Collaborative generative quantum machine learning on a quantum computer

Karol Bartkiewicz,¹,2 Patrycja Tulewicz,¹ Jan Roik,¹ and Karel Lemr²

¹Institute of Spintronics and Quantum Information, Adam Mickiewicz University, PL-61-614 Poznań, Poland
²Joint Laboratory of Optics of Palacký University and Institute of Physics of Czech Academy of Sciences, 17. listopadu 12, 771 46 Olomouc, Czech Republic

We introduce a new approach towards generative quantum machine learning and report on a proof-of-principle experiment demonstrating our approach. Our proposal depends on collaboration between the generators and discriminator, thus, we call it quantum collaborative generative learning. We present numerical evidence that the collaborative approach compares favorably to recently proposed quantum generative adversarial learning. In addition to the results obtained with quantum simulators, we also present experimental results obtained with an actual programmable quantum computer. We investigate how a quantum computer implementing generative learning algorithm could learn the concept of a Bell state. After completing the learning process the network is able both to recognize and to generate an entangled state. This is the first step towards understanding how the concept of quantum entanglement could be learned and demonstrated by a quantum computer.

INTRODUCTION

Generative adversarial network (GAN) machine learning is an intensely studied topic in the field of machine learning and artificial intelligence research [1]. While quantum machine learning research is attracting increasingly more attention both from the industry and the scientific community [2–27], the quantum counterparts of GANs have been proposed in several recent papers works [28–30]. For example, in the proposal put forward by Dallaire-Demers and Killoran in Ref. [29], the authors put much attention to specific circuit ansatz and discuss methods of computing gradients in specific types of variational quantum circuits. It is worth noting that the problem of computing gradients for variational quantum circuits is rather complex and can be also achieved by the parameter-shift rule [31, 32]. In its general form, the proposal of Ref. [29] includes sources of entropy (i.e., bath).

The idea behind GANs is rather simple, and it can be described with three circuits. The first circuit is the generator of real data \( \mathcal{R} \), which is in principle an irreversible transformation depending on a value of a random variable \( z_R \). This transformation at each instance takes the standard input state \( |0\rangle \) and outputs a labeled random state \( \rho_\lambda \). A good example of such a generator is a painter who is asked to draw a cat (the label \( \lambda \) is the animal here). There is not a unique deterministic way of drawing a cat, nor we know how to construct a painter from basic elements. However, we can train a stochastic quantum machine \( \mathcal{G} \) to perform as generator \( \mathcal{R} \) the same task by observing the output of \( \mathcal{R} \) and its labels. However, this is not enough because \( \mathcal{G} \) trained in this way, in general, will not be able to create new original instances, which can be labeled as \( \lambda \). Hence, an additional circuit \( \mathcal{D} \) needs to be considered. This circuit is trained to distinguish between the samples \( \rho_\lambda \) and the random output of \( \mathcal{G} \), and it is referred to as discriminator.

The operation of \( \mathcal{D} \) is optimal, if it assigns value 0 to states generated by \( \mathcal{R} \) and value 1 to states generated by \( \mathcal{G} \). At the same time, the operation of \( \mathcal{G} \) is optimal, if the cross-entropy between its output and states \( \rho_\lambda \) is minimal while the discriminator is most likely to assign value 0 to the output of \( \mathcal{G} \). Thus, a GAN problem is solved by adversarial training of \( \mathcal{D} \) versus \( \mathcal{G} \). The parameters of both the generator and discriminator can be found by numerical optimization or quantum gradient evaluation [29] by dividing the training into rounds of adversarial optimization of both generator and discriminator. The circuits can perform an arbitrary nonlinear computation as long as they are complex enough, admitting an arbitrary unitary operation and measurements on a number of ancillary qubits. However, similarly to classical artificial neural networks, choosing the appropriate architectures for specific concerns is a complex problem which is solved by trial and error. In quantum computing, this is even more so, because the lack of practical error correction limits the complexity of quantum circuits.

The quantum counterpart of the GAN (i.e., QGAN) learning finds Nash equilibrium of two player game, where one of the players generates some output and the second player (discriminator, \( \mathcal{D} \)) tries to tell if the output is generated by the first player (generator \( \mathcal{G} \)) or provided by an external source (\( \mathcal{R} \)). This could be expressed as a min-max problem, where the statistical distance between the outputs of \( \mathcal{G} \) and \( \mathcal{R} \) is minimized over the strategies of the generator, whereas the distance between the outputs of \( \mathcal{D} \) for \( \mathcal{G} \) and \( \mathcal{R} \), respectively, is maximized over the possible strategies of discriminator at the same time. In practice, this type of optimization if performed in rounds, and it is difficult to make the learning process stable.

