OPTICA QUANTUM

Photonic quantum generative adversarial networks for classical data

TIGRAN SEDRAKYAN[1,2] AND ALEXIA SALAVRAKOS[1,*]

[1] *Quandela, 7 Rue Léonard de Vinci, 91300 Massy, France*
[2] *Sorbonne Université CNRS, LIP6, F- 75005 Paris, France*
**alexia.salavrakos@quandela.com*

Received 24 May 2024; revised 6 September 2024; accepted 5 November 2024; published 16 December 2024

In generative learning, models are trained to produce new samples that follow the distribution of the target data. These models were historically difficult to train, until proposals such as generative adversarial networks (GANs) emerged, where a generative and a discriminative model compete against each other in a minimax game. Quantum versions of the algorithm have since been designed for the generation of both classical and quantum data. While most work so far has focused on qubit-based architectures, in this article we present a quantum GAN based on linear optical circuits and Fock-space encoding, which makes it compatible with near-term photonic quantum computing. We demonstrate that the model can learn to generate images by training the model end-to-end experimentally on a single-photon quantum processor.

https://doi.org/10.1364/OPTICAQ.530346

1. INTRODUCTION

Photonic quantum hardware represents one of the most promising paths for the realization of a quantum computer. There is a strong potential for scaling in the number of qubits [1], and various computation models and architectures have been designed for photonics [2,3]. Moreover, the possibility of demonstrating a near-term computational advantage in tasks such as boson sampling [4] makes it a noteworthy candidate for noisy intermediate-scale quantum (NISQ) technology [5]. Even though building photonic circuits with large numbers of single photons and optical modes is challenging given current technology, the rate of hardware advancement in the field is very encouraging [6].

Nevertheless, little effort has been dedicated so far to exploiting photonic-native architectures for machine learning tasks, i.e., architectures where the components are single photon sources, photon detectors, linear optical circuits with beam splitters and phase shifters, and where the problem is encoded in the Fock space. Some of the existing works concentrate on discriminative learning [7–9], but generative learning models remain mostly understudied [10,11], despite the great potential shown by classical generative models in recent years. While Fock-space-based models do not consist of traditional qubits, they do exhibit quantum properties which could be harnessed in machine learning tasks.

In this work, we propose a quantum generative adversarial network (QGAN) where the generator network is a variational photonic quantum circuit [12]. We train our model on the MNIST [13] dataset of handwritten digits in reduced dimension, using a patch-based image generation approach, in both ideal and noisy settings. We run the full training procedure as a physical experiment on Quandela's quantum processing unit *Ascella* introduced in [9], whose setup consists of a single-photon source connected to an integrated photonic chip and photon detectors. Our work is a proof-of-concept demonstration that photonic quantum adversarial models can be trained to generate classical data, and our results thus contribute to a growing body of literature on near-term quantum machine learning implementations.

2. BACKGROUND

Generative models are designed to produce previously unseen data that follow certain patterns. Their training consists in feeding the model some target training samples that are representative of the desired outcome and optimizing the model so that its outputs grow closer to these target samples. This corresponds to learning the underlying distribution from which the training samples are drawn, i.e., the data generating distribution. Generative models have a long-standing history; however, only recent advances in deep neural networks enabled the creation of deep generative models (DGMs), such as GANs, as well as variational autoencoders (VAEs), autoregressive and diffusion models [14–17]. Among other factors, these advances were made possible due to training techniques and properties of deep neural networks [18].

2.1. Classical GANs

In this work, we focus on GANs as they are realizable on a relatively small scale and can be trained efficiently. First proposed in [14], GANs marked an important milestone in the field of generative learning models. The primary concept of adversarial learning consists of two competing deep neural networks—the generator, often denoted as G, and the discriminator D. The generator accomplishes the task of data generation, by transforming the noise $z \sim p_z(z)$ sampled from the latent space $\mathcal{Z}$ (also known as noise prior) into a fake data sample. Then, both generator and discriminator networks compete against each other in an adversarial zero-sum game, where the generator is trying to produce fake samples close to the real target samples drawn from the data generating distribution $\boldsymbol{x} \sim p_{data}(\boldsymbol{x})$, and the discriminator is trying to classify the real samples from the fake ones. The process is repeated iteratively until the generator starts producing realistic results, which may correspond to a Nash equilibrium being reached for the zero-sum game [19].

The learning process tries to maximize the loss value across the discriminator parameters, so that the discriminator is able to distinguish between the fake and real data. At the same time, it tries to minimize the loss over the generator parameters, so as to generate more realistic samples and confuse the discriminator. Mathematically, this is equivalent to a min–max optimization of a loss function $L(D, G)$ defined on the discriminator and generator models:

$$\min_G \max_D L(D, G), \tag{1}$$

where

$$\begin{aligned} L(D, G) = &\ \mathbb{E}_{x\sim p_{data}(\boldsymbol{x})}[\log(D(\boldsymbol{x}))] \\ &+ \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]. \end{aligned} \tag{2}$$

This can be expanded into the problem of maximizing two separate loss functions, the generator loss L_G and the discriminator loss L_D:

$$\max_{\theta_G} L_G \quad \text{and} \quad \max_{\theta_D} L_D, \tag{3}$$

where

$$\begin{aligned} L_G &= \frac{1}{n}\sum_{i=1}^{n} \log\left(D(G(z_i))\right), \\ L_D &= \frac{1}{n}\sum_{i=1}^{n} \left[\log(D(\boldsymbol{x}_i)) + \log(1 - D(G(z_i)))\right], \end{aligned} \tag{4}$$

with n the number of training samples from the dataset, and $\boldsymbol{x}_i$ and z_i respectively the ith real and noise samples.

In practice, a batch of noise samples from the latent space is supplied to the generator. It produces a batch of results, which are then used for the discriminator. Simultaneously, the discriminator is supplied a batch of samples from the training dataset. The loss is therefore computed by averaging over the batch, which has the added advantage of stabilizing the learning process. For every learning iteration, optimization steps are performed first for the discriminator and then for the generator, using gradient descent (or ascent), where the gradient is computed using backpropagation. The number of optimization steps k for the discriminator depends on the specific use case. The full training algorithm is shown in Section S1 of the Supplement 1.

2.2. Quantum GANs

Quantum generative adversarial networks (QGANs) were introduced in [20,21] as a quantum alternative to GANs. Rather than a single architecture, they consist of a set of concepts where at least one, if not both, of the components—the generator and the discriminator—possess a certain degree of quantum capabilities.

Moreover, not only the networks, but also the data or type of problem, can be quantum or classical, as is discussed in detail in [20]. Additionally, the latent space can also be generated by a quantum source. References [22] and [23] consider the alternative case where only the latent space of an otherwise fully classical GAN is generated quantumly, in qubit-based and photonic-native scenarios, respectively. In [21], an early example for the generation of quantum data, i.e., the generation of a quantum state, is presented. The problem is defined so that, in practice, the generator learns how to implement a CNOT quantum gate. A photonic example for the generation of quantum data was studied recently in [24]. For the generation of classical data with a QGAN, qubit-based models were developed in various works [25–28], and a model where the generator is based on the combination of a linear optical circuit and a neural network was very recently introduced in [29].

