Classification and reconstruction of optical quantum states with deep neural networks

Shahnawaz Ahmed,[1,*] Carlos Sánchez Muñoz,[2] Franco Nori,[3,4,5] and Anton Frisk Kockum[1,†]

[1] *Department of Microtechnology and Nanoscience, Chalmers University of Technology, 412 96 Gothenburg, Sweden*
[2] *Departamento de Fisica Teorica de la Materia Condensada and Condensed Matter Physics Center (IFIMAC), Universidad Autonoma de Madrid, Madrid 28049, Spain*
[3] *Theoretical Quantum Physics Laboratory, RIKEN Cluster for Pioneering Research, Wako-shi, Saitama 351-0198, Japan*
[4] *RIKEN Center for Quantum Computing (RQC), Wako-shi, Saitama 351-0198, Japan*
[5] *Department of Physics, University of Michigan, Ann Arbor, Michigan 48109-1040, USA*

(Received 14 December 2020; accepted 6 July 2021; published 27 September 2021)

We apply deep-neural-network-based techniques to quantum state classification and reconstruction. Our methods demonstrate high classification accuracies and reconstruction fidelities, even in the presence of noise and with little data. Using optical quantum states as examples, we first demonstrate how convolutional neural networks (CNNs) can successfully classify several types of states distorted by, e.g., additive Gaussian noise or photon loss. We further show that a CNN trained on noisy inputs can learn to identify the most important regions in the data, which potentially can reduce the cost of tomography by guiding adaptive data collection. Secondly, we demonstrate reconstruction of quantum-state density matrices using neural networks that incorporate quantum-physics knowledge. The knowledge is implemented as custom neural-network layers that convert outputs from standard feed-forward neural networks to valid descriptions of quantum states. Any standard feed-forward neural-network architecture can be adapted for quantum state tomography (QST) with our method. We present further demonstrations of our proposed QST technique with conditional generative adversarial networks (QST-CGAN) [Ahmed *et al.*, Phys. Rev. Lett. **127**, 140502 (2021)]. We motivate our choice of a learnable loss function within an adversarial framework by demonstrating that the QST-CGAN outperforms, across a range of scenarios, generative networks trained with standard loss functions. For pure states with additive or convolutional Gaussian noise, the QST-CGAN is able to adapt to the noise and reconstruct the underlying state. The QST-CGAN reconstructs states using up to *two orders of magnitude fewer iterative steps* than iterative and accelerated projected-gradient-based maximum-likelihood estimation (MLE) methods. We also demonstrate that the QST-CGAN can reconstruct both pure and mixed states from *two orders of magnitude fewer randomly chosen data points* than these MLE methods. Our paper opens possibilities to use state-of-the-art deep-learning methods for quantum state classification and reconstruction under various types of noise.

DOI: 10.1103/PhysRevResearch.3.033278

I. INTRODUCTION

Neural networks (NNs) are becoming ubiquitous in various areas of physics as a successful machine-learning (ML) technique to solve different tasks [1]. Applications range from particle physics [2], cosmology [3–5], and many-body quantum matter [6] to material sciences [7], and even to discover new physics [8,9]. The NNs are used in classification problems, where the goal is to assign a label to a data sample [10], and for generative tasks, where new data is created after learning the underlying data distribution from samples [11,12]. Deep neural networks (DNNs) have shown impressive results in image classification [13,14], object detection [15], image denoising and inpainting [16–18], deconvolution [19], generating realistic-looking images [20–23], text generation and translation [24,25], as well as for generating audio [26], video [27], simulating gaming graphics [28], and writing computer programs automatically [29]. There are also recent examples of NN-based machine learning successfully applied to grand challenges in life sciences, e.g., the AlphaFold algorithm for protein folding [30,31].