To avoid this issue, we put forward a new kind of ma-
The task of the discriminator is to decide, for every input, if the input was indeed provided by generator \( \mathcal{R} \) or not. The discriminator is trained only on a limited, but large, number of states \( \rho_\lambda \) and their labels. Note that in the classical ML the random variable is needed for the discriminator to make a decision if its input is real or fake, if the fakes are indistinguishable from the real inputs. In the quantum case, this is not necessary as the collapse of a wave function of the discriminator output will achieve the same effect.

The third component is the circuit that is the model circuit of our generator \( \mathcal{G} \) to be trained. This generator processes the same type of input as generator \( \mathcal{R} \) and is provided with an independent random variable \( z_G \). We denote the output of this circuit \( \sigma = |\psi_{\lambda,z_G}\rangle\langle\psi_{\lambda,z_G}| \). The action of the generator is reversible as long as we know the value of the random variable \( z_G \). We assume that this is the case as this is a classical variable. We do not expect \( \mathcal{G} \) to mimic the operation of generator \( \mathcal{R} \), as both their outputs are random. Our aim is to minimize the relative-entropy of their outputs, given as

\[
S(\sigma_{\lambda,z_G}||\rho_{\lambda}) = \text{Tr}(\sigma_{\lambda,z_G}^{\dagger} - \text{Tr}(\sigma_{\lambda,z_G}^{\dagger} \log \sigma_{\lambda,z_G} \rho_{\lambda}))
\]  

or in terms of Newton–Mercator series as

\[
S(\sigma_{\lambda,z_G}||\rho_{\lambda}) = (1 - \rho_{\lambda}) + \left( (1 - \rho_{\lambda})^2 / 2 + (1 - \rho_{\lambda})^3 / 3 + \ldots \right)
\]  

and

\[
\langle \psi_{\lambda,z_G} | \sigma_{\lambda,z_G} | \psi_{\lambda,z_G} \rangle = (|0\rangle_{z_G} \langle 0|_{z_G})_{\lambda,z_G} \sigma_{\lambda,z_G} |0\rangle_{z_G} \langle 0|_{z_G}
\]

This quantity can be measured experimentally and related to the cost function of the GAN training, which depends also on the correct performance of the discriminator.

In the case of the generator \( \mathcal{G} \) training, one minimizes the distance between states sampled from \( \mathcal{G} \) and \( \mathcal{R} \). This could be viewed as minimizing linear relative entropy between the output of the discriminator in the two respective cases, i.e.,

\[
S_L[\eta_1(\sigma_{\lambda,z_G}) || \eta_1(\rho_{\lambda,z_R})] = 1 - \text{Tr}[\eta_1(\sigma_{\lambda,z_G}) \eta_1(\rho_{\lambda,z_R})]
\]

where \( \eta_1(\rho) = \text{Tr}_n(\mathcal{D}(0) | 0 \rangle \langle 0| \otimes \rho^{\dagger}) \), where \( \mathcal{D} \) is designed to imprint the decision probability for \( \mathcal{G} \) or \( \mathcal{R} \) on the amplitude of |0\rangle or |1\rangle of the decision qubit, marked as subsystem initialized as |0\rangle state. The partial trace is performed after the decision qubit for \( n = 1 \). For \( n = 2 \), we trace over the generator mode.

However, when training the discriminator \( \mathcal{D} \), we can redefine this problem as minimizing

\[
S_L[\eta(\sigma_{\lambda,z_G}) || \eta(\rho_{\lambda,z_R})] = 1 - \text{Tr}[\eta_2(\sigma_{\lambda,z_G}) \eta_2(\rho_{\lambda,z_R})]
\]

QUANTUM COLLABORATIVE GENERATIVE NETWORK

For our approach to work, we consider only reversible (unitary) discriminators \( \mathcal{D} \), which are provided with a generated state \( \rho_\lambda \), its label \( \lambda \), a random variable \( z_D \), and a large enough ancillary Hilbert space to enable complex quantum computations. The task of the discriminator is
where $\eta_2(\rho) = X \text{Tr}_2(D|0\rangle\langle 0| \otimes \rho D^+)X$, and $X$ is the bit-flip operation acting on the decision qubit. The learning process is complete when both $\text{Tr}[\eta_2(\sigma_{X,zG}X)\eta_2(\rho_{\lambda,zR}X)]$ and $\text{Tr}(\sigma_{Y,zG}\rho_{\lambda,zR})$ are maximized simultaneously (i.e., one cannot be increased without decreasing the other).