Here, we focus on the latter task of generating classical data, with photonic quantum circuits as a resource. In our scenario, the generator is a fully quantum variational circuit, while the discriminator is a fully classical discriminative neural network. The communication between both networks happens by obtaining classical samples from the output measurements of the generator and feeding them to the discriminator, along with the target data. The rest of the training progresses as for classical GANs: the discriminator is optimized first followed by the generator, iteratively.

2.3. Image Generation and Patch-Based Approach

The specific task we consider for our QGAN is to generate images. We use a patch-based approach to exploit only a small number of photons and modes, and thus make our scheme easier to execute experimentally. The patch-based approach involves generating parts of larger images from separate quantum generators, which we call sub-generators, and combining or stacking the outputs together to obtain the full image. Such an approach has previously been shown to work on the 8×8 MNIST dataset in [26].

We focus on this same dataset in our work. It is a downscaled version of the popular 28×28 MNIST dataset of handwritten digits, which was originally used for small-scale classical generative models. It consists of a collection of digit images ranging from 0 to 9, each image being of size 8×8. The dataset contains approximately 560 entries for each digit, with 5621 datapoints in total. Each datapoint consists of 64 pixels and each pixel has continuous intensity value in the interval $[0, 1]$, with 0 being fully black pixels and 1 being fully white ones. Some examples, sampled randomly, are shown in Fig. 1. While the downscaling causes lower quality, the digits still closely resemble actual handwritten numbers and provide enough diversity for the tasks discussed in this work.

3. PHOTONIC QGAN

We present our proposal for a photonic QGAN in Fig. 2, where the quantum generator is implemented using linear optical variational quantum circuits. Having a photonic-native model means that the circuit ansatz consists of optical modes with parameterized phase shifters and beam splitters, which is reflected in Fig. 3. In this framework, we consider as the input and output

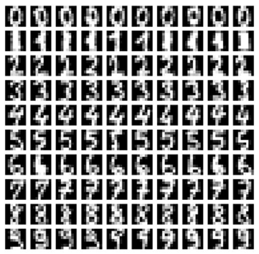

Fig. 1. Randomly sampled entries from the 8×8 MNIST dataset. Each row corresponds to a separate digit, sorted in increasing order 0–9.

states of the circuit the Fock states of n photons in m modes, as in [7]. We denote an input Fock state as $|\vec{n}_{\text{in}}\rangle = |n_1^{\text{in}}, \ldots, n_m^{\text{in}}\rangle$, where n_i^{in} indicates the number of photons in mode i. Naturally, $\sum_i n_i^{\text{in}} = n$. Likewise, we can write an output Fock state as $|\vec{n}_{\text{out}}\rangle = |n_1^{\text{out}}, \ldots, n_m^{\text{out}}\rangle$: they are detected as arrangements of photons in the output modes, which we denote as $s = (n_1^{\text{out}}, \ldots, n_m^{\text{out}})$. If there is no photon loss, the n_i^{out} sum to n as well.

An obvious way to design the quantum generator is to consider that one output state corresponds to one data sample, and to define a mapping between the Fock states and the space of the training data. As a simple example, let us imagine that we want to generate integers between 0 and 100, according to a certain target data distribution. We can then choose the number of modes m and the number of photons n such that there are at least 100 possible output Fock states, and map each output state to an integer. In this scenario, one run of the quantum circuit produces one sample. This approach, which we could call sample based, is used, for instance, in related work on photonic quantum circuit Born machines [30].

However, while we are limited to small-scale devices, such as the 12 modes and 6 photons of the *Ascella* processor [9], using this approach also means that we are restricted in the type of datasets that we can consider. Let us suppose that we aim to generate 8×8 MNIST digits and that we use the patch-based approach as mentioned in the previous section with four patches, so that each circuit must generate images of 16 pixels at a time. If each pixel only had two intensity values (black or white), the dimension of the resulting space would be 2^{16}. This is already beyond what we could implement on *Ascella*, and even more so if we considered the actual range of pixel intensities of the digit images. For this reason, and for the purpose of this work, we propose an alternative new mapping in the next section.

3.1. Distribution-Based Mapping

In this approach, we compute the probability distribution on the output Fock states by performing several thousands of measurement shots at the end of the generator circuit. This discrete

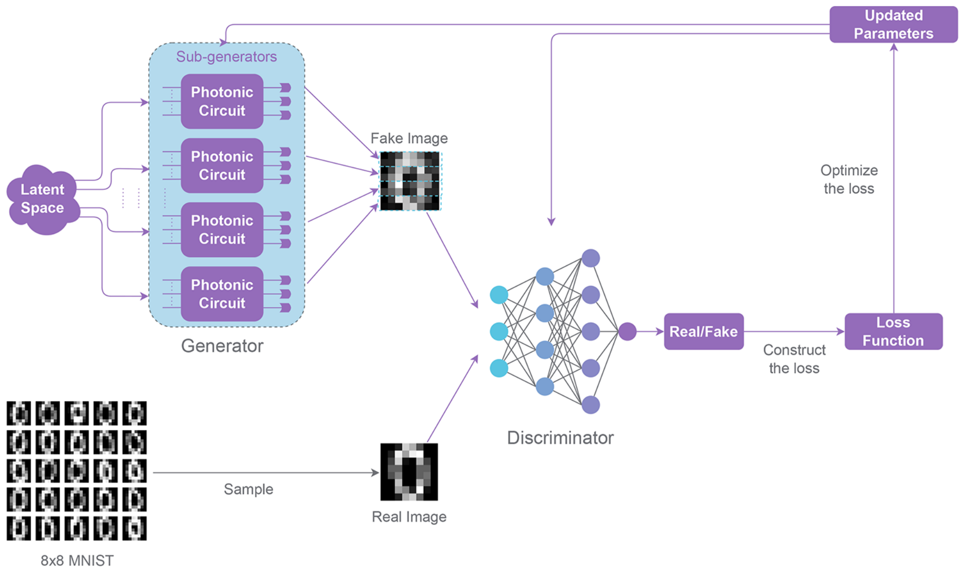

Fig. 2. Proposal for image generation with a photonic QGAN. Noise from the latent space is fed into each sub-generator of the patch-based approach. These are variational photonic quantum circuits detailed in Fig. 5. The output distribution of the sub-generators is mapped to image pixels, which are then recombined together to form a complete image, following the patch-based approach. The fake images are provided to the discriminator, along with the real images. The discriminator is a classical neural network and classifies the image as real or fake. Based on these results, the loss is constructed as per Eq. (4). After optimizing the loss, the parameters of the generator and the discriminator are updated.

