversarial networks (QGANs), which have been studied as a quantum counterpart of classical GANs widely used in image manipulation and generation tasks. The existing work on QGANs is still limited to small-scale proof-of-concept examples based on images with significant downscaling. Here, we integrate classical and quantum techniques to propose a new hybrid quantum–classical GAN framework. We demonstrate its superior learning capabilities over existing quantum techniques by generating 28×28 pixels grayscale images without dimensionality reduction or classical pre/postprocessing on multiple classes of the standard Modified National Institute of Standards and Technology (MNIST) and Fashion MNIST datasets, which achieves comparable results to classical frameworks with three orders of magnitude less trainable generator parameters. To gain further insight into the working of our hybrid approach, we systematically explore the impact of its parameter space by varying the number of qubits, the size of image patches, the number of layers in the generator, the shape of the patches, and the choice of prior distribution. Our results show that increasing the quantum generator size generally improves the learning capability of the network. The developed framework provides a foundation for future design of QGANs with optimal parameter set tailored for complex image generation tasks.

INDEX TERMS Machine learning algorithms, quantum circuit, quantum computing, quantum algorithm, quantum simulation.

I. INTRODUCTION

Generative adversarial networks (GANs) are one of the best examples of deep learning success in generative learning [1]. It consists of a generator and a discriminator competing against each other, where the generator attempts to generate realistic data (such as images) whereas the discriminator attempts to differentiate between real and generated data. Ultimately, the goal of the framework is to have the generator distribution replicate the training data distribution, which is mathematically equivalent to minimizing the Jensen–Shannon (JS) divergence between them [1]. GANs have been deployed in many application areas, such as image generation [2], future prediction in videos [3], text-to-image synthesis [4], and image-to-image translation [5]. Despite their empirical success, GANs suffer from a variety of problems during training in practice, namely vanishing gradients, mode collapse, and a lack of stopping criteria [6], [7]. There have been many proposed improvements to tackle these problems. One particular proposal is the Wasserstein GAN (WGAN) [7], where the training is reformulated to minimize the Wasserstein distance instead. The WGAN framework has demonstrated empirically that it can effectively tackle the aforementioned problems. However, as with other classical GANs, training on complex datasets

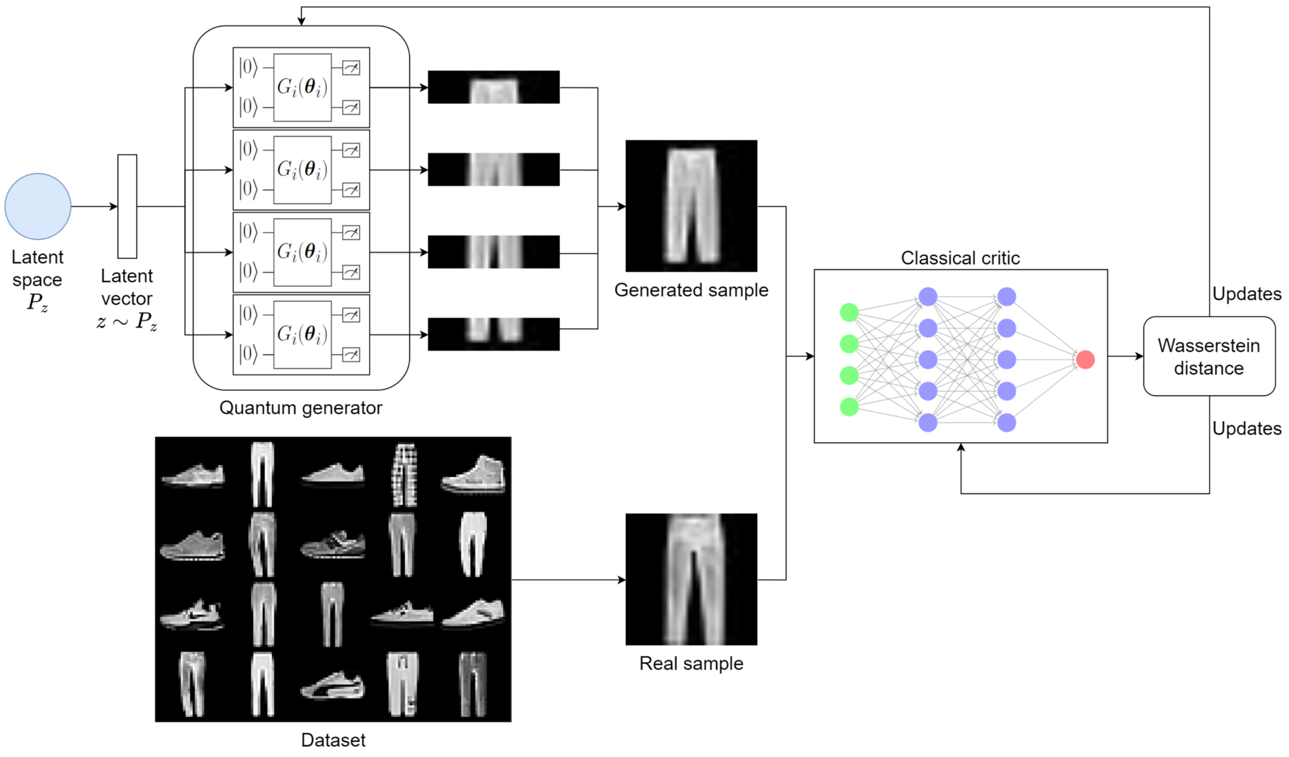


FIGURE 1. Overview of PQWGAN framework. Our framework is identical to the WGAN-GP framework, with the difference being the fake images are now being generated by a quantum generator. The framework operates as follows. First, a latent vector z is sampled from the latent space and is encoded using R_Y rotations as $|z\rangle$ in each subgenerator of the quantum generator. The relevant qubits are measured at the end of the circuit and postprocessed to create a patch of the image. The patches are stitched together to form a complete generated image. The generated and real images are then passed to the critic, which estimates the Wasserstein distance between the generator and real distribution. Finally, these statistics are used to update the generator and the critic.

requires large networks and amounts of computational resources.

The emergence of quantum computing as a new computing paradigm has led to quantum algorithms that show great promise to solve many of the computationally hard problems in computer science, such as Shor's algorithm for efficient prime number factorization [8]. Since quantum mechanics can generate counter intuitive patterns in data, it is believed that quantum computers can recognize classically challenging patterns [9]. These promises have led to the development of quantum machine learning (QML), where quantum algorithms are used to improve existing machine learning techniques [15]. Benefits that QML brings include providing speed-ups in training time [10], [11], [12] or obtaining better model performance [13], [14], [16], [17], [18], [19]. This may allow QML models to complement or even replace classical methods in the future as the complexity of the tasks continuously increases. However, we are currently in the noisy intermediate-scale quantum (NISQ) era of quantum computation [20]. Reliably executing large-scale quantum algorithms on current quantum hardware is difficult due to engineering challenges, such as noise mitigation. Under these restrictions, much QML research has been focused on quantum algorithms that are compatible with NISQ devices [9], such as developing hybrid quantum classical solutions with parameterized quantum circuits (PQCs) [21].

The intersection of quantum computing and GANs have led to the birth of a new research direction known as quantum generative adversarial networks (QGANs) [22], [23], which aims to push forward generative learning. Currently, QGANs are still in its infancy, and many proposed frameworks deal with low-dimensional data, such as simple probability distributions [24], [25], [26]. In the realm of image generation, QGANs built using quantum generators have only been able to generate low-resolution images in [27] or require dimensionality reduction with principal component analysis (PCA) [28], [29]. Also, there exists an important knowledge gap in QGANs on how varying different parameters within the quantum generator affects the performance and output quality of QGANs. In previous works, the evaluation of the QGANs are conducted on low-dimensional data, where the images have either been compressed to a lower dimensional space in [28] and [29], or are from a synthetic 2×2 pixels dataset in [27]. This restricts their application for realistic problems and the scope of the acquired understanding is also limited.

In this article, we aim to bridge the gap between classical and quantum GANs by generating high-dimensional data in the form of images. Specifically, we propose a new hybrid quantum–classical framework as shown in Fig. 1, which we call the patch quantum Wasserstein GAN (PQWGAN). Our framework leverages the theoretical benefits that the

WGAN-gradient penalty (GP) [30] brings to improve the patch strategy QGAN [27]. The patch strategy QGAN splits the output image generated into different patches, each generated by a separate quantum circuit. This is useful in the NISQ era, where splitting up the output can reduce the quantum resources required. On the other hand, the WGAN-GP is an extension to the WGAN that has improved convergence properties owing to the use of a GP for regularization. Individually, the patch QGAN with the GAN framework [27] and the WGAN-GP framework with a single PQC quantum generator (see Section VII-A) are both unable to generate high-resolution images. However, combining these two ideas our PQWGAN is capable of generating high-resolution images without dimensionality reduction, which was not possible using previous QGANs directly.

First, we numerically demonstrate the viability of the framework by applying it to learn to generate full resolution 28×28 pixels images from the standard Modified National Institute of Standards and Technology (MNIST) [31] and Fashion MNIST (FMNIST) [32] datasets. To the best of authors' knowledge, this is the first demonstration of a QGAN that uses quantum circuits as a generator that can successfully generate images without dimensionality reduction or classical pre/postprocessing at this scale. Second, we gain a deeper understanding of the effects that varying different quantum generator parameters have on output quality by experimenting directly on the 28×28 pixels images. Parameters explored include the number of patches, qubits, layers, shape of patches, and choice of prior. Simulations provide the crucial insight that increasing the generator size in general correlates with better output quality. Our results demonstrate that our framework has the potential to serve as a foundation for future QGAN research on more complex tasks.

The rest of this article is organized as follows. First, we go over some preliminaries in Section II and related work on QGANs in Section III. Next, we introduce our novel PQWGAN framework in Section IV and our experimental setup in Section V. Then, we put our PQWGAN to the test by applying it to generate images of MNIST and FMNIST in Section VI and evaluating the effects of different parameters in Section VII. Finally, Section VIII concludes this article.

II. PRELIMINARIES

A. GENERATIVE ADVERSARIAL NETWORKS

GANs were first proposed in [1]. The framework consists of a discriminator and a generator that compete in an adversarial game. The generator and discriminator can in theory be any machine learning model, but are commonly neural networks due to its empirical success. The generator G takes an input noise vector z sampled from some distribution P_z (e.g., Gaussian) and produces an output. The goal is to have the learned distribution P_G match the real distribution P_{data}. On the other hand, the discriminator D takes an input $\boldsymbol{x}$ and outputs the probability it believes $\boldsymbol{x}$ originated from P_z. If the probability is greater than one half, the input is classified as originating from the real data and vice versa. The goal of D is to maximize the probability of assigning the correct labels, whereas the goal of G is to produce samples that confidently fools D. The objective of the GAN training can be expressed in terms of a zero-sum game

$$\min_G \max_D \mathbb{E}_{\boldsymbol{x}\sim P_{\text{data}}}[\log D(\boldsymbol{x})] + \mathbb{E}_{z\sim P_z}[\log(1 - D(G(z)))]. \quad (1)$$

B. WASSERSTEIN GENERATIVE ADVERSARIAL NETWORKS

As briefly mentioned in Section I, GANs suffer from a variety of problems during training due to the mathematical properties of minimizing the JS divergence. To mitigate these issues, there have been efforts to reformulate GAN training with completely different objectives in order to obtain better theoretical guarantees. One of the most successful framework is the WGAN [7]. The WGAN minimizes the Wasserstein distance, and the value function of the WGAN is

$$\min_G \max_{D\in\mathcal{D}} \mathbb{E}_{\boldsymbol{x}\sim P_{\text{data}}}[D(\boldsymbol{x})] - \mathbb{E}_{z\sim P_z}[D(G(z))] \quad (2)$$

where $\mathcal{D}$ is the set of 1-Lipschitz functions. Instead of having a discriminator that produces a binary output as in GANs, the discriminator now outputs a score, which is interpreted as the Wasserstein distance between P_G and P_{data}. Hence, the discriminator is known as a critic instead. Minimizing the Wasserstein distance exhibits much nicer theoretical guarantees than minimizing the JS divergence, and is shown to converge in many instances where the JS divergence fails to do so [7]. First, the gradient of the critic with respect to the input is much better behaved than that of the discriminator in GANs, allowing the generator to be trained more easily and the critic to be trained to optimality without having to deal with vanishing gradients. Next, the WGAN has shown empirical evidence of being able to avoid mode collapse, as the authors were able to train various discriminator and generator architectures that were previously hard to train successfully. Also, since the WGAN value function provides an estimate of the Wasserstein distance, it has empirically observed to be correlated with sample quality of the generator. Hence, this can be used as a stopping condition in WGAN training.