In the quantum case, i.e., for generating a specific pure state, both values are at most 1, and these values can be reached simultaneously.

It is possible to introduce a figure of merit that combines the two linear entropy functions to obtain the collaborative cost function

$$J = \frac{1}{2}(S_L[\eta_1(\sigma_{X,zG})||\eta_1(\rho_{\lambda,zR})] + S_L[\eta_2(\sigma_{Y,zG})||\eta_2(\rho_{\lambda,zR})]).$$ \hfill (6)

The linear combination of the two functions does not have to be balanced. In fact, both can be considered as a condition under which the other is minimized. However, because of this symmetry, we decided not to write the Lagrange multipliers directly.

As we demonstrate below, such a function can be measured directly in a single circuit. To this end, we propose connecting conjugated circuits to form a circuit that has $D$ interfaced with its reverse of $D$ with a conditional $X$-gate in between (i.e., Pauli $\sigma_x$ operation) in the first mode as depicted in Fig. [1]. To reduce the complexity of this circuit, let us note that the labels marking the class to which a given state belongs to can be purely classical. This means that generator $G$ and discriminator $D$ can be controlled by a classical variable $\lambda$, which simplifies the quantum circuit from Fig. [1] to the one depicted in Fig. [1]. Note that the middle (generator) mode in Fig. [1] can in general represent an arbitrary number of modes, i.e., $\rho$ and $\sigma$ can be of arbitrary large Hilbert space.

The circuit can be considered as working in two regimes, depending on detecting $|0\rangle$ or $|1\rangle$ in the last mode in Fig. [1]. In the latter case, the linear relative-entropy between the generators can be measured by feeding states $\rho_{\lambda,zG}$ to the circuit and for the fixed values of $\lambda$ and $z_G$, and consecutively measuring the rate at which the state of the generator line of the circuit is projected on $|0\rangle$. However, this is only the case if the reversible discriminator returns $|1\rangle$ for a state generated by $G$ and $|0\rangle$ for a state provided by $R$. The probability of this process is proportional to the rate at which the top line is projected onto $|0\rangle$. Given that the top mode is projected onto $|0\rangle$, the middle line measures the linear cross-entropy. In the opposite case (the decision mode is detected to be in $|1\rangle$), the operation of the discriminator failed to be reversed and the detection rates of the middle line are meaningless. Hence, both the discriminator and the generator work at their best, if the joint detection rates of $|0\rangle$ in both top-most circuit modes in Fig. [1] are maximized simultaneously. This is why we refer to the learning process as collaborative. However, there exist solutions to this optimization problem, where the generator $G$, taken separately from the discriminator, does not perform similarly to $R$. To address this issue, we consider the regime where only the similarity between $G$ and $R$ is maximized ($|0\rangle$ detected in the third mode in Fig. [1]). More generally, we could consider the collaborative learning as a process where both $D$ and $G$ are trained cooperatively, under the condition that $G$ also is improving separately.

To optimize the performance of the quantum setup, we propose to update its parameters using the Nelder-Mead algorithm or gradient descent to minimize the cost function $J$.

To consider a possible ansatz for the discriminator, let us again consider the regime, where the $X$ operation is active in the decision mode. While maximizing the detection rates for $|0\rangle$ in the generated state mode by varying the parameters of generator $G$, we are making it less likely to detect $|0\rangle$ in the decision line. If the operations of $G$ and $R$ are identical, then gate $X$ will flip the top qubit and could not achieve maximal two-fold detection rates of $|0\rangle$ in both modes, unless we allow $D$ to become a Hadamard gate $H$, conditioned on the similarity of $R$ and $G$ circuits. Note that, while maximizing the detection rates of $|0\rangle$ in the decision line by varying the parameters of the discriminator $D$, in general, we do not necessarily decrease the value of relative entropy. If during the training the discriminator becomes a separable operation similar to $\sqrt{H} \otimes 1$, and the generator $G$ is very close to operating as $R$. Then, by optimizing $G$ even further we would not influence the detection rate in the top mode, i.e., the discriminator stops learning. In fact, the detection rate stops varying with $G$ as soon as the operation $D$ becomes separable. This suggests that inseparability of $D$ is necessary to train the discriminator. Thus, it must be ensured during the design of $D$ that its outcome in the decision line is strongly correlated with the generator mode. This can be easily achieved by making the discriminator to consist of gates controlled by the generator output mode, targeting only the discriminator decision mode. The discriminator should also admit arbitrary unitary transformations between the controlled operations. This guarantees that the output of a discriminator is state-dependent and the optimization works as described above.