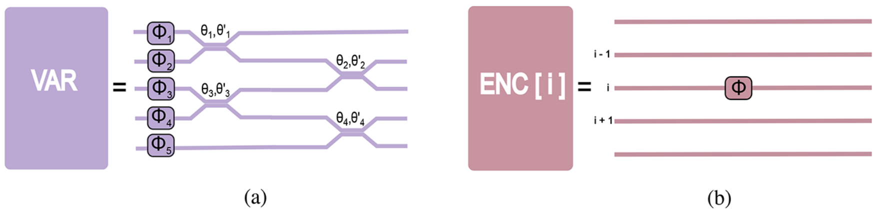

Fig. 3. Structures of the (a) variational layers and (b) encoding or noise layers. Phase shifters are depicted by squares and beam splitters by crossing between the modes. Parameters of the variational layers are trainable and parameters of the encoding layers are sampled from the latent space.

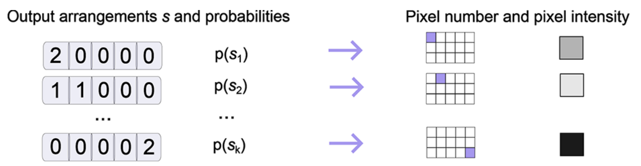
Output arrangements s and probabilities
Pixel number and pixel intensity
2 0 0 0 0
p(s1)
1 1 0 0 0
p(s2)
0 0 0 0 2
p(sk)

Fig. 4. Distribution-based mapping. Each output Fock state observed as arrangement $s = (n_1^{\text{out}}, \dots, n_m^{\text{out}})$ is mapped to a pixel number in the image, and the associated estimated probability is mapped to the intensity of the pixel.

output distribution is then mapped to a discrete distribution on integers. If needed, binning may be performed so that several output Fock states correspond to the same integer. The index of a bin, i.e., the integer, corresponds to the location of a pixel on the image, while the probability of the bin corresponds to the pixel intensity, as shown in Fig. 4. This allows us to obtain continuous pixel intensity values. To cover their full range, the probability values are renormalized to the interval [0, 1] using min–max normalization.

It is important to note that the number of possible output states of the generator does not always match the number of pixels necessary for the image or the patch, so the output distribution of each sub-generator may be trimmed equally on each tail, under the assumption that tails of distributions do not carry much information.

When transforming Fock states to integers, while we can apply an arbitrary mapping scheme, it would intuitively make sense if photon arrangements physically close to each other in the device would correspond to integers that are close as well. We thus consider outputs with the most number of photons in the rightmost modes as closer to 0 in their integer mapping. Moreover, the larger the number of photons in the rightmost mode, the smaller is the mapped integer. Correspondingly, states with a larger number of photons in the leftmost modes are considered further away from 0. Naturally, the number of available integers corresponds to the number of distinguishable states.

This approach is most efficient when photon number resolving (PNR) detectors are available. One of the main advantages photon number resolution provides is that it allows us to observe a larger amount of output states for given values of m and n. However, PNR detectors remain difficult to design with current technology. With threshold detectors, the only accessible values are binary—0 (no photon) or 1 (click, i.e., presence of photons). Learning is of course still possible, but the mapping differs since several states with photon bunching are indistinguishable from each other. A sample mapping for three modes and three photons is shown in Table 1.

This mapping assumes ideal conditions without photon loss. In noisy settings, the mapping does not need to be updated if PNR is available, since lossy states can be properly detected and filtered out of the final distribution with postselection. However, with threshold detectors, photon loss introduces an ambiguity in the output distribution and another mapping is necessary. In practice, lossless states with photon bunching cannot be distinguished from lossy states and they are both discarded in postselection. For the case of three photons with three modes, the number of output states reduces to one—$|1, 1, 1\rangle$. In such a situation, m or n must be increased to recover the necessary number of integers for the image size.

Table 1. Fock State to Integer Mapping Table for a Noiseless Setup with Three Modes and Three Photons

Integer	PNR	No PNR (state)	No PNR (pattern)[a]
0	$\|0,0,3\rangle$	$\|0,0,3\rangle$	$\|0,0,\text{click}\rangle$
1	$\|0,1,2\rangle$	$\|0,3,0\rangle$	$\|0,\text{click},0\rangle$
2	$\|0,2,1\rangle$	$\|0,2,1\rangle$, $\|0,1,2\rangle$	$\|0,\text{click},\text{click}\rangle$
3	$\|0,3,0\rangle$	$\|3,0,0\rangle$	$\|\text{click},0,0\rangle$
4	$\|1,0,2\rangle$	$\|2,0,1\rangle$, $\|1,0,2\rangle$	$\|\text{click},0,\text{click}\rangle$
5	$\|1,1,1\rangle$	$\|2,1,0\rangle$, $\|1,2,0\rangle$	$\|\text{click},\text{click},0\rangle$
6	$\|1,2,0\rangle$	$\|1,1,1\rangle$	$\|\text{click},\text{click},\text{click}\rangle$
7	$\|2,0,1\rangle$	—	—
8	$\|2,1,0\rangle$	—	—
9	$\|3,0,0\rangle$	—	—

[a]The last column clarifies the detection pattern without PNR.

3.2. The Ansatz

In our patch-based approach, images are generated in horizontal patches by separate sub-generators and eventually stacked vertically to form the full image. Each sub-generator corresponds to a linear optical quantum circuit. When designing the ansatz, we considered setups with two quantum sub-generators, where each sub-generator generates patches of 32 pixels, as well as setups with four quantum sub-generators, where the patches contain 16 pixels.

In a given setup, all the sub-generators of the generator have the same structure. This structure consists of variational layers, and encoding or noise layers, as described in Fig. 3. The variational layers contain the parameters that are optimized during the training of the model and their structure is inspired from [31,32]. The encoding or noise layers are used to introduce noise z into the model (here sampled from a normal distribution). These encoding-reuploading layers consist of phase shifters. They ensure that the resulting distribution over pixel intensities is different for each input noise sample, and that the model can thus generate a variety of data points, as well as generalize better. In general, noise-reuploading adds to the nonlinearity of the input–output mapping, improving the diversity in the generated images and encouraging the model to learn patterns rather than memorizing them [33,34].

We explored several structures for the sub-generator circuits. This structure can be adjusted by alternating the number and the arrangement of the variational and encoding layers. We display in Fig. 5 the circuits that we found to be fairly efficient at solving our image generation task. The smallest circuit only has one encoding layer, while the largest one has three. The number of modes may vary compared with the circuits displayed in Figs. 4 and 5, but the general layer structure is preserved. It is important to note that considering the empiric nature of the findings, the circuit configurations are not guaranteed to be optimal, but are rather a heuristic combination of an educated guess and non-exhaustive search.