To enforce the 1-Lipschitz constraint, Arjovsky et al. [7] proposed to clip the gradients of the each critic parameter within a fixed range such as $[-0.01, 0.01]$. However, the choice of the clipping range poses new problems. If the gradient magnitude is large, it can take a long time for the critic to reach optimality, but if the magnitude is small it can also easily lead to vanishing gradients. Instead, Gulrajani et al. [30] proposed the WGAN-GP, where a GP is used instead to enforce the 1-Lipschitz constraint. The new value function is

$$\min_G \max_{D\in\mathcal{D}} \mathbb{E}_{\boldsymbol{x}\sim P_{data}}[D(\boldsymbol{x})] - \mathbb{E}_{z\sim P_z}[D(G(z))] - \lambda\mathbb{E}_{\hat{\boldsymbol{x}}\sim P_{\hat{x}}}[(||\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})||_2 - 1)^2] \quad (3)$$

where λ is a constant and $P_{\hat{x}}$ is a distribution sampled uniformly in between P_{data} and P_G. In the WGAN framework, the optimal critic has unit gradient norm for straight lines between P_{data} and P_G. Hence, by enforcing this condition in (3), the critic is able to be trained to optimality without vanishing gradients. This is supported by the fact that the WGAN-GP framework was applied to successfully train many random variations of the DCGAN architecture, such as having different activation functions, depth, use of batch normalization and filter count [30]. Compared with the GAN framework, the WGAN-GP framework is able to successfully train a significantly larger portion of these random architectures to some minimum Inception score [33] (which quantitatively measures the output variety and quality of a GAN) on the 32×32 pixels ImageNet dataset.

III. RELATED WORK ON QGANS

The notion of a QGAN was first introduced theoretically in [22], and demonstrated to be viable numerically in [23]. In general, QGANs can take in either quantum or classical data. For classical data, Lloyd and Weedbrook [22] claimed that although there are no guarantees for quantum advantage, it is reasonable to expect that the quantum GAN can learn the data distribution in less time due to efficient quantum algorithms to solve linear equations, such as the Harrow–Hassidim–Lloyd algorithm [34]. However, early methods focus on generating relatively simple low-dimensional distributions. As such, many of the proposed QGAN frameworks provide limited use for high-resolution image generation in the NISQ era.

A. QGANS FOR IMAGE GENERATION

Given a quantum computer with n qubits and the task of generating an M-dimensional output, Huang et al. [27] suggested the batch and patch strategies for QGANs in the NISQ era. The patch strategy is useful for the case where $n < \lceil \log M \rceil$, which is likely the case in higher resolution image generation on NISQ devices. In the patch QGAN, the generator is composed of k quantum circuits that are each sampled to generate a patch of the image, whereas the discriminator can be either a classical or quantum classifier. The resulting patches are then stitched together to form the final image. The patch QGAN was able to generate 8×8 pixels images of handwritten digits of 0 s and 1 s by training on the optical recognition of handwritten digits dataset [35] on both simulations and a superconducting quantum computer. Although this approach successfully generated images of handwritten digits, the quality of the images were quite low.

Another method explored to generate images is the state fidelity-based QGAN (QuGAN) [28]. The core of the framework is a swap test to measure the fidelity between the discriminator and generator state for the loss function. Recently, the IQGAN [29] was proposed as an extension to the QuGAN framework. It features a new trainable classical to quantum encoder to embed classical data and a more compact quantum generator that avoids costly two qubit gates. Both frameworks carried out experiments using simulations and real devices on a subset of the MNIST dataset compressed using PCA, and were both able to successfully generate the target images. However, due to the use of an inverse PCA to generate the images from a low-dimensional representation, the images are often quite blurry. Also, since the diversity of the output only stems from the randomness when doing a finite measurement on the generator state, it may be difficult to scale to more complex tasks, such as having more digits.

B. QUANTUM WASSERSTEIN GENERATIVE ADVERSARIAL NETWORKS

The idea of a fully quantum version of the WGAN (qWGAN) was proposed in [36] and [37]. Both of these works were concerned with the task of learning to generate pure and mixed states using the Wasserstein distance with PQCs. In both cases, simulations showed that the frameworks are able to learn the target states and converge to a high fidelity quickly. However, since their framework is designed to work on quantum data, it cannot be directly applied to image generation, which is what we are interested in.

Another extension of the WGAN is the QWGAN-GP [38], which considered a hybrid quantum–classical version of the WGAN-GP. In this framework, the generator is a single PQC that takes a latent vector as input, while the remaining components of the QWGAN-GP are identical to the WGAN-GP. Experiments on the credit card fraud dataset [39] showed that the QWGAN-GP has comparable performance to a fully connected WGAN-GP architecture on anomaly detection while having less trainable parameters. However, the results indicate that the dataset is too simple, as both the classical and quantum networks converge to the optimum with a low-dimensional latent vector and low depth for the generators.

C. COMPARISON OF OUR WORK TO EXISTING QGANS

A summary of the related works on image generation with QGANs mentioned in Section III-A can be found in Table I.

IV. PQWGAN FRAMEWORK

In this section, we present the PQWGAN framework for generating high-resolution images on NISQ devices. A comparison of our work to existing QGANs for image generation is shown in Appendix A. The PQWGAN integrates the patch method for image generation on NISQ devices [27] and the WGAN-GP [30]. We choose to use WGAN-GP due to its improved convergence properties compared with weight clipping. Furthermore, during initial explorations with QGANs, we noticed that when applying the GAN loss function as in (1), the QGANs exhibited unstable behavior, and were hard to train. Hence, this challenge further motivated us to use a theoretically stable method, such as WGAN-GP. In our case, the setup is the same as in WGAN-GP, but instead of a classical generator, we replace it with a patch quantum generator as in [27]. The overall architecture of the PQWGAN is shown in Fig. 1.

TABLE 1 Comparison of Our Work to Existing QGANs for Image Generation

Paper	Dataset(s)	Output size	Maximum number of classes	Notes
This work	MNIST, FMNIST	28×28 pixels	3	Does not require pre/post-processing
Experimental Quantum Generative Adversarial Networks for Image Generation [27]	Handwritten digits	8×8 pixels	2	Low image quality
QuGAN: A Quantum State Fidelity based Generative Adversarial Network [28]	MNIST	4 dimensions	3	Uses PCA to compress images
IQGAN: Robust Quantum Generative Adversarial Network for Image Synthesis On NISQ Devices [29]	MNIST	16 dimensions	3	Uses PCA to compress images

Algorithm 1: Algorithm to Generate an Image From the Patch Quantum Generator.

```
Input: Image dimensions $H \times W$, number of ancilla qubits
       $A$, number of data qubits $D$, number of
       sub-generator layers $L$, number of patches $P$,
       generator parameters $\boldsymbol{\theta} = [\theta_1, ..., \theta_P]$, latent
       variable $\boldsymbol{z}$.
1 for $i = 1, ..., P$ do
2     $|\psi_i\rangle \leftarrow \mathcal{U}_{i,L,\theta_i}\,|\boldsymbol{z}\rangle$
3     $\rho_D \leftarrow \mathrm{Tr}_A\left(\frac{(|0\rangle\langle 0|)^{\otimes A}\otimes\mathbb{I}|\psi_i\rangle\langle\psi_i|}{\langle\psi_i|(|0\rangle\langle 0|)^{\otimes A}\otimes\mathbb{I}|\psi_i\rangle}\right)$
4     measure $\rho_D$ in computational basis to obtain
        $G_i(\boldsymbol{z}) \leftarrow [p(0), ..., p(2^D - 1)]$
5     $G_i'(\boldsymbol{z}) \leftarrow \frac{G_i(\boldsymbol{z})}{\max(G_i(\boldsymbol{z}))}$
6     discard excess pixel values to obtain
        $G_i''(\boldsymbol{z}) \leftarrow G_i'(\boldsymbol{z})[:\frac{HW}{P}]$
7 $G(\boldsymbol{z}) \leftarrow [G_1''(\boldsymbol{z}), ..., G_P''(\boldsymbol{z})]^T$
8 return $G(\boldsymbol{z})$
```

A. STRUCTURE OF QUANTUM GENERATOR

Algorithm 1 shows the process of generating an image using the quantum generator. The inner workings of our quantum generator is very similar to that of [27] and is detailed here, with the schematic diagram plotted in Fig. 2. The generator first takes in a N-dimensional latent vector $z = (z_1, z_2, \ldots, z_N)$ from some distribution p_z (e.g., uniform, Gaussian). The latent vector is then encoded in each subgenerator using a layer of R_Y rotations parameterized by the components of z. So, starting in the $|0\rangle^{\otimes n}$ state, we obtain the latent state $|z\rangle$ by applying the encoding circuit

$$|z\rangle = R_Y^1(z_1)R_Y^2(z_2)\ldots R_Y^N(z_N)|0\rangle^{\otimes N}$$

where $R_Y^i(z_i)$ is the R_Y gate applied to the ith qubit with the rotation angle z_i. Then, the latent state passes through the L layers of the hardware-efficient ansatz structure [41], each consisting of parameterized arbitrary rotations $R(\phi, \theta, \omega)$ followed by CNOT gates on each adjacent qubit to generate entanglement. The $R(\phi, \theta, \omega)$ gate can be expressed as

$$R(\phi, \theta, \omega) = R_Z(\omega)R_Y(\theta)R_Z(\phi)$$

$$= \begin{bmatrix} e^{-i(\phi+\omega)/2}\cos(\theta/2) & e^{-i(\phi-\omega)/2}\sin(\theta/2) \\ e^{-i(\phi-\omega)/2}\sin(\theta/2) & e^{-i(\phi+\omega)/2}\cos(\theta/2) \end{bmatrix}$$

This gate was chosen as it can represent any single-qubit rotation that we want up to a phase shift. Furthermore, it can be easily decomposed as a series of ZYZ gates, which can be implemented on a real device. The L parameterized layers act essentially as one big unitary operation $\mathcal{U}_L(\boldsymbol{\phi_i}, \boldsymbol{\theta_i}, \boldsymbol{\omega_i})$ that performs a linear transformation on the state $|z\rangle$, and the resulting quantum state generated by the ith subgenerator is

$$|\psi_{G_i}\rangle = \mathcal{U}_L(\boldsymbol{\phi_i}, \boldsymbol{\theta_i}, \boldsymbol{\omega_i})|z\rangle.$$

The success of deep learning methods, such as neural networks lies in its ability to learn nonlinear transformations of its input. To introduce nonlinearity into the subgenerators, we make a partial measurement M on the ancilla qubits, then trace out the ancilla qubits to obtain the resulting state of the data qubits. Since we will be making projector measurements, the state-of-the-data qubits $|\psi_D\rangle$ after tracing out the ancilla qubits will be

$$|\psi_D\rangle = \mathrm{Tr}_A\left(\frac{M\otimes\mathbb{I}|\psi_{G_i}\rangle\langle\psi_{G_i}|}{\langle\psi_{G_i}|M\otimes\mathbb{I}|\psi_{G_i}\rangle}\right).$$

In our case, we pick the partial measurement to be $M = (|0\rangle\langle 0|)^{\otimes A}$ for simplicity. Hence, the final state of the data qubits will be

$$\rho_D = \mathrm{Tr}_A\left(\frac{(|0\rangle\langle 0|)^{\otimes A}\otimes\mathbb{I}|\psi_{G_i}\rangle\langle\psi_{G_i}|}{\langle\psi_{G_i}|(|0\rangle\langle 0|)^{\otimes A}\otimes\mathbb{I}|\psi_{G_i}\rangle}\right).$$

The state now depends on $|\psi_{G_i}\rangle$ in both the denominator and the numerator, which is in turn dependent on $|z\rangle$. Hence, the state is a nonlinear transformation of $|z\rangle$. We, then, measure the probability of each computational basis state-of-the-data qubits to obtain the sub-generator output given by

$$G_i(z) = [p(0), p(1), \ldots, p(2^{D-1})]$$

We would like each element of the generator output to have values between [0,1] to be interpreted as pixel values. Although it is possible to interpret the probabilities as pixel values directly, it would be problematic due to the normalization constraint, which would not give us the desired pixel values. Hence, we apply postprocessing by taking

$$G_i'(z) = \frac{G_i(z)}{\max(G_i(z))}$$

in order to obtain valid pixel values. Since the size of the outputs from the quantum circuit are power of 2 s, we only keep the first HW/P pixels to create a patch with the correct dimensions. Finally, the output from all the subgenerators are stacked together to form an image of size $H \times W$

$$G(z) = [G_1'(z), \ldots, G_P'(z)]^T.$$

It is possible to use other more complex nonlinear transformations from classical machine learning, such as passing the output through different activation functions as discussed in [27]. However, due to time constraints we deemed it to be out of scope.