**EXAMPLE: MINIMAL CIRCUIT**

Now, let us apply the general circuit from Fig. [1] to the simplest case, where each circuit mode corresponds to a single qubit. In this case the generator $G_1$ can be an arbitrary single-qubit unitary matrix. Similarly, in this regime, the generator is a general two-qubit unitary gate [24]. Two equivalent circuits implementing this operation are depicted in Fig. [2].

When designing a dedicated quantum circuit it could be beneficial to to replace the $XU\dagger XU$ operation from Fig. [2] with a controlled-phase gate implementing a con-
controlled rotation $R(\hat{z}, \beta)$ around $z$-axis and single-qubit rotation $R(\hat{a}, \alpha)$ around an axis given by versor $\hat{a}$. Then, in order to train QGAN it is necessary to find parameters for which

$$XU^\dagger XU = e^{i\omega} R(\hat{a}, \alpha) R(\hat{z}, \beta) R^\dagger(\hat{a}, \alpha).$$

(7)

Any unitary operation can be expressed as a rotation and a phase shift, i.e.,

$$XU^\dagger XU = e^{i\omega} \left( \cos \frac{\delta}{2} - i \hat{n} \cdot \hat{\sigma} \sin \frac{\delta}{2} \right) = e^{i\omega} R(\hat{n}, \delta),$$

(8)

where

$$-2i e^{i\omega} n_j \sin \frac{\delta}{2} = \text{Tr}(XU^\dagger XU \sigma_j),$$

$$2e^{i\omega} \cos \frac{\delta}{2} = \text{Tr}(XU^\dagger XU),$$

(9)

and $\sigma_j$ for $j = x, y, z$ are the Pauli matrices. These equations allow to express the parameters of matrix $U$ in terms of $\omega, \hat{n}$, and $\delta$. Moreover, from the cyclic property of the trace we learn that $n_x = 0$ and $\omega = 0$.

Next, let us write

$$R(\hat{n}, \delta) = R(\hat{z}, \gamma) R(\hat{y}, \eta) R(\hat{z}, \delta) R^\dagger(\hat{y}, \eta) R^\dagger(\hat{z}, \gamma),$$

(10)

by using Pauli matrix algebra we can show that

$$n_x = \sin \eta \cos \gamma, \quad n_y = \sin \eta \sin \gamma, \quad n_z = \cos \eta,$$

(11)

which allows us to compute $\eta$ and $\gamma$ for any $\hat{n}$. Having $n_x = 0$, we get $\gamma = \pi/2$. We also conclude that $R(\hat{a}, \alpha) = R(\hat{z}, \pi/2) R(\hat{y}, \eta) \text{ and } \beta = \delta$. Similarly, we can express the remaining operators, i.e.,

$$A = R(\hat{n}_a, \delta_a), \quad B = R(\hat{n}_b, \delta_b), \quad G = R(\hat{n}_g, \delta_g),$$

(12)

and

$$n_{l,x} = \sin \eta_l \cos \gamma_l, \quad n_{l,y} = \sin \eta_l \sin \gamma_l, \quad n_{l,z} = \cos \eta_l,$$

(13)

for $l = a, b, g$. Now, for a given $\lambda$, we can select the training parameters of the discriminator to be $\theta_{D,\lambda} = (\delta, \eta, \delta_a, \eta_a, \gamma_a, \eta_b, \delta_b, \eta_b, \gamma_b)$. Similarly, for the generator the training parameters are $\theta_{G,\lambda} = (\delta_g, \eta_g, \gamma_g)$. After the training is completed and parameters $\theta_{D,\lambda}$ and $\theta_{G,\lambda}$ are found we can compute the explicit forms of $U, A, B$ (i.e., $D$) and $G$. Note that this has to be repeated for each $\lambda$ and that for a given $\lambda$ we can express operator $U$ as

$$U = \cos \frac{\phi}{2} - i (\hat{\sigma} \cdot \hat{\sigma}) \sin \frac{\phi}{2}.$$  