3.3. Training and Optimization

The training of photonic QGANs progresses similarly to the regular GANs, as in Algorithm S1. The classical discriminator is trained first for one step using backpropagation-enabled stochastic gradient ascent to maximize L_D, after which the quantum generator is trained for several steps of simultaneous perturbation stochastic approximation (SPSA) [35] iterations, to

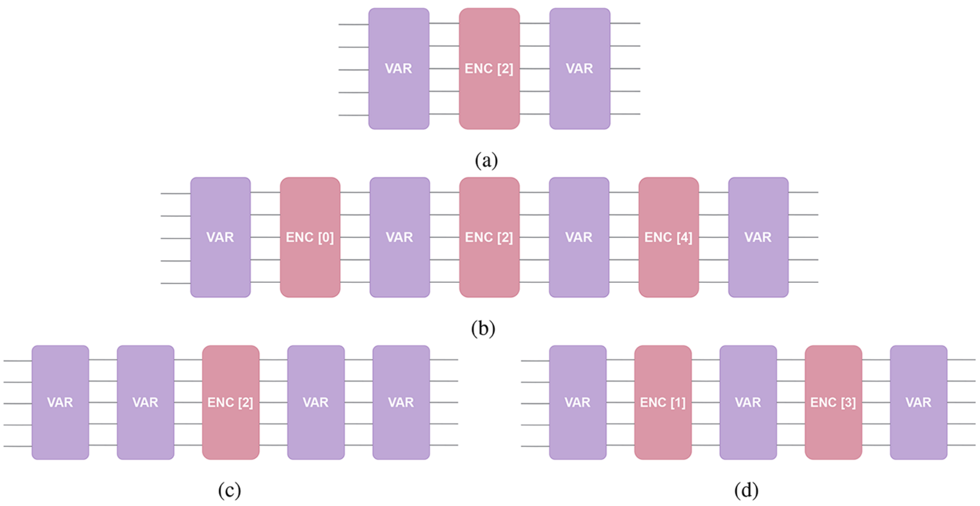

Fig. 5. Sub-generator circuit structures used for photonic QGAN training. (a) Circuit setup A, (b) circuit setup B, (c) circuit setup C, (d) circuit setup D. Layer structures are shown in Fig. 4.

optimize L_G. All the parameters are updated, and this process is repeated until the maximum number of training epochs is reached.

SPSA is an optimization technique based on a stochastic approximation of the gradient. Due to the fact that this approximation is an almost unbiased estimator of the gradient, convergence of the method is guaranteed under reasonably general conditions. The primary advantage of the SPSA algorithm lies in the amount of circuit evaluations necessary for approximating the gradient and its robustness to noise, including the noise induced by quantum sources [36,37]. SPSA requires only two evaluations, which allows cutting back on costly reconfiguration of linear optical gates, and considerably reduces the duration of both simulations and experiments. Indeed, the number of training steps of the models scale only constantly (rather than linearly) with the number of parameters.

We initialize the parameters for SPSA in a way which allows the initial generator pseudo-gradients to be large enough for a successful kick-start to the optimization. To achieve this, parameters are initialized randomly, the initial gradients are computed and if the values are too small, a reinitialization is performed. This parameter reinitialization is repeated until the starting pseudo-gradient values are in a desired range. Initialization performed in this way does not guarantee convergence but, in most cases, allows the generator enough starting optimization momentum to be able to compete with the more exact gradient calculation of the classical discriminator, thus enabling a balanced training.

3.4. Numerical Experiments: Ideal Simulations and Assessment

We perform our numerical experiments using Quandela's software package *Perceval* [38], designed for linear optical quantum circuits. We include an abstract pseudocode in Section S1 of the Supplement 1, and our Python code can be found in a companion Github repository [39].

Ideal conditions for simulations assume a perfect single-photon source, ideal components, no photon loss, perfect detectors, and the absence of sampling errors. First, we focus on the generation of digit "0" for the design of the model and the optimization of its hyperparameters. We then further study the best model for the generation of other digits and for noisy simulations in the next sections. We include all details about the model optimization and hyperparameter search in Section S2 of the Supplement 1.

We present our results in Fig. 6. The loss evolution plots describe how the values of the generator loss L_G and the discriminator loss L_D progress with the training epochs. For each configuration, we run ten training instances, and we display the average over these runs as a bold line and the standard deviation as a shaded area. We observe that even averaged over ten runs, loss values have small constant fluctuations throughout the training, which is a strong characteristic of adversarial training. When the L_D loss increases, the L_G loss decreases and vice versa. In the image evolution plots, we see that the models start with noisy outputs and that most of them produce quite realistic "0" terms by the end of the training (iteration 1500). The generated image plot shows several samples from the trained model, after the last epoch has been completed.

It appears that there is no specific loss value where the training can be stopped. However, good models generally reach an equilibrium where L_G and L_D losses start to fluctuate around the same moving average, without increasing or decreasing further. This is clearly observed in Fig. 6 (c) where both configurations converge to an equilibrium after approximately 1000 iterations. This equilibrium is centered around a value close to 0.69, which corresponds to $\log 2$. Taking a look at loss from Eq. (4), it becomes clear that $L_G = -\log 2$, when $D(G(z)) = 1/2$, that is, when the discriminator prediction about whether the fake image is real is as good as a random guess. Despite the fact this equilibrium is reached around iteration 1000, we note that further training does improve the results. Therefore, cutting off the training once the losses reach the value of $\log 2$ is not necessarily a desirable strategy.

Loss does not constitute a good assessment metric, and thus it makes sense to inspect the generated results using other qualitative or quantitative metrics. Important quantitative metrics in the assessment of generative models are the quality and the diversity of generated images. The quality measures the similarity to the original training data, while the diversity measures the variability between the generated images. Essentially, these