B. STRUCTURE OF CRITIC

The critic in this case is the same as in WGAN-GP, which is a classical neural network. We believe that a classical neural network would be better for the PQWGAN framework as it is developed for NISQ devices in mind. Recall that the critic is responsible for taking in an image and outputting a real value that serves as an estimate for the Wasserstein distance. This poses several problems for a quantum critic. First, we would have to load high-dimensional data into a quantum circuit, which is hard to do in practice due to the amount of quantum resources required. Second, the learning process of quantum circuits are not as well understood as classical neural networks. There are limited solutions to problems that are frequently encountered in quantum learning, such as barren plateaus [42].

C. TRAINING OBJECTIVE

We adopt the objective of WGAN-GP defined in (3) to train the PQWGAN. A subtle difference to WGAN-GP is that the output from the generator originates from multiple subgenerators as detailed in Section IV-A. The use of this objective is motivated by the empirical observation that WGAN-GP can be used to train a variety of different architectures successfully with minimal hyperparameter tuning. We view this as an important benefit for the NISQ era since quantum resources are scarce. Although quantum models can be prototyped using quantum simulators, the difficulty of simulating quantum circuits means that searching for optimal hyperparameters is a time-consuming and tedious task. Until we can more efficiently execute quantum circuits, fine-grained hyperparameter tuning will most likely be out of reach for larger models. Hence, this objective will in principle allows us to train QGANs in general with a greater success rate.

D. TRAINING ALGORITHM

The training algorithm for PQWGAN follows the WGAN-GP training algorithm, except for the use of quantum generators (see Algorithm 2). With the generator now being split into subgenerators, we have to update the parameters of different subgenerators given a loss that is calculated on the entire images. Furthermore, the loss function $\mathcal{L}(\boldsymbol{w}, \boldsymbol{\theta})$ depends on both the critic and generator parameters, respectively. Since the training is done in alternating steps, and either the critic or generator is assumed to be fixed while training, we can still use the parameter shift rule [44] to compute the gradient. Assuming we have N_G subgenerators with n parameters each, the generator's parameters can be expressed as a vector $\boldsymbol{\theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{N_G}] = [\theta_{1,1}, \ldots, \theta_{1,n}, \ldots, \theta_{N_G,1}, \ldots, \theta_{N_G,n}]$. Hence, the gradient of the jth parameter of the ith subgenerator with respect to the loss is

$$\frac{\partial \langle \mathcal{L}(\boldsymbol{w}, \boldsymbol{\theta}) \rangle}{\partial \theta_{i,j}} = \frac{1}{2}(\langle \mathcal{L}(\boldsymbol{w}, [\theta_{1,1}, \ldots, \theta_{i,j} + \pi/2, \ldots, \theta_{N_G,n}]) \rangle - \langle \mathcal{L}(\boldsymbol{w}, [\theta_{1,1}, \ldots, \theta_{i,j} - \pi/2, \ldots, \theta_{N_G,n}]) \rangle).$$

Algorithm 2: PQWGAN Training Algorithm.

```
Input: Gradient penalty coefficient $\lambda$, critic iterations per
       generator iteration $n_C$, number of epochs $n_{epochs}$,
       batch size $m$, Adam hyperparameters $\eta_1, \eta_2, \beta_1, \beta_2$.
1  initialise critic parameters $\boldsymbol{w}$, sub-generator parameters $\boldsymbol{\theta}$
2  for $epoch = 1, ..., n_{epochs}$ do
3      for $t = 1, ..., n_C$ do
4          for $i = 1, ..., m$ do
5              Sample real data $\boldsymbol{x} \sim \mathbb{P}_{data}$, latent variable
                $\boldsymbol{z} \sim p_{\boldsymbol{z}}$, random number $\epsilon \sim U[0,1]$
6              $\boldsymbol{x}' \leftarrow quantum\_generator(\boldsymbol{\theta}, \boldsymbol{z})$
7              $\hat{\boldsymbol{x}} \leftarrow \epsilon \boldsymbol{x} + (1-\epsilon)\boldsymbol{x}'$
8              $L_D^{(i)} \leftarrow D(\boldsymbol{x}') - D(\boldsymbol{x}) + \lambda(||\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})||_2 - 1)^2$
9          $\boldsymbol{w} \leftarrow$ Adam $(\nabla_{\boldsymbol{w}} \frac{1}{m} \sum_{i=1}^{m} L_D^{(i)}, \boldsymbol{w}, \eta_1, \beta_1, \beta_2)$
10     for $i = 1, ..., m$ do
11         Sample latent variable $\boldsymbol{z} \sim p_{\boldsymbol{z}}$
12         $\boldsymbol{x}' \leftarrow quantum\_generator(\boldsymbol{\theta}, \boldsymbol{z})$
13         $L_G^{(i)} \leftarrow -D(\boldsymbol{x}')$
14     $\boldsymbol{\theta} \leftarrow$ Adam $(\nabla_{\boldsymbol{\theta}} \frac{1}{m} \sum_{i=1}^{m} L_G^{(i)}, \boldsymbol{\theta}, \eta_2, \beta_1, \beta_2)$
```

V. EXPERIMENTAL SETUP

A. DATASET AND LIBRARIES

We pick the publicly available MNIST [31] and FMNIST [32] datasets to conduct our experiments on. Although MNIST and FMNIST are simple datasets for classical GANs, they are still a considerable step-up in terms of complexity to previous studies of QGANs especially with the full 28×28 resolution. Due to resource limitations, we will only be using the first 1000 samples of every class that we include in our training set to ensure that the training process can be carried out in a reasonable amount of time.

All models are implemented in Python3 using PyTorch [46] and PennyLane [47]. PyTorch is a high-performance machine learning library, whereas PennyLane is a QML library that provides interfaces to PyTorch. The training of all PQWGANs are simulated without noise using high-performance computing resources from the National Computational Infrastructure, Pawsey and the University of Melbourne.

B. CLASSICAL CRITIC AND GENERATOR STRUCTURES

To systematically investigate the performance of the quantum generator, we fix the critic in the PQWGAN to be the same in all of our experiments. The critic is a fully connected network with two hidden layers of 512 and 256 neurons, respectively. Both of these hidden layers have a leaky rectified linear unit (ReLU) activation with a slope of 0.2. The final hidden layer is connected to an output layer of one neuron with no activation to obtain a real-valued output.

To contrast our framework with classical GANs, we compare our results with a WGAN-GP. For consistency, the critic used is the same as in the PQWGAN. The generator is now also a fully connected network with three hidden layers of 256, 512, and 1024 neurons, respectively. Again, the hidden layers all have leaky ReLU activations with a slope of 0.2. Finally, the hidden layers are connected to an output layer consisting of the same number of neurons as the output pixels with a tanh activation, which are then rearranged to form an image.

Since our experiments are conducted on relatively simple datasets, we opted for a simple architecture across all classical components of our experiments. To validate the capability of the classical parts in learning, we applied the WGAN-GP to learn the full MNIST and FMNIST datasets. In both datasets, the Wasserstein distance converges toward 0, while manually inspecting the outputs confirmed that the generator is indeed learning successfully. This shows that our classical components should not affect the learning capabilities of the PQWGAN.

C. HYPERPARAMETERS

Unless otherwise specified, the hyperparameters for all our experiments are chosen as follows. We follow the default values for the learning process in WGAN-GP [30], where we use the values $\lambda = 10, n_C = 5$, and Adam [48] for optimization with hyperparameters $\beta_1 = 0$ and $\beta_2 = 0.9$. We decided on having 28 subgenerators generating 28 patches, so that one patch would correspond to one row of pixels in the image. Each subgenerator also has one ancilla qubit. Furthermore, after some hyperparameter tuning, we found that the learning rate for the quantum generator needs to be higher than the classical critic to learn, and we set the learning rate to be 0.01 and 0.0002 for the generator and critic, respectively. Also, in our initial explorations the quantum generator was observed to learn quicker when using a uniform prior, so we chose a uniform prior over a Gaussian prior. The uniform prior is restricted to be in the range $[0, 1)$ instead of $[-\pi, \pi)$. Although using the latter can cover the whole range of possible rotations in the quantum circuit, we found that it led to poorer learning due to the larger space that the generator has to learn from.

Due to the time required to simulate the quantum circuits, we use a batch size of 25 to ensure that the generator is sufficiently updated during the training process. In all our experiments, we train the generator for 600 iterations, which is equivalent to processing 3000 batches of data in total. Depending on the number of classes used in our experiments, this corresponds to 37.5 or 25 epochs for the two and three class experiments, respectively. We pick this number as it is a considerable number of epochs for training under current resource constraints while also being able to be completed in a reasonable amount of time.

VI. IMAGE GENERATION WITH PQWGAN

In this section, we apply our PQWGAN framework to generate images of MNIST and FMNIST. To compare between classical and quantum learning, we repeat each task with a fully classical WGAN-GP with identical hyperparameters except for the use of a Gaussian prior and the learning rate, where it was a constant 0.0002 for both the generator and critic. These exceptions were made to ensure that we are not severely crippling the learning ability of the WGAN-GP. We also sampled from a latent space that has the same number of dimensions as available in the quantum case to ensure that the WGAN-GP cannot exploit extra dimensions in the latent space. Our results are shown in Fig. 3.

A. BINARY MNIST

First, we applied the PQWGAN to generate digits 0 and 1 from MNIST. In this task, each subgenerator consists eight layers of seven data qubits and one ancilla qubit to conserve resources. As such, the latent vector has eight dimensions.

Looking at Fig. 3(a) and (b), it is evident that for both the classical and quantum case, the generators are successfully learning to generate images of 0 s and 1 s. Furthermore from Fig. 3(d), we can see that the Wasserstein distance slowly decreases as the training progresses, which corresponds to an increase in image quality when we inspect the generated images during the training process. However, there are still imperfections in both cases, which allows them to be easily distinguished from the real images. In particular for the quantum case, there are some samples that resemble neither a 0 nor 1 and appear to be a combination of the two, such as row 3 column 3 in Fig. 3(a). These mixed images can be attributed to the incomplete learning process when the generator has yet not learned a comprehensive mapping for the entire latent space. Due to the limitations of running quantum simulations, 600 generator iterations is a very small amount compared with experiments conducted in the classical case, where the number is around 10^4–10^5 generator iterations albeit for a more complex task. Furthermore, in classical GAN training, the generator also outputs samples that look like a combination of the different classes in the early stages of training before slowly learning to diversify into the separate digits. The similar behavior between PQWGAN and WGAN-GP suggests that the problem of having mixed images can be mitigated by training the generator more.

Another imperfection in the quantum case is the fuzziness of the images. Even in images that look plausible, such as row 1 column 3 in Fig. 3(a), we can still see that the edges are not very sharp. Since the pixel values originate from the amplitudes of the quantum state at the end of the circuit, to

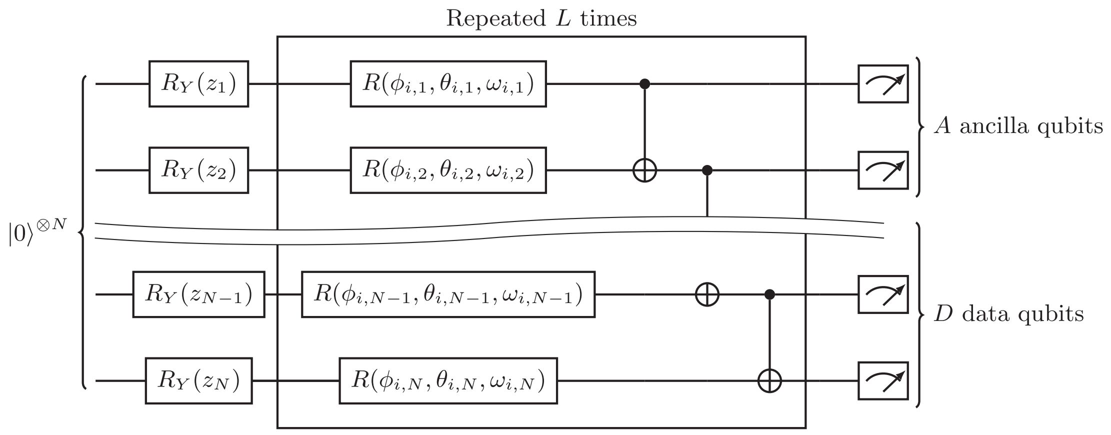

Repeated L times
$|0\rangle^{\otimes N}$
$R_Y(z_1)$
$R_Y(z_2)$
$R_Y(z_{N-1})$
$R_Y(z_N)$
$R(\phi_{i,1}, \theta_{i,1}, \omega_{i,1})$
$R(\phi_{i,2}, \theta_{i,2}, \omega_{i,2})$
$R(\phi_{i,N-1}, \theta_{i,N-1}, \omega_{i,N-1})$
$R(\phi_{i,N}, \theta_{i,N}, \omega_{i,N})$
A ancilla qubits
D data qubits

FIGURE 2. **Quantum circuit of a subgenerator. First, each component of the latent vector is encoded into the rotation angle of a R_Y gate. Then, the state passes through L layers of arbitrary parameterized rotations R and CNOTs using the hardware-efficient structure. The subscripts i, j of the parameters of the R gates refer to the ith layer and the jth qubit. The ancilla qubits are measured to perform a nonlinear transformation on the state. In this article, we use only one ancilla qubit and pick the resulting state where the ancilla is 0. Finally, the data qubits are measured to form a patch.**

generate a completely dark pixel, we require the corresponding amplitude to be 0. However, due to the nature of a highly entangled circuit, it is very difficult for a particular state to have exactly 0 amplitude. This effect is magnified by the postprocessing step, where the amplitudes are normalized to be in [0, 1]. In terms of image sharpness, classical GANs have an advantage since it is easier for the optimization process to change the value of a specific pixel.