(14)

By solving Eq. (14) we obtain

$$m_x = 0, \quad m_y = n_x, \quad m_z = n_z, \quad \delta = -2\phi.$$  

(15)

As a summary, the concept of the dedicated minimal (in the sense of number controlled unitary gates) QCGN experiment is given in Fig. 2, where the controlled block is implemented as discussed above. As already mentioned, it consists of only one controlled phase gate and single-qubit rotations. Such experiment could be performed on the platform of linear optics (see, e.g., [34]). Moreover, dedicated circuits for generator output of higher dimensions can be constructed in a similar fashion by reducing the number of the controlled-unitary gates.

**EXPERIMENTAL SINGLE-QUBIT QCGN**

Let us consider a proof-of-principle experiment, where $\lambda$ labels the bases in which states are prepared. If $\lambda = x$ the generator prepares $\mathcal{R}$ at random state $(|0\rangle + |1\rangle)/\sqrt{2}$ or $(|0\rangle - |1\rangle)/\sqrt{2}$. The eigenstates of the remaining Pauli matrices $\sigma_y$ and $\sigma_z$ are prepared if $\lambda = y, z$. This in general requires feeding generators $\mathcal{R}$ and $\mathcal{G}$ with uncorrelated bivariate random variables $z_R$ and $z_G$ (baths), respectively. In addition, we require that the QCGN performs equally well for all combinations of values of the random variables. Let us train a QCGN with $\mathcal{R}$ set as a Hadamard matrix proceeded by $X^{\alpha_0}$ operation, i.e., $\lambda = x$. To make the training process more transparent, let us focus on the special case of $z_g = 0$, only $(|0\rangle + |1\rangle)/\sqrt{2}$ is generated by $\mathcal{G}$.

In the experiment we deal with finite numbers of shots, which can lead to random fluctuations in the measured values of the minimized cost function. To establish a sufficient number of shots we analyzed the impact of
FIG. 3. Shaded areas depict the results of 100 Monte Carlo simulations of QCGN training for a) noiseless gates and shot noise, b) and c) noise model inducing both shot-noise and a decoherence model for quantum processor ibmq_jakarta. Connected points represent for a) and b) mean over training processes, for c) real experiments performed as described in the main text. For the cost function $J$ the region associated with the noise model does not encapsulate the experimental data. This means that the noise model provided for the quantum processor by the vendor, could be inadequate for circuit depths of order 27. Note that this is not the case for real/fake state fidelity $F$ associated with the trained generator, probability $p$ ($q$) of a real (fake) state being recognized as real by the discriminator.

From our numerical simulations, it follows that for our specific problem the training to perform well already for about $10^5$ shots for about $10^2$ evaluations of the cost function. When using more than $10^6$ shots the performance of Nelder-Mead algorithm further improves, reaching 70 cost function evaluations needed to find the minimum of the cost function. The speed of the convergence of this algorithm for this particular problem can be slightly improved by choosing a larger initial simplex. The requirements on the number of function evaluations and the number of coincidences make it feasible to implement conjugated QCGN on a contemporary quantum computer. The results of the experiment are show in Fig. 3.

We performed our experiments on ibmq_jakarta quantum processor. Note that due to technical solutions used in IBMQ processors we cannot directly implement the circuit given in Fig. 2. The processors, physically implement controlled-phase gates, controlled-not gates, and single-qubit rotations. This results in a circuit that performs 27 steps (circuit depth 27, 3 qubit circuit) before evaluating the cost function $J$. Independent 3 experiments were used to measure 16 values of real/fake state fidelity $F$ (circuit depth 15, 1 qubit circuit), probability $p$ of a real state (generated by $R$) being classified by $D$ as being real (circuit depth 11, 2 qubit circuit), probability $q$ of a fake state (generated by $G$) being classified as being real (circuit depth 11, 2 qubit circuit). These experiments were performed for parameter values found after each epoch of training.