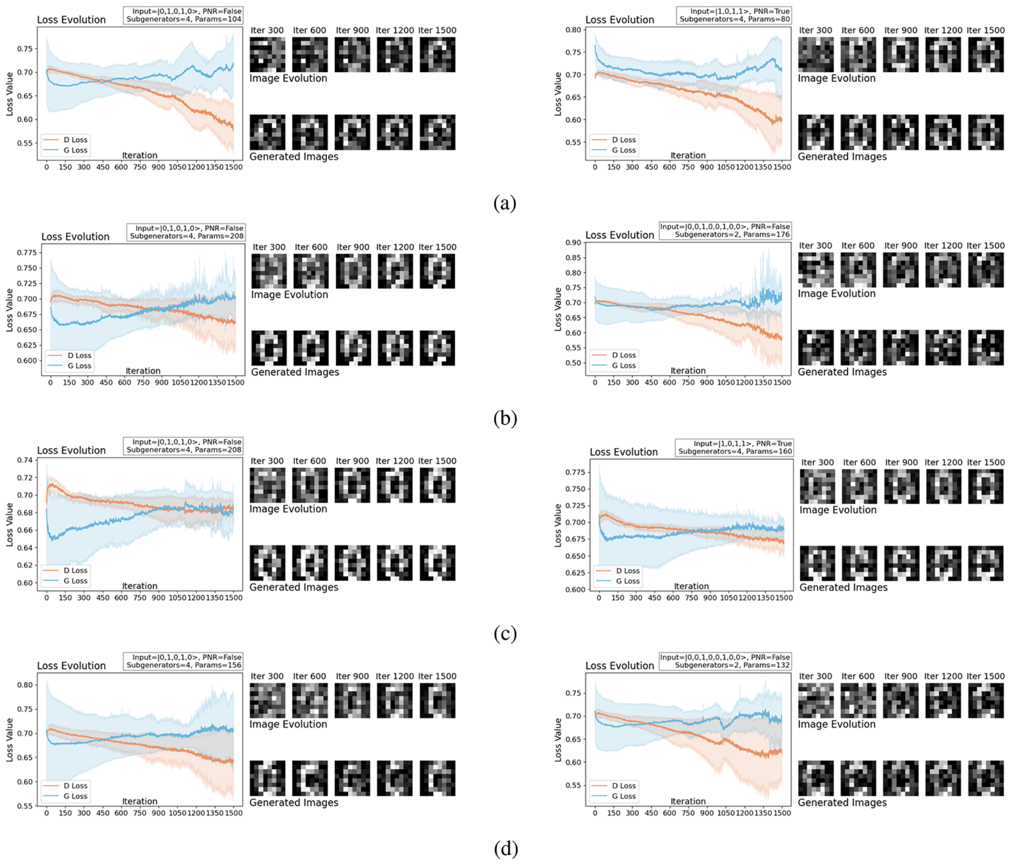

Fig. 6. Best training configurations for: (a) circuit setup A; (b) circuit setup B; (c) circuit setup C; (d) circuit setup D. Each panel contains a plot with the evolution of the loss function, the evolution of the generated images, as well as the final generated images of the trained model, and a small infobox detailing the model hyperparameters. Panels on the left and right correspond to different sets of hyperparameters.

metrics allow to evaluate a model's capability to learn well from data instead of memorizing it.

A common choice of measures of similarity and diversity are the inception score (IS) [40] and Fréchet inception distance (FID) [41], which measure both the quality and the diversity of generated images. However, for a simple model working with lower dimensional images, such as ours, a simpler metric suffices. For this purpose, we chose the structural similarity index measure (SSIM) [42], which works by weighing in the comparative properties of luminance (l), contrast (c), and structure (s), defined as follows for patches x and y of images to be compared:

$$l(x,y)=\frac{2\mu_x\mu_y+c_1}{\mu_x^2+\mu_y^2+c_1},\quad c(x,y)=\frac{2\sigma_x\sigma_y+c_2}{\sigma_x^2+\sigma_y^2+c_2},\quad s(x,y)=\frac{\sigma_{xy}+c_3}{\sigma_x\sigma_y+c_3},$$

where μ and σ are correspondingly the mean and the standard deviation of pixel intensity for the given patch, σ_{xy} is the covariance between patches x and y, and c_1, c_2, and c_3 are stabilizing constants, preventing a division by a small value. With these notions in mind, SSIM is defined as

$$\mathrm{SSIM}(x,y)=l(x,y)\cdot c(x,y)\cdot s(x,y)=\frac{(2\mu_x\mu_y+c_1)\,(2\sigma_{xy}+c_2)}{\left(\mu_x^2+\mu_y^2+c_1\right)\left(\sigma_x^2+\sigma_y^2+c_3\right)}.$$

SSIM compares large images patch by patch and then averages the metrics across the patches; however, for our case, due to the small sizes of images, it compares them immediately. SSIM varies in the range $[-1,1]$, with values close to 1 indicating similarity, 0 indicating absence of similarity, and values close to -1 indicating anticorrelation between images.

To evaluate the models, we compute the similarity score, which computes a pairwise SSIM of 500 random real images from the training dataset against 500 images generated by the quantum generator and then average across all the images. We also compute the the diversity score: first we compute the pairwise SSIM among the generated images, and again average across all images. This value is then subtracted from 1 to follow the logic of high diversity corresponding to more diverse (rather than more similar) images.

To have a baseline for comparison, we train a small GAN with a classical generator, which has approximately the same number of parameters as the largest generator considered in this work (around 330). This model is one of the simplest possible generators with only one small hidden layer, built in a way to allow for a fair comparison with quantum analog, with the training hyperparameters identical to those used for quantum generators. We note that a classical generator like this is heavily underparameterized and would not normally be used in a setting other than a comparative one; however, it mimics well the behavior of small quantum models discussed here.

We plot the similarity against the diversity for all the models tested, along with corresponding scores of the classical model on a scatter plot presented in Fig. 7. First, we note that diversity and similarity are anticorrelated for small models discussed here. In addition, the quantum model with the highest similarity

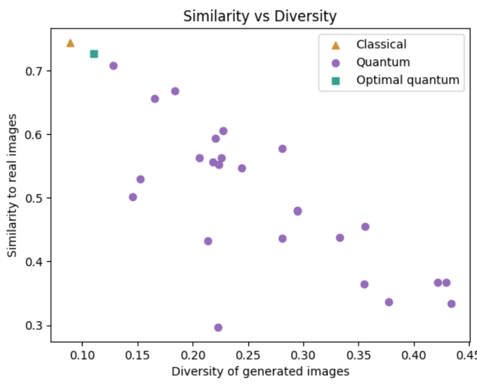

Fig. 7. Similarity plotted against diversity of generated images.

(highlighted as a square, corresponding to Fig. 6, left) has a performance comparable to that of the classical model (highlighted as a triangle), with slightly worse similarity and a slightly better diversity. Moreover, the quantum generator has approximately 2/3 parameters of the classical one (208 versus 332).

Visual inspection confirms that indeed the results are of generally high quality for this model, albeit not very diverse. We thus choose this model for further testing, since it compares the best to the classical alternative. We refer the reader to Section S2 of the Supplement 1 for further analysis of the results and for hypotheses as to why some configurations perform better than others.

We apply the model from the left-hand side of Fig. 6 (c) to the generation of other digits. We present results for some digits in Fig. 8, while the rest of the digits can be found in the Github repository [39]. An additional 500 iterations were employed here to properly assess the convergence for different digits. When comparing to target data, we see that the model performs fairly well for most of the digits, and each sampled digit has recognizable contours. Importantly, the model either converges or is bound to do so for all the digits, which is indicated by a narrowing of the standard deviation for the L_G loss. While some digits may require more training iterations to reach an equilibrium state, we can assert that the model starts producing realistic results around iteration 1500.

Note that, for simplicity, we restricted our model to generating one digit at a time (unlike classical GANs trained on MNIST). This is a similar approach to previous QGAN proposals such as [26], but our model could be extended to a conditional QGAN in future work. It would then be able to generate all the digits depending on the input label, which could be supplied as a model input or encoded through an additional layer.

3.5. Noisy Simulations

We now use the same model for noisy simulations. The *Perceval* package allows us to specify various parameters such as the emission probability of the source, the photon loss regime, and photon distinguishability. We set indistinguishability and photon loss to 0.92 each, so as to closely mimic the actual conditions on the *Ascella* processor as it was presented in [9].