The fact that the PQWGAN can generate images of comparable quality to WGAN-GP further supports the result that PQCs have a stronger expressive power than those of classical neural networks [19]. For our WGAN-GP architecture, the generator consists of roughly 1.46 million trainable parameters, whereas our quantum generator consists of 5376 trainable parameters. Yet, our quantum generator is able to keep up with the classical generator. In comparison, a classical generator with a similar parameter count is unable to learn anything meaningful. In the future when the technology is sufficiently developed, it would be interesting to investigate how larger scale QGANs can compare with classical GANs in terms of performance.

B. BINARY FMNIST

Next, we investigate whether the PQWGAN can learn from and generate more complex data, namely images from the classes of T-shirt and trousers from FMNIST. In this experiment, each of the subgenerators now has 11 layers to accommodate the increase in the data complexity. We keep the structure of having seven data qubits and one ancilla qubits and observed that it worked well.

The results in Fig. 3(e) and (f) show that again for both the classical and quantum case, the generators can successfully learn to generate images of T-shirts and trousers. However, the problems of samples being uncertain and fuzzy also persist in this case. With the increased complexity of the task, it is expected that these problems will be more apparent as the generator now has a harder time of generating a sharp image with more detail. Still, we observe that our PQWGAN is attempting to learn more subtle details, such as different shades and thickness of legs in the trousers of Fig. 3(e). This shows that there is potential for quantum generators to learn from more complex images in the future.

Looking at the training curve for the PQWGAN in Fig. 3(h), the Wasserstein distance is not decreasing much and has a high variation as it gets updated. This suggests that the generator is nearing its capacity, and is failing to learn a representation that can capture all the variations in details of the samples it is generating due to its limited expressiveness. To rectify this problem, we could increase the number of layers in each subgenerator to increase the expressiveness of the generator. Furthermore, we could also increase the batch size of the learning process to obtain more stable gradients. However, both these methods are expensive to simulate as they increase the amount of resources required to simulate the circuits and are out of the scope for this work.

C. TRIPLE MNIST

After investigating the performance of our framework for image generation in two classes of a dataset, we now investigate whether our framework can be applied to generate more classes simultaneously. From Section VI-B, we observed that the generator struggles to properly learn the more complex and diverse features on FMNIST with its current size. Hence, we focused on the task of learning on three classes of MNIST. We selected the digits 0, 1, and 3 to learn as they have a distinct structure to them. In this case, we use 11 layers per subgenerator of seven data qubits and one ancilla qubit.

Our results show that both the classical and quantum frameworks are able to generate images that correspond to the three digits. However, for the PQWGAN, there are now artifacts that persist in the same location of every output image. The reason for such simulation results is not fully known. However, it is empirically observed that varying certain parameters, such as the number of data qubits in the

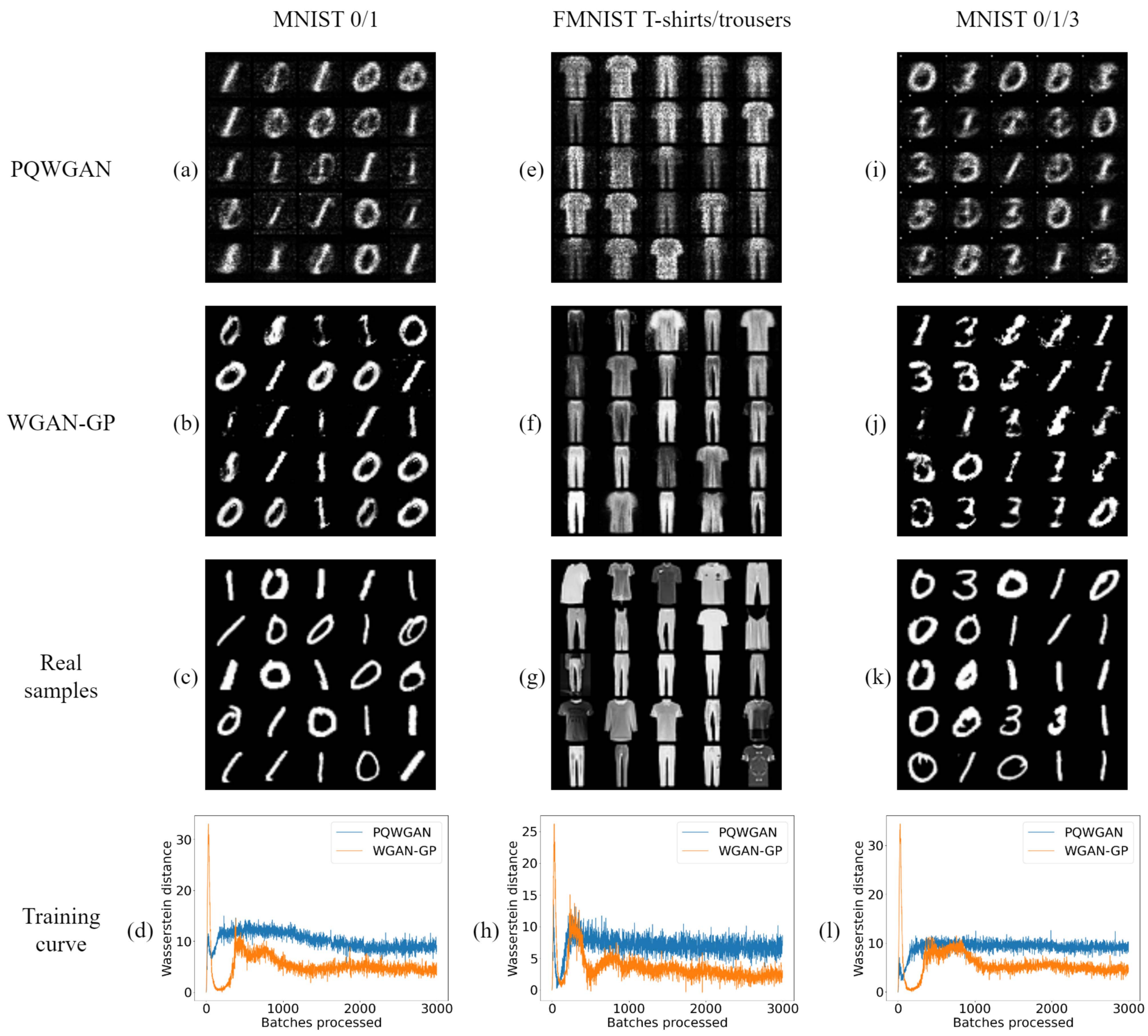


FIGURE 3. Random images generated using PQWGAN and WGAN-GP. The images generated by our PQWGAN are of comparable quality to that using a WGAN-GP albeit having orders of magnitude less trainable parameters. Especially, for MNIST 0/1 and FMNIST T-shirts/trousers, the PQWGAN is able to keep up with WGAN-GP in terms of image sharpness. However, for MNIST 0/1/3 both the PQWGAN and WGAN-GP start to struggle.

quantum generator reduces the impacts of these artifacts. This will be further explored in Section VII.

Ignoring these artifacts, both the classical and quantum frameworks are able to learn to generate images of 0, 1, and 3. However, there is an increased proportion of mixed images in PQWGAN when compared with Section VI-A. This is expected as the dataset is now more complex, with fewer epochs. Furthermore, the Wasserstein distance as shown in Fig. 3(l) is plateauing, which suggests that the generator is nearing its capacity. Hence, we require more layers and a longer training process to better learn from triple MNIST.

D. WALKING IN THE LATENT SPACE

In classical GANs, interpolations between two points in the latent space have been used to demonstrate that the generator has learned to output a smooth mapping instead of memorizing specific samples [2]. Here, we perform linear interpolation to visualize the mapping that the PQWGAN has learned. There exists more complex interpolation methods, such as spherical linear interpolation [49] for high-dimensional latent spaces ($>$ 50 dimensions). However, as the dimensions of our latent space is equal to the number of qubits in a subgenerator, it is of low dimensions ($<$ 10 dimensions) due to resource constraints. Furthermore, since we are only interested in verifying that the generator has learned a smooth mapping for now, we use linear interpolation as it is more intuitive.

We pick two latent vectors corresponding to two well-defined images and divide the straight line connecting them into ten equal segments. Then, we use the points that lie on the ends of these segments to generate outputs. The results are shown in Fig. 4. For all our experiments, including both the two and three class experiments, the generator is able to learn a smooth mapping for points in the latent space.

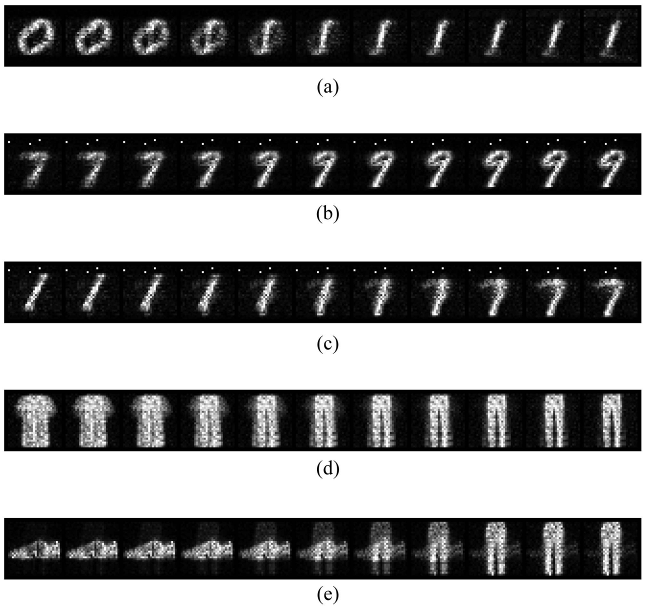


FIGURE 4. Linear interpolation on PQWGAN. Performing linear interpolation between two points of the PQWGAN shows a smooth transition. This shows that our PQWGAN can learn a smooth mapping from the latent space to the output space. (a) 0–1. (b) 7–9. (c) 1–7. (d) T-shirt–trousers. (e) Sneakers–trousers.

This shows that our generator is indeed capable of learning a meaningful mapping from the latent space to the output space.

VII. EFFECTS OF GENERATOR PARAMETERS

To this end, we showed that our framework can be successfully applied to generate high-resolution images of both MNIST and FMNIST, we now turn to investigate how the structure of the quantum generator will affect the quality of the images generated to provide guidance on how to pick the generator architecture. Since there are many different choices that one can make for the generator to generate an image, we choose a set of parameters that we believe has a significant impact on the learning process. Namely, we investigate how altering the number of patches, qubits and layers in the generator, the shape of the patches, and the choice of prior distribution affects the resulting image quality. In each experiment, we vary the parameter in question and fix all other parameters. The training curves obtained from these experiments can be found in Appendix C.

A. NUMBER OF PATCHES

First, we investigate how the number of patches affects the quality of our generated samples. To focus on the effects of the number of patches, we use one ancilla qubit and the minimum number of data qubits required for the patch size in every subgenerator. Furthermore, we adjust the number of layers per subgenerator to keep a similar number of trainable parameters so that the expressiveness of the generator stays relatively constant. We pick the structure with 28 subgenerators and ten layers as a baseline, then vary the number of layers with the number of batches accordingly. Our various generator structures are given in Table 2, and we applied these generators to learn from MNIST 0/1 and FMNIST T-shirt/trousers.

TABLE 2 Various Generator Structures Used to Investigate the Effects of Different Number of Patches

Number of patches	Number of data qubits	Number of Layers	Total number of parameters
1	10	153	5049
2	9	84	5040
4	8	47	5076
7	7	30	5040
14	6	17	4998
28	5	10	5040

From Fig. 5, the results show that there is a significant effect in terms of the number of patches on the quality of the outputs. Starting with only one patch, the generator fails to output anything meaningful. Instead if we inspect intermediate outputs during the training process, the outputs oscillate between being dark and something that resembles a mode collapse. As we increase the number of patches, the images become increasingly sharper, and there are fewer mixed images.