For 32000 shots such circuit runs for 15 s per single cost function evaluation. For the random starting point used in Fig. 3 on average, we need 260 evaluations of the cost function to complete 15 training epochs (an epoch corresponds to 5 iterations of the Nelder-Mead algorithm). Our results show that the QCGN training on a quantum processor (see Fig. 3b) performs similarly as predicted by our numerical simulations (see Fig. 3a-b).

The experimental results, shown in Fig. 3, demonstrate that QCGN can be implemented using the available quantum computers, even without applying error correction. However, to obtain our result we applied standard measurement error mitigation method which corresponds to calibrating the detection part of the quantum computer.

To find the smallest number of shots needed for the learning process to complete, we have tested the proposed algorithm both on real quantum processors and simulators available to researchers via the IBMQ project. Each evaluation of the circuit was performed on 8192 shots, which was found to be sufficient to limit the effect of Poisson noise. Due to the technical imperfections of these real devices, the algorithm converged only in about one half of the runs. It should be stressed out, however, that the user can always rerun the algorithm until
it converges. One can observe that the algorithm converges to a non-zero value of the object function, which we also attribute to the experimental noise in the processor. Note that using the noiseless simulators, the algorithm converged on every attempt and the final object function was minimized below 0.001. This supports our finding that the algorithm is performing well, and the convergence difficulties are solely due to the noise in real presently available quantum processors.

**COMPARISON OF QGAN AND QCGN:**

**GENERATING AND RECOGNIZING A BELL STATE**

![Circuits Diagram](https://example.com/circuit.png)

**FIG. 4.** The circuits used for a) the generator $G$, b) the generator $R$ and c) one of the three consecutive layers of the discriminator $D$. Here, $U_i(\lambda)$ for $i \in \{1, 2, 3, 4\}$ is a parameterized single-qubit unitary operation, $R_\delta(\lambda)$ is the controlled-phase gate implementing a controlled rotation around $y$-axis, and $V^{(\lambda)}(\lambda)$ is a parameterized two-qubit unitary operation (operationally equivalent to $G$) for $i \in \{1, 2, 3\}$.

The proposed approach to generative quantum learning is conceptually different from the approaches described in Refs. [29][30]. Both approaches can solve an interesting problem, i.e., given samples of an entangled state, they can learn to generate the entangled states on their own. Moreover, the respective discriminators can be trained to detect the entangled state. However, from our numerical simulations it follows that for the same number of cost function calls, it is the QCGN that will complete the training first.

To illustrate this, let us consider generator $R$, which prepares a maximally entangled two-qubit singlet state $|\Psi^\prime\rangle$. Thus, there is one possible value of $\lambda = \epsilon$. The goal of the QGAN and QCGN training is to train generator $G$ (i.e., find the optimal circuit parameters) without knowing the algorithm used by $R$ nor its internal state $z_R$ by optimization of both the discriminator $D$ and the generator $G$. The circuits used for $D$, $G$, and $R$ are shown in Fig. 4. The circuits used for QGAN and QCGN are shown in Fig. 5 and Fig. 1b.

![Circuits Diagram](https://example.com/circuit.png)

**FIG. 5.** The circuits used for QGAN. a) a circuit that evaluates the performance of $D$ on the $|\Psi\rangle$ (computes $p$), b) a circuit that evaluates the performance of $D$ on the $|\psi\rangle$ (computes $q$), and c) a circuit that compares $R$ with $G$ (computes $F$).

To compare the dynamics of the training process we use (as in the previous section) three figures of merit: (i) probability $q$ of the fake ($G$-generated) state to be recognized as real by discriminator $D$, (ii) probability $p$ of the real state ($R$-generated) state to be recognized as real by discriminator $D$, (iii) the distance between $D = 1 - F$ the $G$-generated and $R$-generated states (linear entropy).

Because of the complexity of this example, here we use a gradient descent method (i.e., BSGF), which guarantees at most as many function evaluations as the Nelder-Mead method.

**CONCLUSIONS**

We have proposed a new efficient approach towards generative quantum machine learning. We have tested the proposed QCGN approach experimentally on a small-scale programmable quantum processor. The experimental results shown in Fig. 3 confirm the feasibility of implementing QCGN on a NISQ device. In addition to being conceptually different from a QGAN, our approach in all of the cases that we investigated numerically required fewer cost function evaluations than QGAN. Note that QCGN computes only $J$ and $p, q, F$ are computed addi-
FIG. 6. Comparison of the dynamics of the QCGN and QGAN training process as a function of the number of shots. The first column corresponds to the parameters for QCGN, while the second, corresponds to the parameters for QGAN. The consecutive rows illustrate the individual dynamics parameters: a) and b) probability $p$ of a real state being recognized as real by the discriminator, c) and d) probability $q$ of a fake state being recognized as real by the discriminator, e) and f) state fidelity $F$ associated with the trained generator.