In addition to the imperfect source and losses throughout the circuit, noisy simulations also introduce sampling error. For ideal simulations, the distribution of the output with exact probabilities is directly available in *Perceval*. However, in this case, the distribution is obtained by collecting 10^5 measurement shots and postselecting lossy outputs. A discrete distribution obtained in such a way gets closer to the exact distribution with an increasing number of measurement shots. However, in an experimental setting, collecting a high number of shots requires precious time, and 10^5 shots were found to be an optimal compromise for the accuracy/training time trade-off.

As discussed in Section 3.1, the lack of PNR detectors combined with photon loss shrink the size of the output space. This requires us to change some hyperparameters and we display two strategies in Fig. 9.

Clearly, the added noise considerably slows down the convergence. While the noiseless version reaches an equilibrium around iteration 900 on average, for the noisy version on the right of Fig. 9, even 1500 iterations are not enough. This version, which has more sub-generators, shows a diverging learning trend. This might be caused by the increased number of parameters, making it harder to train this model under the predefined learning rate restrictions (see Supplement 1, Section S2). Nevertheless, for the model on the left which has less sub-generators but a higher number of modes, one can see how the standard deviation of the generator decreases throughout the training, indicating that it is likely to eventually converge. Moreover, the generated results are satisfactory when applying a manual check. Results comprehensible to the human eye are available by iteration 600 for the ideal version, while the noisy version requires more than 1000.

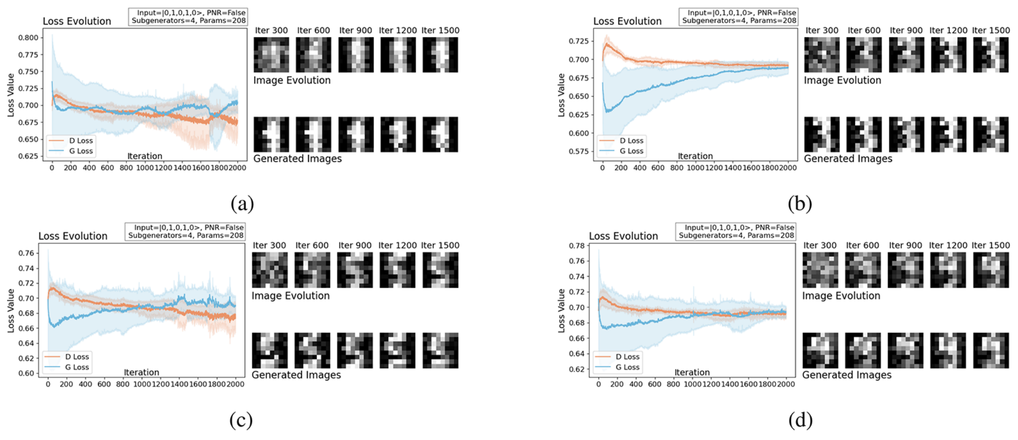

Fig. 8. Training results for different digits: (a) digit 1; (a) digit 3; (a) digit 5; (a) digit 9.

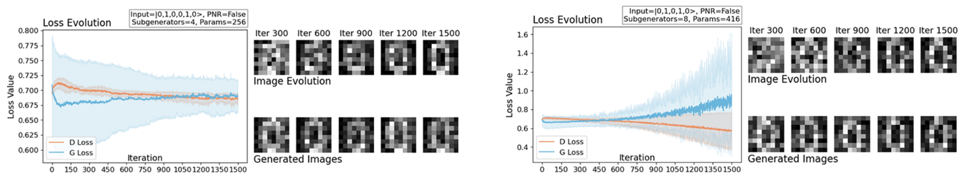

Fig. 9. Training results for noisy simulations. The model on the left has a higher number of modes and the model on the right contains more sub-generators, which compensates for a smaller output space in the lossy case.

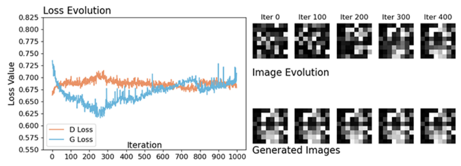

Fig. 10. Training results for the experiment on the *Ascella* quantum processor.

Overall, training in the presence of noise and sampling errors comparable to those of a real quantum processor is still viable, albeit slower. Techniques for quantum error mitigation [43] might improve the results of noisy simulations and could be explored in future work.

3.6. Physical Experiment

Based on the insights gathered from our simulations, we run an experiment on Quandela's processor *Ascella*, with our best-performing model. The experimental setup is the following. A gated InGaAs quantum dot placed inside a micropillar cavity [44] is excited by a laser as in the scheme of [45] to produce single photons on demand. A demultiplexer converts the train of produced photons into single photons arriving simultaneously on the chip, which is a 12-mode interferometer designed according to the Clements scheme [46]. The phase shifters on the chip are tuned thermo-optically to apply the intended operations, and this process is optimized following the characterization procedure of [47]. The photons then enter superconducting nanowire single-photon detectors (SNSPDs), which are threshold detectors. As mentioned in the previous section, the setup is affected by photon loss and distinguishability.

The circuit that corresponds to our best-performing model is of reasonable depth, making it compatible with the *Ascella* chip, which is naturally reused for each sub-generator. To make the duration of the experiment tractable, we decrease the total number of training iterations to 1000, with three SPSA steps for the generator at each iteration, which results in a total of 3000 SPSA steps. The results for one training instance are displayed in Fig. 10.

While the training is slow, following the trend of noisy simulations, the results are promising. Importantly, we observe that the model is learning, with the loss functions behaving as they should, tending toward the equilibrium value, and results improving throughout the training. A point that would require improvements in future experiments is that the generated images do not present much diversity. There are only minor differences between the "0" terms which might be improved by adding more encoding phase shifters into the ansatz. Nevertheless, the generated "0" terms are of fairly good quality, with characteristic contours of the digit, demonstrating the experimental feasibility of photonic QGANs.

4. DISCUSSION AND FUTURE WORK

In this work, we showed that photonic quantum circuits can work as a key component in a generative learning pipeline to produce images. This is in contrast to previous work where linear optical circuits were used for smaller-scale tasks such as latent space generation, without going as far as using a photonic generator. To the best of our knowledge, our experiment is the first demonstration of a photonic GAN with a fully quantum generator for classical data. Additionally, with the aim of transparency and collaboration within the quantum machine learning community, we make our code available online at [39].

Most of the QGAN literature concentrates on smaller scale models for generation of lower resolution images, such as 3×3 bars-and-stripes. However, the task of generating larger images is tractable with currently available hardware if the right approach is used—such as our distribution-based mapping. This highlights the importance of looking for alternative strategies when working with near-term quantum hardware. Using our mapping along with noise reuploading gave us greater flexibility and, in combination with patch-based learning, the model could generate higher resolutions images.