The drop in the output quality as we decrease the number of patches is likely due to the effects of barren plateaus. It has been shown that for a hardware-efficient ansatz, training with a global cost function exhibits barren plateaus regardless of circuit depth [50]. In our framework, the cost function is global as we are directly comparing the final state of the subgenerators to images, which can be thought of as generated from some arbitrary state. As we increase the number of patches, the effect of barren plateaus decreases due to having smaller quantum circuits in the subgenerators. Hence, the subgenerators are able to more effectively explore the Hilbert space, which allows it to quickly search for a mapping. This is evident from the training curves, where the Wasserstein distance converges to a lower value and also has less variance as we increase the number of patches in both experiments.

On the other hand, artifacts do not exist when we have fewer patches for the MNIST experiments. Hence, using less patches may be useful for avoiding artifacts. However, the quality of the outputs decreases significantly as we decrease the number of patches. Hence, having more patches will in general corresponds to better output quality.

B. NUMBER OF QUBITS

Next, we investigate how the number of qubits affects the quality of our generated outputs. We focus on varying the number of data qubits used to generate a patch by increasing the number of data qubits while keeping the same number of patches and discarding an increasing number of pixels. Specifically, we focus on the architecture with 28 patches in our subgenerator, and have 5–8 data qubits while having

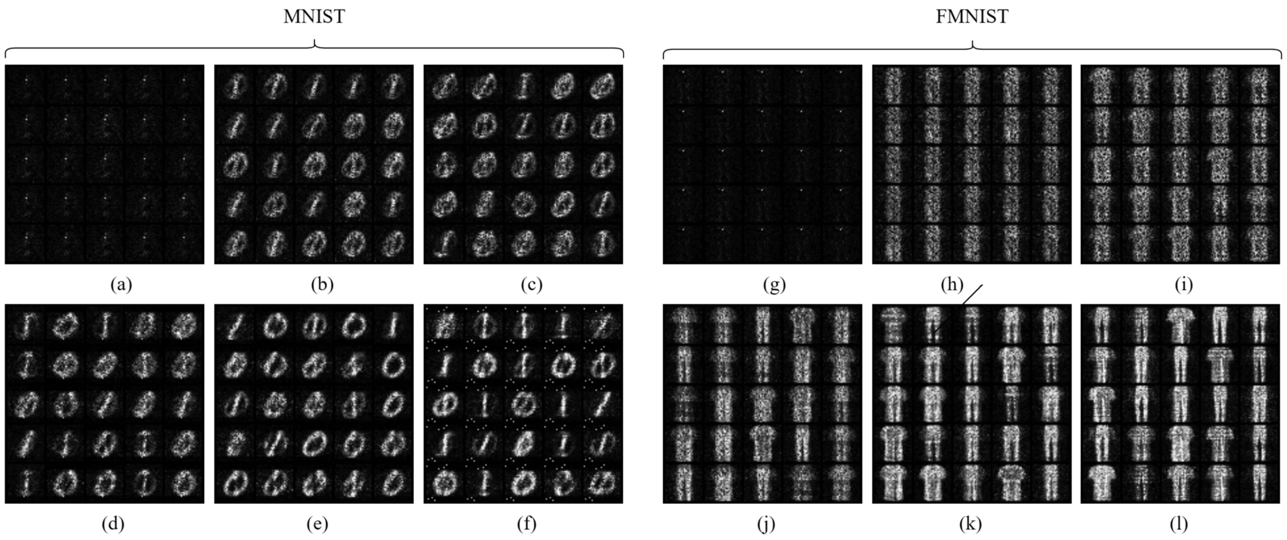


FIGURE 5. **Effect of number of patches. Varying the number of patches without changing other parameters will change the expressiveness of the generator by adding more trainable parameters. Hence, in our experiments we fixed the number of trainable parameters to be roughly constant and varied the number of data qubits and layers accordingly. The images show random outputs sampled from generators with varying number of patches trained on MNIST and FMNIST. In both cases, having a small number of patches led to poor learning. As we increase the number of patches, the generated images become sharper. MNIST. (a) 1 patch. (b) 2 patches. (c) 4 patches. (d) 7 patches. (e) 14 patches. (f) 28 patches. FMNIST. (g) 1 patch. (h) 2 patches. (i) 4 patches. (j) 7 patches. (k) 14 patches. (l) 28 patches.**

TABLE 3 **Various Generator Structures Used to Investigate the Effects of Different Number of Data Qubits**

Number of data qubits	Number of Layers	Number of parameters
5	15	7560
6	13	7644
7	11	7392
8	10	7560

one ancilla qubit. Again, to keep the expressiveness of the subgenerator similar, we keep the parameter count roughly the same by varying the number of layers accordingly. The list of generator configurations tested is given in Table 3. Since varying the data qubits may have an effect on the learning capabilities of the subgenerators, we apply these configurations to the more difficult tasks, namely MNIST 0/1/3 and FMNIST trousers/sneakers to obtain a more pronounced effect.

From Fig. 6, the outputs from generators with extra data qubits are in general smoother and less convoluted, which is supported by the fact that the Wasserstein distance converges to a lower value when we increase the number of qubits. With more qubits, the generator has more flexibility as it can utilize the extra data qubits to manipulate the final amplitude into a higher quality image. However, there is a diminishing return in terms of output quality and Wasserstein distance as we continue to increase the number of qubits. From visually inspecting the outputs and training curve included in Fig. 12 from Appendix C, there does not seem to be much benefit going past six qubits in terms of the image sharpness for these two tasks.

In addition to improving the output quality, we observe that having additional qubits helps in reducing the amount of artifacts present in the final outputs. Other than running our framework with more data qubits on these two tasks, we also tried having more data qubits for the experiments in Section VI when trying to achieve better outputs. We observed a similar trend, where in general having more data qubits also reduced the amount of artifacts present in the learned outputs. To investigate this, we inspected the outputs of the quantum circuits of the subgenerators for our two experiments. Since we are running simulations, we can obtain precisely the quantum state produced at the end of the quantum circuits. We observed that there is a large difference between the sum of the probabilities of the basis states that are used and discarded. In both experiments for the five-qubit case, the sum of the probabilities of the used basis states were very often greater than 0.9, whereas for the eight-qubit case it was very often less than 0.1. This suggests that the presence of artifacts is due to the lack of excess qubits for the subgenerators to iron out the imperfections in a highly entangled output state.

C. NUMBER OF LAYERS

We now turn to investigate the effect of having different number of subgenerator layers. Unlike previous experiments, varying the number of layers will directly change the number of parameters, which affects the expressiveness of the circuit. As we add more layers, the quantum circuit can learn more complex transformations of the input quantum state, which should allow the generator to learn a more complex distribution. Hence, to observe a pronounced effect, we again apply it to the more complex tasks of generating MNIST 0/1/3 and

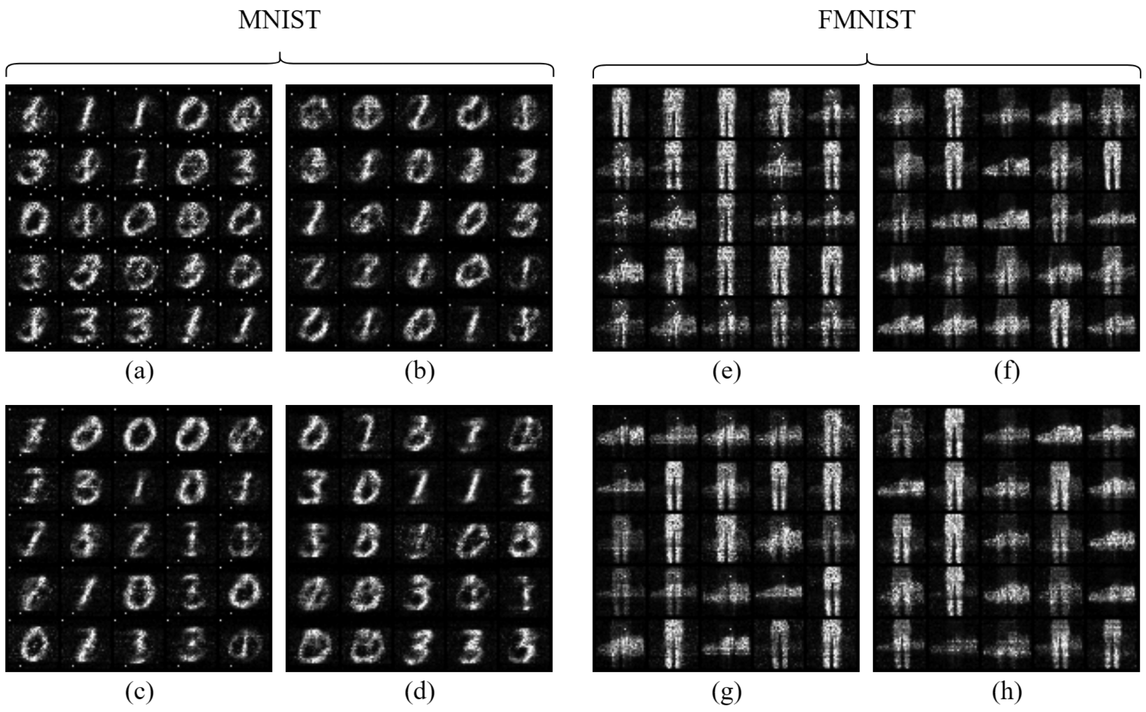


FIGURE 6. **Effect of number of qubits. Varying the number of data qubits without varying the number of layers will change the expressiveness of the generator by adding more trainable parameters. Hence, in our experiments we fixed the number of trainable parameters to be roughly constant by using 28 patches while having a varying number of layers. The images show random outputs sampled from generators with varying number of data qubits trained on MNIST and FMNIST. In both cases as we increase the number of qubits, the outputs become sharper and less convoluted while also leading to less artifacts. MNIST. (a) 5 qubits. (b) 6 qubits. (c) 7 qubits. (d) 8 qubits. FMNIST. (e) 5 qubits. (f) 6 qubits. (g) 7 qubits. (h) 8 qubits.**

FMNIST trousers/sneakers. To be consistent with previous experiments and to conserve resources, we use 28 patches with five data qubits and one ancilla qubit for each generator. We experimented with having 5, 10, 15, 20, and 25 layers of parameterized gates.

From Fig. 7, the quality of the outputs increases with more layers and there are less mixed images. Furthermore, the Wasserstein distance converges to a lower value albeit having greater variance. However, the gain is marginal as we keep increasing the number of layers, and there is not a big improvement as we go past 15 layers. The marginal increase in output quality and the larger variance in Wasserstein distance estimates can be explained by the expressibility of PQCs [51]. As we add more layers to our subgenerators, it can represent a larger set of unitaries $\mathbb{U}$. Hence, as we add more layers to our generator, it becomes increasingly likely that the generator is complete, and that $\mathbb{U}$ contains an acceptable solution. However, the larger set of unitaries also means that the optimization process has to search through a larger space for the solution. Hence, it is likely that the optimization process has to navigate a more complex parameter space, which leads to a higher variance in the Wasserstein distance.

On the other hand, the fact that the Wasserstein distance starts to plateau in all configurations suggests that our choice of ansatz may not be the most suitable for the image generation task. Our choice of the PQC structure is chosen such that it can represent any hardware-efficient ansatz of repeating single qubit rotations followed by CNOT gates. However, there are no theoretical motivations as to why we should use a hardware-efficient ansatz instead of other types of ansatz for image generation. Problem-inspired ansatz, such as the quantum alternating operator ansatz [52] have been shown to be useful in more efficiently implementing solutions to combinatorial optimization problems. Thus, it is possible that there are other types of ansatz that are more suitable for image generation.

D. SHAPE OF PATCHES

Next, we investigate how the shape of the generated patches affects the final outputs. In previous experiments, the data points generated by a patch always fit entirely in one row and are then stacked vertically. In this experiment, we considered having 7×4 patches and stacking them across horizontally. We are now more focused on whether the generator is able to generate images with a different layout of patches, and so we apply it to the easier tasks of generating MNIST 0/1 and FMNIST T-shirts/trousers. Again, we use 28 patches of ten layers with five data qubits and one ancilla qubit for consistency.