FIG. 7. Comparison of the dynamics of the QCGN and QGAN training process. The first column corresponds to the parameters for QCGN, while the second, corresponds to the parameters for QGAN. The consecutive columns illustrate the individual dynamics parameters: a) and b) probability $p$ of a real state being recognized as real by the discriminator, c) and d) probability $q$ of a fake state being recognized as real by the discriminator, e) and f) state fidelity $F$ associated with the trained generator.

Another advantage of the proposed QCGN algorithm over QGAN is its higher learning depth. For QCGN, $n+2$ qubits are required to solve the problem, while for QGAN this value corresponds to as much as $3n+2$ qubits.

Also in Fig. 7 showing the dynamics of the learning process optimized after a set of function calls corresponding to epochs, we see a much smoother function for the QCGN case. Again, it takes significantly fewer epochs (as little as 4), relative to QGAN (about 250), for the discriminator to obtain a stable probability, at 0.5. Another advantage of QCGN approach is that it is slightly faster to settle on a high fidelity value, as seen in panels Fig. 7e) and f).

Note that in our numerical simulations, we have investigated how a quantum computer could learn the concept of a Bell state. After the training the network is able both to recognize and to generate this state. A next interesting step in would be to extend the notion of Bell state to...
an arbitrary entangled state to investigate how the concept of entanglement could be learned and understood by a quantum computer. Solving this problem would require combining the presented concepts and methods with more sophisticated classical machine learning.

Finally, it is interesting to note that the proposed generative learning approach falls into the category of kernel-based machine learning, where the initial part of the circuit can be viewed as the state preparation step, whereas the second part can be interpreted as kernel evaluation circuit.

ACKNOWLEDGMENTS

We would like to thank Dawid Maskalaniec and Mateusz Slysz for stimulating discussions and for their contributions to numerical calculations. Authors acknowledge financial support by the Czech Science Foundation under the project No. 19-19002S. The authors also acknowledge the projects Nos. LO1305 and CZ.02.1.01./0.0./0./16_019/0000754 of the Ministry of Education, Youth and Sports of the Czech Republic financing the infrastructure of their workplace. P.T. is supported by the Polish National Science Centre (NCN) under the Maestro Grant No. DEC-2019/34/A/ST2/00081. J.R. acknowledges the internal Palacky University grant IGA-PrF-2021-004. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