In terms of scaling, we point out that our approach requires the approximation of the output probability distribution and may thus have a high sampling cost, like many quantum machine learning models. Photonics at least has the benefit of providing fast sampling which improves runtime, a strength which will become more apparent as levels of photon loss decrease with the advancement of the technology. However, beyond a certain dimension of the problem, it might be necessary to modify the approach, for instance, using our model together with a VAE that transforms the data.

We note that while the size of the Fock space grows exponentially with the number of modes and photons, it is also directly linked to the dimension of the target data, given our mapping. Many problems of interest might thus be reachable with a reasonably sized device. When comparing the native photonic approach to the qubit-based approach of [26], we see how, in terms of resources used, on photonic platforms, it may be advantageous to use the Fock space over the Hilbert space. As we saw above, a model can be built which uses five modes and two photons for the generation of 8×8 images, with the input state $|0, 1, 0, 1, 0\rangle$, as compared with eight modes and four photons, which would be needed in dual-rail encoding to have qubits as in [26].

Concerning the optimization, we note that SPSA seems to work surprisingly well on variational linear optical circuits. Especially in the noisy experimental setup, SPSA allowed for stable learning at a constant cost, albeit requiring more iterations to converge. Fine-tuning the optimization and the model, in general, was an important aspect of the work. Future work may concentrate on improving the convergence rates of optimization techniques for photonic platforms, which can be used directly to improve the performance of our model. Indeed, only few works so far have focused on exploring methods like parameter-shift rules [48] in the context of quantum photonics [49–51]. As mentioned in the text, other improvements to this work include the extension of the model to a conditional QGAN, as well as the integration of error mitigation techniques for sources of noise such as photon loss and distinguishability [52].

Overall, our QGAN implementation is quite flexible and modular, so the model can be further explored for generation of other types of classical data, by fine-tuning parameters and introducing some changes to the circuit structure. For instance, a generator may consist of sub-generators with different structures, depending on the task at hand. We could also consider a quantum discriminator and training data based on a quantum source, for the generation of quantum states, for instance. Additionally, the source of the latent space could also be changed to be quantum, by using another boson-sampling-based circuit as a source of noise, as in [23].

In terms of applications, it is not always clear for which problems a quantum model will perform best—a recent study showed that several popular quantum classifiers do not outperform their classical counterparts on standard datasets [53]. A solution that is often proposed is to focus on quantum data instead, e.g., to generate quantum states. However, problems containing classical data are overall much more common. There, the concept of inductive bias proves very helpful to follow, requiring that the model contain some information about the structure of the problem at hand. For this reason, high-energy physics and quantum chemistry are often cited as areas where quantum models might be most useful on classical data. For example, in [54], a QGAN is trained to detect anomalous events that cannot be described by the standard model; in [55], a QGAN was used to generate calorimeter shower images; and in [56,57], QGANs and QCBMs were used for the generation of new molecules.

We saw that our best-performing generator achieves similar performance to its classical counterpart for a lower number of trainable parameters. Such benchmarks can be interesting to focus on, as an alternative to speedups. In the same spirit, the energy consumption of quantum versus classical computers are currently being explored [58].

Funding. Agence Nationale de la Recherche (Plan France 2030 OQuLus ANR-22-PETQ-0013); European Commission: EIC Quondensate (101130384).

Acknowledgment. We would like to thank Pierre-Emmanuel Emeriau for fruitful discussions and feedback on the manuscript.

Disclosures. The authors declare no conflicts of interest.

Data availability. Data from the numerical simulations and the experiment can be reproduced by the reader using the companion GitHub repository [39] and access to Quandela's cloud platform [59].

Supplemental document. See Supplement 1 for supporting content.

REFERENCES

1. J. Bourassa, R. Alexander, M. Vasmer, *et al.*, "Blueprint for a scalable photonic fault-tolerant quantum computer," Quantum **5**, 392 (2021).
2. R. Raussendorf and H. J. Briegel, "A one-way quantum computer," Phys. Rev. Lett. **86**, 5188–5191 (2001).
3. S. Bartolucci, P. Birchall, H. Bombín, *et al.*, "Fusion-based quantum computation," Nat. Commun. **14**, 912 (2023).
4. S. Aaronson and A. Arkhipov, "The computational complexity of linear optics," in *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing, STOC '11* (Association for Computing Machinery, 2011), pp. 333–342.
5. J. Preskill, "Quantum computing in the NISQ era and beyond," Quantum **2**, 79 (2018).
6. S. I. Davis, A. Mueller, R. Valivarthi, *et al.*, "Improved heralded single-photon source with a photon-number-resolving superconducting nanowire detector," Phys. Rev. Appl. **18**, 064007 (2022).
7. B. Y. Gan, D. Leykam, and D. G. Angelakis, "Fock state-enhanced expressivity of quantum machine learning models," EPJ Quantum Technol. **9**, 16 (2022).
8. K. Bartkiewicz, C. Gneiting, A. Černoch, *et al.*, "Experimental kernel-based quantum machine learning in finite feature space," Sci. Rep. **10**, 1 (2020).
9. N. Maring, A. Fyrillas, M. Pont, *et al.*, "A versatile single-photon-based quantum computing platform," Nat. Photonics **18**, 603–609 (2024).
10. J. Tian, X. Sun, Y. Du, *et al.*, "Recent advances for quantum neural networks in generative learning," IEEE Trans. Pattern Anal. Mach. Intell. **45**, 12321–12340 (2023).
11. A. Sehrawat, "Interferometric neural networks," (2023).
12. M. Cerezo, A. Arrasmith, R. Babbush, *et al.*, "Variational quantum algorithms," Nat. Rev. Phys. **3**, 625–644 (2021).
13. L. Deng, "The MNIST database of handwritten digit images for machine learning research," IEEE Signal Process Mag. **29**, 141–142 (2012).
14. I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial networks," Commun. ACM **63**, 139–144 (2020).
15. D. P. Kingma and M. Welling, "Auto-encoding variationalBayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, eds. (2014).
16. T. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, Vol. 33 H. Larochelle, M. Ranzato, R. Hadsell, *et al.*, eds. (Curran Associates, Inc., 2020), pp. 1877–1901.
17. J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, *et al.*, "Deep unsupervised learning using nonequilibrium thermodynamics," in *Proceedings of the 32nd International Conference on Machine Learning*, Vol. 37 of *Proceedings of Machine Learning Research* F. Bach and D. Blei, eds. (PMLR, 2015), pp. 2256–2265.
18. L. Ruthotto and E. Haber, "An introduction to deep generative modeling," GAMM-Mitteilungen **44**, e202100008 (2021).
19. F. Farnia and A. Ozdaglar, "Do GANs always have Nash equilibria?" in *Proceedings of the 37th International Conference on Machine Learning*, Vol. 119 of *Proceedings of Machine Learning Research* H. D. Daumé III and A. Singh, eds. (PMLR, 2020), pp. 3029–3039.
20. S. Lloyd and C. Weedbrook, "Quantum generative adversarial learning," Phys. Rev. Lett. **121**, 040502 (2018).
21. P.-L. Dallaire-Demers and N. Killoran, "Quantum generative adversarial networks," Phys. Rev. A **98**, 012324 (2018).
22. M. S. Rudolph, N. B. Toussaint, A. Katabarwa, *et al.*, "Generation of high-resolution handwritten digits with an ion-trap quantum computer," Phys. Rev. X **12**, 031010 (2022).
23. H. Wallner and W. R. Clements, "Towards an inductive bias for quantum statistics in GANs," in *ICLR 2023 Workshop on Physics for Machine Learning* (2023).
24. Y. Wang, S. Xue, Y. Wang, *et al.*, "Quantum generative adversarial learning in photonics," Opt. Lett. **48**, 5197 (2023).
25. C. Zoufal, A. Lucchi, and S. Woerner, "Quantum generative adversarial networks for learning and loading random distributions," npj Quantum Inf **5**, 103 (2019).
26. H.-L. Huang, Y. Du, M. Gong, *et al.*, "Experimental quantum generative adversarial networks for image generation," Phys. Rev. Appl. **16**, 024051 (2021).
27. H. Situ, Z. He, Y. Wang, *et al.*, "Quantum generative adversarial network for generating discrete distribution," Inf. Sci. (N. Y.) **538**, 193–208 (2020).