From Fig. 8, the shape of the patches does not have a big effect on the final output quality. This is also supported by the training curves, where the evolution of Wasserstein distance is similar in both patch shapes. However, there is a noticeable gap in the Wasserstein distance estimates, which is due to the location of artifacts in the 7×4 patch images. Due to the way the patches are positioned, any artifacts generated using the

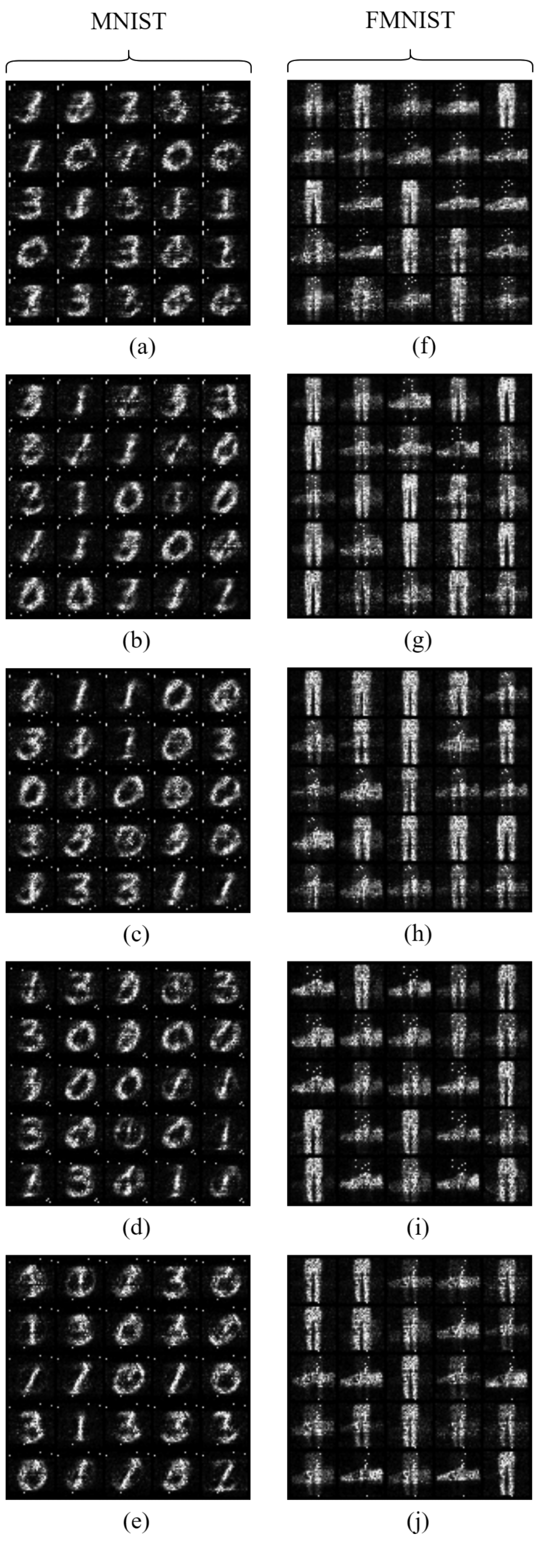


FIGURE 7. Effect of number of layers. In this experiment, we are interested in having a more expressive generator. Hence, we fixed the number of patches and data qubits to be 28 and 5, respectively. The images show random outputs sampled from generators with varying number of layers trained on MNIST and FMNIST. In both cases, increasing the number of layers improves the sharpness of the outputs. However, there is a diminishing return as we keep increasing the number of layers. MNIST. (a) 5 layers. (b) 10 layers. (c) 15 layers. (d) 20 layers. (e) 25 layers. FMNIST. (f) 5 layers. (g) 10 layers. (h) 15 layers. (i) 20 layers. (j) 25 layers.

7×4 patches will appear in center of the final image. Hence, the critic has a very easy time of spotting imperfections that exist, leading to a larger Wasserstein distance estimate.

Ignoring the artifacts, both patch shapes were still able to learn to produce images of the corresponding objects. Given that we already know it works for 1×28 patches, we would expect the generator to be able to successfully learn regardless of the intended shape of its outputs. Initially, we chose having 1×28 patches as it was easier to implement, and did not take into account any of the structure of the images. Since it was able to learn successfully, it showed that the optimization process can successfully guide the patches in producing what it needs for that region. Hence, changing the patch shape would not affect whether the generator can learn to output images. Having said that, changing the patch shape may be useful for more complex datasets with more structure to the images. For example, we can imagine for CIFAR-10 [53] images of natural scenes, it might be beneficial to have patches that corresponds to certain segments of the photo, such as the background, foreground, and objects within.

E. PRIOR DISTRIBUTION

Finally, we investigate the effect of different prior distributions on the outputs. In classical GANs, the prior distribution is usually chosen to be a Gaussian due to its nice mathematical properties and empirical performance. However in our preliminary testing, we observed that the generator only learns to output meaningful samples on some of the inputs from a Gaussian prior. Hence, we opted to use a uniform prior instead. Here, we make a more detailed comparison of the PQWGAN trained on a Gaussian prior and uniform prior. As before, the uniform prior is chosen to be from $U_{[0,1)}$, whereas the Gaussian prior is chosen to be the standard normal $\mathcal{N}(0, 1)$. We applied it to the task of generating MNIST 0/1/3 and FMNIST trousers and sneakers using 28 patches of five data qubits, one ancilla qubit and 15 layers.

From Fig. 9, in both cases the generator with the Gaussian prior has not learned to fully map the latent space to the output space while the uniform prior has, which is due to the small number of training iterations done coupled with the low-dimensional latent space. Since the latent vectors sampled from the Gaussian distribution are concentrated around the mean, the small number of training iterations means that the generator has potentially not fully explored the latent spaces far from the mean. For the uniform prior, this does not happen as the latent vectors are sampled with equal probability. Hence, in our case the generator will likely have learned a more complete mapping of the latent space when using a uniform distribution.

On the other hand, the well-formed samples generated from a Gaussian prior are arguably of better quality than compared with the uniform prior. Especially for the MNIST images, such as row 1 column 1, row 2 column 1, and row 5 column 2 of Fig. 9(b), the images appear sharper and have

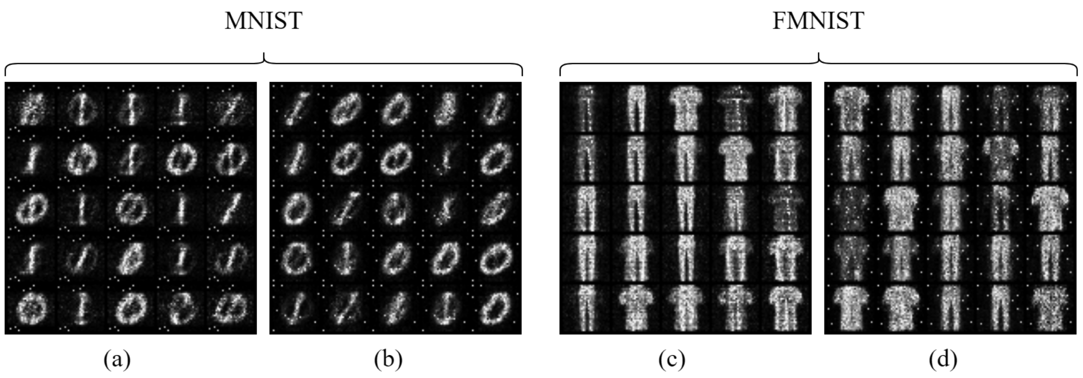


FIGURE 8. Effect of shape of patches. In this experiment, we fixed the number of patches, layers, and data qubits to be 28, 10, and 5, respectively, and varied the shape of the patches. The images show random outputs sampled from generators with either 1 × 28 or 7 × 4 patches trained on MNIST and FMNIST. In both cases, having different patches shapes do not affect the quality of the outputs. However, should artifacts exist, having different patch shapes can lead to artifacts appearing in more obvious locations. MNIST. (a) 1 × 28 patches. (b) 7 × 4 patches. FMNIST. (c) 1 × 28 patches. (d) 7 × 4 patches.

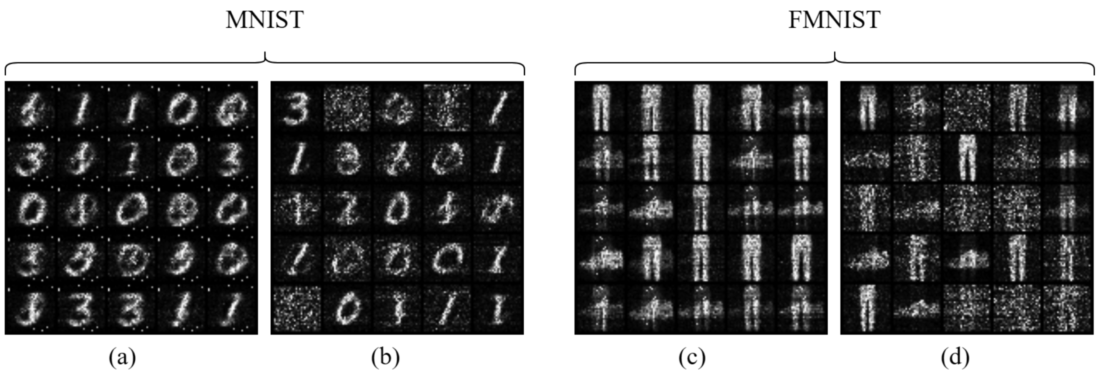


FIGURE 9. Effect of prior distribution. In this experiment, we fixed the number of patches, layers, and data qubits to be 28, 15, and 5, respectively, and varied the choice of prior distribution. The images show random outputs sampled from generators with a uniform or Gaussian prior trained on MNIST and FMNIST. In both cases, the uniform prior allows the generator to learn a more complete mapping for every latent vector, whereas the Gaussian prior allows the generator to produce sharper images for those that are well formed. MNIST. (a) Uniform prior. (b) Gaussian prior. FMNIST. (c) Uniform prior. (d) Gaussian prior.

less noise compared with those in Fig. 9(a). Furthermore, the outputs from the Gaussian prior are free from artifacts. Since the generator is frequently trained on latent vectors centered around the mean, the outputs generated from this region of the latent space will be of better quality. As the training progresses using a Gaussian prior, we observed that there are less instances of white noise generated by the generator, which supports the argument of the white noise coming from an unexplored region of the latent space. In the future when there are more resources to run larger scale training, the PQWGAN trained on a Gaussian prior may generate better outputs than using a uniform prior while avoiding the problem of artifacts.

F. SUMMARY OF EFFECTS OF GENERATOR PARAMETERS

In this section, we have investigated the effects of various parameters on our quantum generator. Here, we summarize some general observations on the impact of different parameter choices on the output quality. First, in terms of number of patches to use, we observe that having more patches in general corresponds to better output quality. In our experiments, we were unable to learn anything meaningful with one patch, and the output quality improved as we increased the patches. Next, as we increase the number of data qubits, the outputs were sharper and less artifacts were observed. Third, as we increased the number of layers in a subgenerator, the images are visibly sharper when we go from 5 to 10 and to 15 layers but are less noticeable from 15 layers onward. Fourth, the shape of the patches do not have much impact on the quality and general sharpness of the outputs, but having square patches will lead to more noticeable artifacts if they exists. Finally, using a uniform prior allows the generator to quickly explore the whole latent space while the Gaussian prior is slower and leads to more white noise in our small-scale experiments. However, the well-formed outputs generated by the Gaussian prior are arguably of higher quality and do not suffer from artifacts.

VIII. CONCLUSION AND FUTURE DIRECTIONS

Classical GANs have seen great success in image generation while QGANs are still far away from that level. However, with the promise that quantum computing brings, QGANs are an exciting area of research. In this article, we proposed our novel PQWGAN framework for image generation and empirically evaluated the effects of varying its different parameters. The PQWGAN is composed of a quantum generator and classical critic, and is inspired by the patch strategy QGAN and the training algorithm of WGAN-GP. We applied our PQWGAN to learn on subsets of 28×28 pixels images of MNIST and FMNIST. Specifically, we successfully learned the two-class subsets of MNIST 0/1 and FMNIST T-shirts/trousers, and the three-class subset of MNIST 0/1/3. We compared these results with a WGAN-GP and found that our PQWGAN was able to achieve comparable results despite having three orders of magnitude less trainable parameters in the generator. To the best of authors' knowledge, this is the first time that a QGAN with a quantum generator has demonstrated that it can successfully learn multimodal image data of this resolution on these standard datasets. Finally, we investigated how varying different parameters has an effect on the output image quality. We found that as a general rule of thumb, having more patches, data qubits, and layers is beneficial, while the shape of the patches did not matter too much. We also found that using a uniform prior is beneficial in the near term, but may be outperformed by a Gaussian prior in the future.

We suggest four future directions to explore. In this work, we have shown the working of our method based on noiseless simulations due to the challenges associated with the limited capabilities of the current generation of NISQ devices. The scope of this article was to develop a new quantum generative learning technique, which can produce high-resolution images and demonstrate its working in the theoretical setting. However, future work could implement these methods on quantum processors to test and benchmark their performance in the presence of noise. The patch strategy QGAN [27] successfully learned to generate simple 8×8 pixels images on a 12-qubit superconducting computer. With 100+ qubit devices currently available and 1000+ qubit devices projected to arrive in the near future, it would be interesting to observe how our framework performs on a real quantum computer. Second, we could try more complex images, such as natural scenes with CIFAR-10 [53] or human faces with CelebA [54] in future work to see how our framework holds up in the presence of color and irregular image structure. Given adequate quantum resources, our framework is theoretically capable of breaking down and handling these larger generative tasks. Third, we noted that our choice of the quantum circuit for the generator lacked theoretical motivations. Like how convolutions have allowed deep learning for images to leap forward, there may be some operations that a quantum circuit can do that are especially helpful for images. Future work could look to experimenting with different ansatz to discover image manipulation techniques more suited for QML. Finally, our critic in this work is implemented using classical neural networks. As the field of QML develops, having a quantum critic can be beneficial due to the promises that QML brings. Combining the last three directions, our framework has the potential to be the foundation of new methods to push forward what is currently capable of QGANs, and work toward closing the gap in what is possible with classical and QML methods.