[1] M. Mohri, A. Rostamizadeh, and A. Talwalkar, Foundations of Machine Learning (MIT Press, Cambridge, MA, 2012).
[2] M. Schuld, I. Sinayskiy, and F. Petruccione, An introduction to quantum machine learning, Contemp. Phys. 56, 172 (2015).
[3] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature 549, 195 (2017).
[4] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, Quantum machine learning: a classical perspective, Proc. R. Soc. A 474, 20170551 (2018).
[5] V. Dunjko and H. J. Briegel, Machine learning & artificial intelligence in the quantum domain: a review of recent progress, Rep. Prog. Phys. 81, 074001 (2018).
[6] A. Pepper, N. Tischler, and G. J. Pryde, Experimental realization of a quantum autoencoder: The compression of qutrits via machine learning, Phys. Rev. Lett. 122, 060501 (2019).
[7] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, arXiv:1903.10563 (2019).
[8] X.-D. Cai, D. Wu, Z.-E. Su, M.-C. Chen, X.-L. Wang, L. Li, N.-L. Liu, C.-Y. Lu, and J.-W. Pan, Entanglement-based machine learning on a quantum computer, Phys. Rev. Lett. 114, 110504 (2015).
[9] R. Chatterjee and T. Yu, Generalized coherent states, reproducing kernels, and quantum support vector machines, Quantum Inf. Commun. 17, 1292 (2017).
[10] J. Gao, L.-F. Qiao, Z.-Q. Jiao, Y.-C. Ma, C.-Q. Hu, R.-J. Ren, A.-L. Yang, H. Tang, M.-H. Yung, and X.-M. Jin, Experimental machine learning of quantum states, Phys. Rev. Lett. 120, 240501 (2018).
[11] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big data classification, Phys. Rev. Lett. 113, 130503 (2014).
[12] M. Schuld and J. Killoran, Quantum machine learning in feature Hilbert spaces, Phys. Rev. Lett. 122, 040504 (2019).
[13] V. Trávníček, K. Bartkiewicz, A. Černoch, and K. Lemr, Experimental measurement of the Hilbert-Schmidt distance between two-qubit states as a means for reducing the complexity of machine learning, Phys. Rev. Lett. 123, 260501 (2019).
[14] P. L. McMahon, A. Marandi, Y. Haribara, R. Hamerly, C. Langrock, S. Yamate, T. Inagaki, H. Takesue, S. Utsumiy, K. Aihrara, R. L. Byer, M. M. Fejer, H. Malbuchi, and Y. Yamamoto, A fully programmable 100-spin coherent ising machine with all-to-all connections, Science 354, 614 (2016).
[15] D. Pierangeli, G. Marcucci, and C. Conti, Large-scale photonic ising machine by spatial light modulation, Phys. Rev. Lett. 122, 213902 (2019).
[16] Y. Shen, N. C. Harris, S. Skirlo, M. Prabhu, T. Baehr-Jones, M. Hochberg, X. Sun, S. Zhao, H. Larchelle, D. Englund, et al., Deep learning with coherent nanophotonic circuits, Nat. Photonics 11, 441 (2017).
[17] J. Bueno, S. Maktoobi, L. Froehly, I. Fischer, M. Jacquot, L. Larger, and D. Brunner, Reinforcement learning in a large-scale photonic recurrent neural network, Optica 5, 756 (2018).
[18] F. Tacchino, C. Macchiavello, D. Gerace, and D. Bajoni, An artificial neuron implemented on an actual quantum processor, npj Quantum Information 5, 26 (2019).
[19] S. C. Kak, Quantum neural computing, in Advances in Imaging and Electron Physics, Vol. 94 (Elsevier, 1995) p. 259.
[20] E. Farhi and H. Neven, Classification with quantum support vector machines, Phys. Rev. Lett. 114, 140504 (2015).
[21] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature 567, 209 (2019).
[22] A. Kandala, K. Temme, A. D. Córcoles, A. Mezzacapo, J. M. Chow, and J. M. Gambetta, Error mitigation extends the computational reach of a noisy quantum processor, Nature 567, 491 (2019).
[24] V. Dunjko, Y. Ge, and J. I. Cirac, Computational speedups using small quantum devices, [Phys. Rev. Lett. 121, 250501 (2018)].
[25] Y. Li and S. C. Benjamin, Efficient variational quantum simulator incorporating active error minimization, [Phys. Rev. X 7, 021050 (2017)].
[26] J. Preskill, Quantum Computing in the NISQ era and beyond, [Quantum 2, 79 (2018)].
[27] K. Fujii and K. Nakajima, Harnessing disordered-ensemble quantum dynamics for machine learning, [Phys. Rev. Applied 8, 024030 (2017)].
[28] S. Lloyd and C. Weedbrook, Quantum generative adversarial learning, [Phys. Rev. Lett. 121, 040502 (2018)].
[29] P.-L. Dallaire-Demers and N. Killoran, Quantum generative adversarial networks, [Phys. Rev. A 98, 012324 (2018)].
[30] C. Zoufal, A. Lucchi, and S. Woerner, Quantum generative adversarial networks for learning and loading random distributions, [npj Quantum Information 5, 1 (2019)].
[31] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, [Phys. Rev. A 98, 032309 (2018)].
[32] M. Schulz, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Evaluating analytic gradients on quantum hardware, [Phys. Rev. A 99, 032331 (2019)].
[33] J. Jašek, K. Jiráková, K. Bartkiewicz, A. Černoch, T. Fürst, and K. Lemr, Experimental hybrid quantum-classical reinforcement learning by boson sampling: how to train a quantum cloner, [Opt. Express 27, 32454 (2019)].
[34] K. Lemr, K. Bartkiewicz, A. Černoch, M. Dušek, and J. Soubusta, Experimental implementation of optimal linear-optical controlled-unitary gates, [Phys. Rev. Lett. 114, 153602 (2015)].
[35] IBM Quantum, [https://quantum-computing.ibm.com/], 2021.