28. J. Romero and A. Aspuru-Guzik, “Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions,” Adv. Quantum Technol. **4**, 2000003 (2021).
29. H. Ma, L. Ye, F. Ruan, *et al.*, “Quantum generative adversarial networks in a silicon photonic chip with maximum expressibility,” arXiv (2024).
30. A. Salavrakos, T. Sedrakyan, J. Mills, *et al.*, “An error-mitigated photonic quantum circuit born machine,” arXiv (2024).
31. S. Shankar and D. Towsley, “Variational boson sampling,” in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (Springer, 2022).
32. J. Shi, Y. Tang, Y. Lu, *et al.*, “Quantum circuit learning with parameterized boson sampling,” IEEE Trans. Knowl. Data Eng. **35**, 1965–1976 (2023).
33. S. Chaudhary, P. Huembeli, I. MacCormack, *et al.*, “Towards a scalable discrete quantum generative adversarial neural network,” Quantum Sci. Technol. **8**, 035002 (2023).
34. A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, *et al.*, “Data re-uploading for a universal quantum classifier,” Quantum **4**, 226 (2020).
35. J. Spall, “Implementation of the simultaneous perturbation algorithm for stochastic optimization,” IEEE Trans. Aerosp. Electron. Syst. **34**, 817–823 (1998).
36. M. Wiedmann, M. Hölle, M. Periyasamy, *et al.*, (2023).
37. F. Sauvage and F. Mintert, “Optimal quantum control with poor statistics,” PRX Quantum **1**, 020322 (2020).
38. N. Heurtel, A. Fyrillas, G. de Gliniasty, *et al.*, “Perceval: a software platform for discrete variable photonic quantum computing,” Quantum **7**, 931 (2023).
39. T. Sedraykyan, "Photonic QGAN," GitHub, 2024, https://github.com/Quandela/photonic-qgan.
40. T. Salimans, I. Goodfellow, W. Zaremba, *et al.*, “Improved techniques for training GANs,” in *Advances in Neural Information Processing Systems*, Vol. 29 D. Lee, M. Sugiyama, U. Luxburg, *et al.*, eds. (Curran Associates, Inc., 2016).
41. M. Heusel, H. Ramsauer, T. Unterthiner, *et al.*, “GANs trained by a two time-scale update rule converge to a local Nash equilibrium,” in *Advances in Neural Information Processing Systems*, Vol. 30 I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, eds. (Curran Associates, Inc., 2017).
42. Z. Wang, A. Bovik, H. Sheikh, *et al.*, “Image quality assessment: from error visibility to structural similarity,” IEEE Trans. on Image Process. **13**, 600–612 (2004).
43. S. Endo, Z. Cai, S. C. Benjamin, *et al.*, “Hybrid quantum-classical algorithms and quantum error mitigation,” J. Phys. Soc. Jpn. **90**, 032001 (2021).
44. N. Somaschi, V. Giesz, L. De Santis, *et al.*, “Near-optimal single-photon sources in the solid state,” Nat. Photonics **10**, 340–345 (2016).
45. S. E. Thomas, M. Billard, N. Coste, *et al.*, “Bright polarized single-photon source based on a linear dipole,” Phys. Rev. Lett. **126**, 233601 (2021).
46. W. R. Clements, P. C. Humphreys, B. J. Metcalf, *et al.*, “Optimal design for universal multiport interferometers,” Optica **3**, 1460–1465 (2016).
47. A. Fyrillas, O. Faure, N. Maring, *et al.*, “Scalable machine learning-assisted clear-box characterization for optimally controlled photonic circuits,” Optica **11**, 427–436 (2024).
48. M. Schuld, V. Bergholm, C. Gogolin, *et al.*, “Evaluating analytic gradients on quantum hardware,” Phys. Rev. A **99**, 032331 (2019).
49. G. de Felice and C. Cortlett, “Differentiation of linear optical circuits,” arXiv (2024).
50. A. Pappalardo, P.-E. Emeriau, G. de Felice, *et al.*, “A photonic parameter-shift rule: enabling gradient computation for photonic quantum computers,” arXiv, 2024, https://arxiv.org/abs/2410.02726.
51. G. Facelli, D. D. Roberts, H. Wallner, *et al.*, “Exact gradients for linear optics with single photons,” arXiv, 2024, https://arxiv.org/abs/2409.16369.
52. S. Endo, S. C. Benjamin, and Y. Li, “Practical quantum error mitigation for near-future applications,” Phys. Rev. X **8**, 031027 (2018).
53. J. Bowles, S. Ahmed, and M. Schuld, “Better than classical? the subtle art of benchmarking quantum machine learning models,” arXiv (2024).
54. E. Bermot, C. Zoufal, M. Grossi, *et al.*, “Quantum generative adversarial networks for anomaly detection in high energy physics,” in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, 2023).
55. F. Rehm, S. Vallecorsa, M. Grossi, *et al.*, “A full quantum generative adversarial network model for high energy physics simulations,” arXiv (2024).
56. M. G. Vakili, C. Gorgulla, A. Nigam, *et al.*, “Quantum computing-enhanced algorithm unveils novel inhibitors for KRAS,” arXiv (2024).
57. P.-Y. Kao, Y.-C. Yang, W.-Y. Chiang, *et al.*, “Exploring the advantages of quantum generative adversarial networks in generative chemistry,” J. Chem. Inf. Model. **63**, 3307–3318 (2023).
58. A. Auffèves, “Quantum technologies need a quantum energy initiative,” PRX Quantum **3**, 020101 (2022).
59. https://cloud.quandela.com/