APPENDIX A
FURTHER DISCUSSIONS OF THE PQWGAN FRAMEWORK

A. POTENTIALS OF A QUANTUM CRITIC

A quantum critic is an interesting research avenue in the future. In classical GAN training, the discriminator and generator architectures are usually chosen such that one does not significantly overpower the other [43]. Although it is unclear whether this is a desirable property to have in GANs, having a rough idea of the expressive power of the GAN components can be used to help stabilize the training process. However, the relationship between the expressibility of classical neural networks and PQCs is unclear. Hence, having a quantum critic will be advantageous in this case as it can give a rough estimate of the expressiveness of the generator and critic.

In practice, a quantum critic may be constructed in the same way as a subgenerator with PQCs. Instead of measuring all qubits, we can measure one qubit at the end of the circuit to obtain a value, such as the Z expectation value. Then, to obtain an unbounded real-valued output to serve as an estimate of the Wasserstein distance, we could pass the output value through a tan function, similar to an activation in a classical neuron. However, due to the limited time and scope of this work, we were not concerned with this.

B. DISCUSSIONS OF THE TRAINING PROCESS

Arjovsky et al. [7] proved that for WGAN, the optimization process is principled when the critic and generator are constructed with neural networks. Here, we argue that this is also the case for our form of PQCs. We rely on the following assumption and theorem proved in [7].

Assumption 1: Let $g : \mathcal{Z} \times \mathbb{R}^d \rightarrow \mathcal{X}$ be locally Lipschitz between finite dimensional vector spaces. Denote $g_\theta(z)$ as the result of evaluating g with parameters θ at z. g satisfies the assumption for some probability distribution p over $\mathcal{Z}$ if there exists local Lipschitz constants $L(z, \theta)$ such that

$$\mathbb{E}_{z\sim p}[L(z, \theta)] < +\infty$$

Theorem 1: Let $\mathbb{P}_r$ be some distribution. Let $\mathbb{P}_\theta$ be the distribution generated by $g_\theta(z)$ where g is some function satisfying assumption 1 and z is some random variable with density $p(z)$. There exists a solution $f : \mathcal{X} \rightarrow \mathbb{R}$ to

$$\max_{||f||_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[f(x)] \tag{A1}$$

and

$$\nabla_\theta W(\mathbb{P}_r, \mathbb{P}_\theta) = -\mathbb{E}_{z\sim p(z)}[\nabla_\theta f(g_\theta(z))]$$

when both terms are well defined.

Intuitively, Theorem 1 tells us that it is possible for the generator to learn the target distribution under the WGAN objective as defined in (2) using the min–max method in GANs.

The inner maximization correspond to the Wasserstein distance reformulated under the Kantorovich–Rubinstein duality. By searching for a function f from the family of 1-Lipschitz functions that maximizes the difference in expectations in (A1), we obtain the Wasserstein distance between our target distribution and our generator distribution. We would like to minimize the Wasserstein distance between the target distribution and generator distribution, and hence we can use the usual gradient descent of on the Wasserstein distance. This corresponds to the outer minimization in the WGAN objective. With the GP term in WGAN-GP, the optimal critic is shown to have unit norm for straight lines connecting samples from $\mathbb{P}_r$ and $\mathbb{P}_\theta$. Hence, if expected value of the norm of the gradient deviates from 1, the critic will be penalized and will unlikely be the one that maximizes (A1). This preserves the nice properties of Theorem 1 and is empirically observed to support this claim. So, to show that the training of PQWGAN is also principled, it suffices to show that the quantum generator satisfies Assumption 1. Using the theorem proved in [45] on the Lipschitz continuity of PQCs, we argue that our generator satisfies this assumption.

Theorem 2. Given a function $f\colon [0, 2\pi]^M \rightarrow \mathbb{R}$ of the form $f(\boldsymbol{\theta}) = \langle\psi|U^\dagger(\boldsymbol{\theta})OU(\boldsymbol{\theta})|\psi\rangle$, where $|\psi\rangle \in \mathbb{C}^n$ is some arbitrary state for a finite n, $U(\boldsymbol{\theta})$ is a quantum circuit parameterized by $\boldsymbol{\theta}$ consisting of an arbitrary number of fixed gates, and M parameterized gates $U_i(\theta_i) = \exp(-i(\theta_i + c_i)H_i)$ for some constant c_i and Hermitian H_i. Then, for any observable O and any set of Hermitian operators, $f(\boldsymbol{\theta})$ is L-Lipschitz with

$$L = \sqrt{M}\left[\max_i\left(\sup_{\boldsymbol{\theta}}\left(\left|\frac{\partial f(\boldsymbol{\theta})}{\partial\theta_i}\right|\right)\right)\right]$$

Since each subgenerator is a PQC of the form stated in Theorem 2, the subgenerators are Lipschitz continuous. Then, to obtain our final output, we apply a series of linear transformations to the subgenerator outputs, which preserves Lipschitz continuity. Hence, our full generator satisfies Assumption 1 and the optimization process is principled.

APPENDIX B
ADDITIONAL IMAGE GENERATION EXPERIMENTS
A. COMPLEX BINARY FMNIST

For the case of FMNIST T-shirt/trousers in the main text, the generator is having an easier time as since many of the pixels of the T-shirt and trousers are overlapping. So, the difference between them are mostly around the sleeves of the T-shirt and between the legs of the trousers. Hence, we also investigated whether the our framework can learn distinctly different patterns in the form of trousers and sneakers. In this experiment, each subgenerator consists of 13 layers of six data qubits and one ancilla qubits, and thus the latent space will have seven dimensions. The results are shown in Fig. 10.

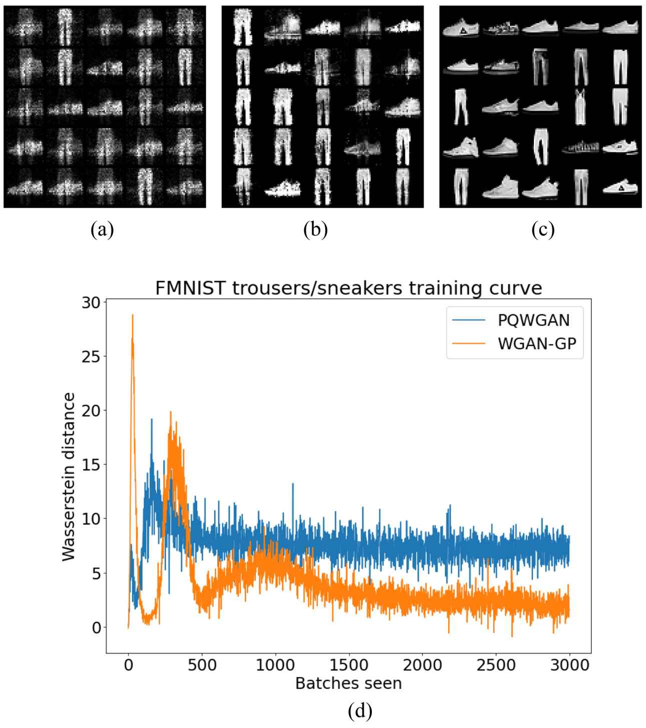


FIGURE 10. Samples and training curves of classical and quantum architecture on FMNIST trousers/sneakers. These samples are generated randomly from (a) PQWGAN, (b) WGAN-GP, and (c) the dataset—real samples. (d) Tracked Wasserstein distance of PQWGAN and WGAN-GP during training.

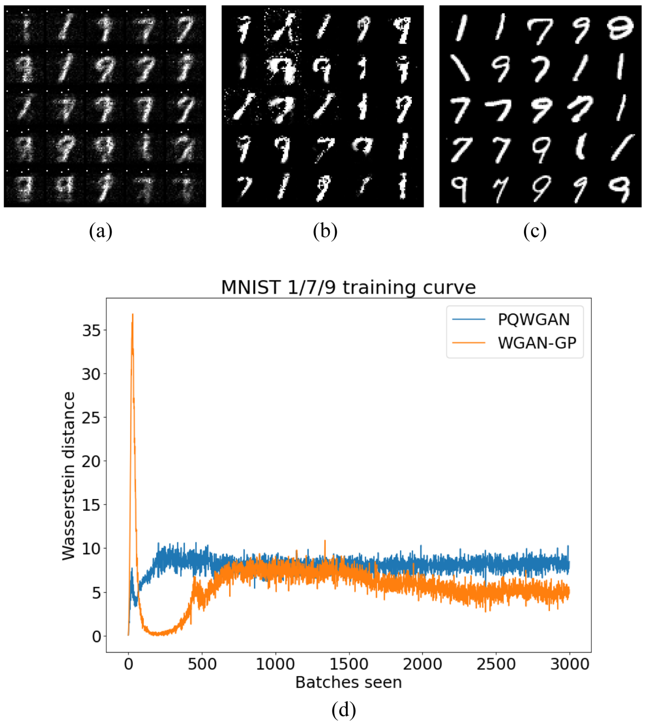


FIGURE 11. Samples and training curves of classical and quantum architecture on MNIST 1/7/9. These samples are generated randomly from (a) PQWGAN, (b) WGAN-GP, and (c) the dataset—real samples. (d) Tracked Wasserstein distance of PQWGAN and WGAN-GP during training.

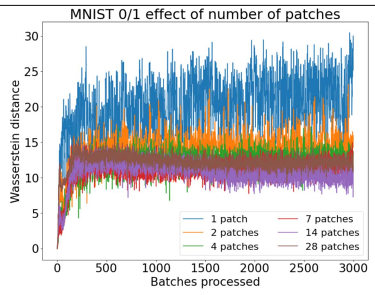
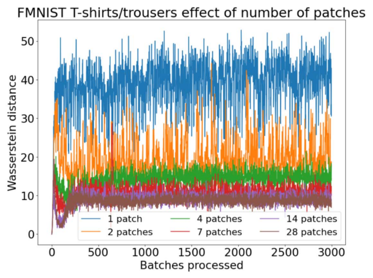
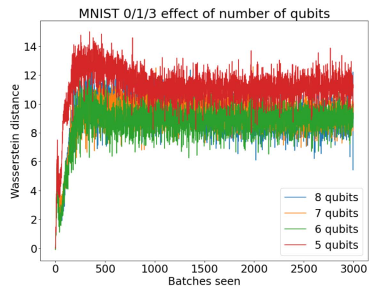
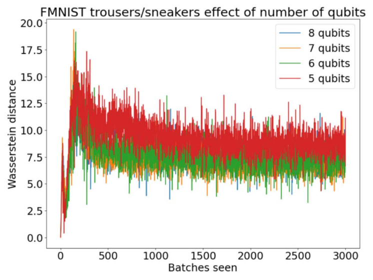
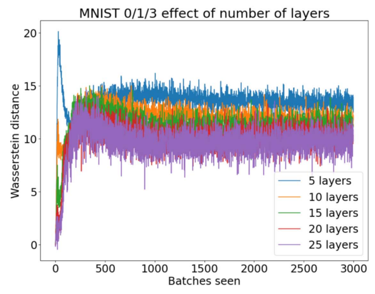
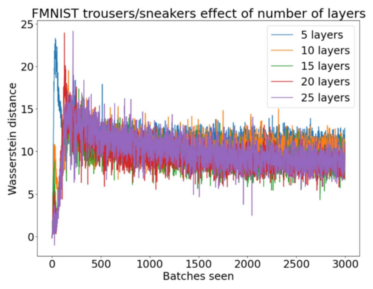
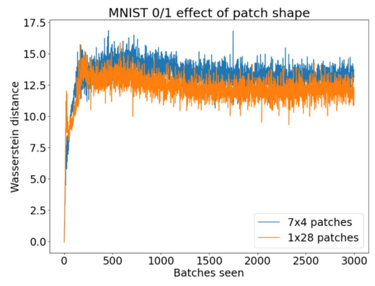
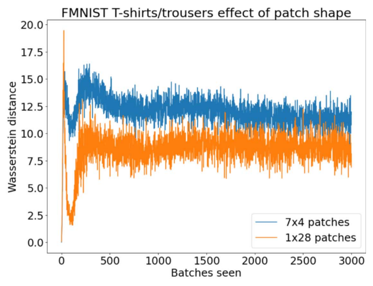
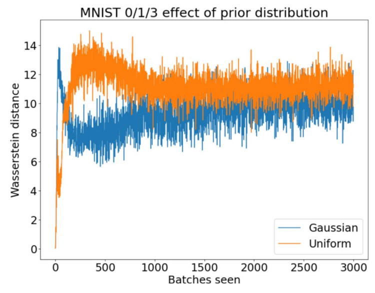
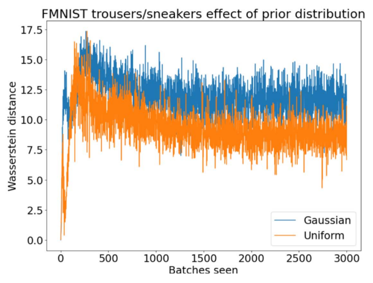
Dataset / Parameter	MNIST	FMNIST
Number of patches		
Number of qubits		
Number of layers		
Shape of patches		
Prior distribution		

FIGURE 12. Collection of the training curves from the parameter experiments conducted in Section VII.

Compared to the previous results, the PQWGAN is struggling more to converge to a set of parameters that can generate a comprehensive mapping of the latent space in this case. This is evident when we look at the training curve, where the Wasserstein distance of the PQWGAN converges and plateaus at a high value even if we continue to train it. Compared with the binary cases in the main text, there is a higher proportion of mixed samples of trousers and sneakers. This behavior is expected since the dataset is both more complex and has a greater difference between the two classes that we are trying to learn. Again, this points to the problem of having a limited expressiveness of the generator, and hence being unable to capture the details of the dataset we are learning from. However, the PQWGAN is clearly still able to learn something meaningful, as it not only is able to generate images of trousers and sneakers, but also there exists some different details within the same class, such as different shapes of sneakers in row 3 column 5 and in row 5 column 5 of Fig. 10(a). Hence, it is reasonable to expect that in the future when more resources are available, we can have larger models and training processes that can mitigate this problem.

B. TRIPLE MNIST

We also applied our PQWGAN to the simpler task of learning the digits 1, 7, and 9. Compared with the MNIST 0/1/3 in main text, the digits are similar in the sense that they are all have some form of a vertical stroke, and hence should be easier to generate. In this experiment, we use 11 layers per subgenerator and seven data qubits. The results are shown in Fig. 11.

In this experiment, both the classical and quantum frameworks are able to learn to output images that corresponds to the three digits. If we ignore the existence of the artifacts, the PQWGAN samples have a comparable quality to the WGAN-GP samples. Some of the samples generated from both these frameworks closely resemble the real samples, while others are still noisy. Furthermore, from the training curve, the PQWGAN achieves a similar Wasserstein distance compared with the WGAN-GP in the early stages of training. However, the WGAN-GP slowly converges to a lower score while it seems that the PQWGAN is starting to plateau, which can be attributed to the generator being not expressive enough for the PQWGAN.

APPENDIX C
TRAINING CURVES OF PARAMETER EXPERIMENTS

A collection of the training curves from the parameter experiments conducted in Section VII can be found in Fig. 12.

Data availability: The data that support the findings of this study are available within the article.

Code availability: The code to run the simulations can be found at https://github.com/jasontslxd/PQWGAN.

Competing financial interests: The authors declare no competing financial or nonfinancial interests.

ACKNOWLEDGMENT

The computational resources were provided by the National Computing Infrastructure and the Pawsey Supercomputing Center through the National Computational Merit Allocation Scheme, as well as by the University of Melbourne's Research Computing Services and the Petascale Campus Initiative.

REFERENCES

[1] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680, doi: 10.48550/arXiv.1406.2661.
[2] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434v2*, doi: 10.48550/arXiv.1511.06434.
[3] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, *arXiv:1609.02612*, doi: 10.48550/arXiv.1609.02612.
[4] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 1060–1069, doi: 10.48550/arXiv.1605.05396.
[5] H. Emami, M. M. Aliabadi, M. Dong, and R. B. Chinnam, "SPA-GAN: Spatial attention GAN for image-to-image translation," 2019, *arXiv:1908.06616v3*, doi: 10.48550/arXiv.1908.06616.
[6] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," 2017, *arXiv:1701.04862v1*, doi: 10.48550/arXiv.1701.04862.
[7] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017, *arXiv:1701.07875v3*, doi: 10.48550/arXiv.1701.07875.
[8] P. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. Symp. Found. Comput. Sci.*, 1994, pp. 124–134, doi: 10.1109/SFCS.1994.365700.
[9] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, 2017, doi: 10.1038/nature23474.
[10] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for Big Data classification," *Phys. Rev. Lett.*, vol. 113, no. 13, 2014, Art. no. 130503, doi: 10.1103/PhysRevLett.113.130503.
[11] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum algorithms for supervised and unsupervised machine learning," 2013, *arXiv:1307.0411v2*, doi: 10.48550/arXiv.1307.0411.
[12] N. Wiebe, A. Kapoor, and K. M. Svore, "Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning," *Quantum Inf. Comput.*, vol. 15, no. 3–4, pp. 316–356, 2015, doi: 10.48550/arXiv.1401.2142.
[13] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, "Quantum Boltzmann machine," *Phys. Rev. X*, vol. 8, no. 2, 2018, Art. no. 021050, doi: 10.1103/PhysRevX.8.021050.
[14] V. Havlíček et al., "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, no. 7747, pp. 209–212, 2019, doi: 10.1038/s41586-019-0980-2.
[15] M. T. West et al., "Towards quantum enhanced adversarial robustness in machine learning," *Nature Mach. Intell.*, vol. 5, 2023, Art. no. 581, doi: 10.1038/s42256-023-00661-1.
[16] Maxwell T. West et al., "Benchmarking adversarially robust quantum machine learning at scale," *Phys. Rev. Res.*, vol. 5, 2023, Art. no. 023186, doi: 10.1103/PhysRevResearch.5.023186.
[17] M. T. West, M. Sevior, and M. Usman, "Reflection equivariant quantum neural networks for enhanced image classification," *Mach. Learn.: Sci. Technol.*, vol. 4, 2023, Art. no. 035027, doi: 10.1088/2632-2153/acf096.
[18] M. T. West, M. Sevior, and M. Usman, "Boosted ensembles of qubit and continuous variable quantum support vector machines for B. Meson flavor tagging," *Adv. Quantum Technol.*, vol. 6, no. 10, Oct. 2023, Art. no. 2300130, doi: 10.1002/qute.202300130.
[19] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, "Expressive power of parametrized quantum circuits," *Phys. Rev. Res.*, vol. 2, Jul. 2020, Art. no. 033125, doi: 10.1103/PhysRevResearch.2.033125.
[20] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, 2018, Art. no. 79, doi: 10.22331/q-2018-08-06-79.
[21] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, "Parameterized quantum circuits as machine learning models," *Quantum Sci. Technol.*, vol. 4, no. 4, 2019, Art. no. 043001, doi: 10.1088/2058-9565/ab4eb5.
[22] S. Lloyd and C. Weedbrook, "Quantum generative adversarial learning," *Phys. Rev. Lett.*, vol. 121, no. 4, 2018, Art. no. 040502, doi: 10.1103/PhysRevLett.121.040502.
[23] P.-L. Dallaire-Demers and N. Killoran, "Quantum generative adversarial networks," *Phys. Rev. A.*, vol. 98, no. 1, 2018, Art. no. 012324, doi: 10.1103/PhysRevA.98.012324.
[24] C. Zoufal, A. Lucchi, and S. Woerner, "Quantum generative adversarial networks for learning and loading random distributions," *npj Quantum Inf.*, vol. 5, no. 1, 2019, Art. no. 103, doi: 10.1038/s41534-019-0223-2.
[25] A. Assouel, A. Jacquier, and A. Kondratyev, "A quantum generative adversarial network for distributions," 2021, *arXiv:2110.02742v1*, doi: 10.48550/arXiv.2110.02742.

[26] J. Li, R. Topaloglu, and S. Ghosh, “Quantum generative models for small molecule drug discovery,” 2021, *arXiv:2101.03438v1*, doi: 10.48550/arXiv.2101.03438.

[27] H.-L. Huang et al., “Experimental quantum generative adversarial networks for image generation,” *Phys. Rev. Appl.*, vol. 16, 2021, Art. no. 024051, doi: 10.1103/PhysRevApplied.16.024051.

[28] S. A. Stein et al., “QUGAN: A quantum state fidelity based generative adversarial network,” in *Proc. IEEE Int. Conf. Quantum Comput. Eng.*, 2021, pp. 71–81, doi: 10.1109/QCE52317.2021.00023.

[29] C. Chu, G. Skipper, M. Swany, and F. Chen, “IQGAN: Robust quantum generative adversarial network for image synthesis on NISQ devices,”2022, *arXiv:2210.16857v1*, doi: 10.48550/arXiv.2210.16857.

[30] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved training of Wasserstein GANs,” 2017, *arXiv:1704.00028v3*, doi: 10.48550/arXiv.1704.00028.

[31] Y. LeCun, C. Cortes, and C. J. Burges, “The MNIST database,” [Online]. Available: http://yann.lecun.com/exdb/mnist/

[32] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms,” 2017, *arXiv:1708.07747v2*, doi: 10.48550/arXiv.1708.07747.

[33] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training GANs,” 2016, *arXiv:1606.03498v1*, doi: 10.48550/arXiv.1606.03498.

[34] A. W. Harrow, A. Hassidim, and S. Lloyd, “Quantum algorithm for linear systems of equations,” *Phys. Rev. Lett.*, vol. 103, 2009, Art. no. 150502, doi: 10.1103/PhysRevLett.103.150502.

[35] D. Dua and C. Graff, “UCI machine learning repository,” 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[36] S. Chakrabarti, Y. Huang, T. Li, S. Feizi, and X. Wu, “Quantum Wasserstein generative adversarial networks,” 2019, *arXiv:1911.00111v1*, doi: 10.48550/arXiv.1911.00111.

[37] B. T. Kiani, G.De Palma, M. Marvian, and Z. Liu, and S. Lloyd, “Learning quantum data with the quantum Earth mover’s distance,” *Quantum Sci. Technol.*, vol. 7, no. 4, 2022, Art. no. 045002, doi: 10.1088/2058-9565/ac79c9.

[38] D. Herr, B. Obert, and M. Rosenkranz, “Anomaly detection with variational quantum generative adversarial networks,” *Quantum Sci. Technol.*, vol. 6, no. 4, 2021, Art. no. 045004, doi: 10.1088/2058-9565/ac0d4d.

[39] Machine Learning Group - ULB, “Credit card fraud detection,” 2016. [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

[40] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” 2013, *arXiv:1312.6114v10*, doi: 10.48550/arXiv.1312.6114.

[41] A. Kandala et al., “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” *Nature*, vol. 549, no. 7671, pp. 242–246, 2017, doi: 10.1038/nature23879.

[42] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, “Barren plateaus in quantum neural network training landscapes,” *Nature Commun.*, vol. 9, no. 1, Nov. 2018, Art. no. 4812, doi: 10.1038/s41467-018-07090-4.

[43] I. Goodfellow, “NIPS 2016 tutorial: Generative adversarial networks,” 2017, *arXiv:1701.00160v4*, doi: 10.48550/arXiv.1701.00160.

[44] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, “Evaluating analytic gradients on quantum hardware,” *Phys. Rev. A*, vol. 99, no. 3, 2019, Art. no. 032331, doi: 10.1103/PhysRevA.99.032331.

[45] R. Sweke et al., “Stochastic gradient descent for hybrid quantum-classical optimization,” *Quantum*, vol. 4, 2020, Art. no. 314, doi: 10.22331/q-2020-08-31-314.

[46] A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” 2019, *arXiv:1912.01703v1*, doi: 10.48550/arXiv.1912.01703.

[47] V. Bergholm et al., “PennyLane: Automatic differentiation of hybrid quantum-classical computations,” 2018, *arXiv:1811.04968v4*, doi: 10.48550/arXiv.1811.04968.

[48] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, *arXiv:1412.6980v9*, doi: 10.48550/arXiv.1412.6980.

[49] T. White, “Sampling generative networks,” 2016, doi: 10.48550/arXiv.1609.04468.

[50] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, “Cost function dependent barren plateaus in shallow parametrized quantum circuits,” *Nature Commun.*, vol. 12, no. 1, 2021, Art. no. 1791, doi: 10.1038/s41467-021-21728-w.

[51] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, “Connecting ansatz expressibility to gradient magnitudes and barren plateaus,” *PRX Quantum*, vol. 3, Jan. 2022, Art. no. 010313, doi: 10.1103/PRXQuantum.3.010313.

[52] S. Hadfield, Z. Wang, B. O’Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, “From the quantum approximate optimization algorithm to a quantum alternating operator ansatz,” *Algorithms*, vol. 12, no. 2, 2019, Art. no. 34, doi: 10.3390/a12020034.

[53] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” M.S. thesis, Univ. Toronto, ON, Toronto, Canada, Tech. Rep., 2009.

[54] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 3730–3738, doi: 10.1109/ICCV.2015.425.