In quantum information and computing [32–37], some of the problems faced in characterizing and controlling quantum systems can be translated to tasks in ML. Many of these problems are data-driven and NN-inspired techniques have been used to successfully address them, e.g., identifying phase transitions [38], detecting nonclassicality or entanglement of quantum states [39–43], design of quantum experiments [44–47], quantum error correction [48–52], characterizing and calibrating quantum devices [53,54], and foundational questions [55]. Moreover, automatic differentiation, a technique used to train neural networks, has been used for wave function positivization [56], speeding up quantum optimal control [57],

[*] shahnawaz.ahmed95@gmail.com
[†] anton.frisk.kockum@chalmers.se

and other tasks in real-life experiments such as better state transfer in the presence of dissipation [58].

For quantum state characterization, even distinguishing two different quantum states can become challenging. For example, telling a coherent source of light and a thermal state apart can be difficult due to the close similarity of their data in low-photon regimes [41]. Similarly, distinguishing different mode superpositions of twisted light [59] in a landmark experiment necessitated the use of neural networks.

Beyond just identifying properties of the quantum states or classifying them, reconstructing a full quantum state description presents an even more challenging task, called quantum state tomography (QST) [60–63]. The challenges arise mainly due to the exponentially large Hilbert-space dimension required to fully describe the state [33,64–66]. For example, k two-level quantum systems (qubits) have a Hilbert space of dimension $N = 2^k$ and require up to $N^2 - 1$ real numbers to fully determine a density matrix describing the state. Therefore, QST requires clever data processing to extract a good representation of a state from noisy data [67–72]. The presence of noise further complicates the problem; for additive Gaussian noise, one reconstruction method [73] has computational complexity $O(N^4)$.

The success of NNs in other fields has prompted their application to several quantum state classification and reconstruction tasks. The motivation is that NNs are universal function approximators [74–76] that can learn maps from noisy input data to class labels, or act as variational ansätze for quantum states [77–79]. The variational ansätze can be learned from data by minimizing some loss metric between the predictions from the NN-based model and the data. From a computational learning perspective, approximately learning a quantum state has a linear scaling in the number of quantum bits [80].

In this paper, outlined in Fig. 1, we connect the tasks of quantum state classification and reconstruction in a general way to discriminative and generative problems in ML. We demonstrate the feasibility of using DNNs for classification and reconstruction, showing how to flexibly adapt them for different scenarios, e.g., noise or scarce data. Crucial components of our methods include incorporating knowledge of quantum physics and other prior information into the network.

Many previous applications of DNNs for classifying quantum data [40–42,81,82] consider properties like nonclassicality or entanglement. In these works, more complicated noise models, beyond simple detection inefficiencies, are not considered. Since the classification task we tackle seems rather straightforward for DNNs, we attempt to go beyond the standard paradigm (training on simulated data, testing on new data) and demonstrate results with different types of noise for general states and measurements [see Figs. 1(a)–1(d)]. We also propose an adaptive data-collection method using a trained DNN to extract interesting patterns in the data and leverage it for adaptive tomography [see Fig. 1(f)].

In quantum state reconstruction [see Fig. 1(g)], one of the most popular neural-network approaches is to use restricted Boltzmann machines (RBMs) to map the underlying Boltzmann probability distribution of an RBM to the distribution of measurement outcomes on a quantum state [77,78,83–85] [see Fig. 1(h)]. This technique has some shortcomings, e.g., difficulties with sampling and lack of straightforward training for larger models. Recently, there has therefore been proposals to instead use feed-forward architectures, including recurrent neural networks (RNNs) and transformers, for QST [86–88]. Unlike RBMs, such neural-network architectures are straightforward to train, without any need for sampling steps, using gradient-based optimization with backpropagation. However, state-of-the-art results for generative tasks in ML often use variational autoencoders (VAEs) [11,89] and generative adversarial networks (GANs) [12,90], which only recently are beginning to be explored for learning quantum states [91–94] [see Figs. 1(h)–1(i)].

Results on the reconstruction of multiqubit states suggest several benefits of NN-based reconstruction over standard techniques [84,95–97]. In Ref. [87], states with up to 90 qubits are reconstructed in simulation. The ideas follow from Ref. [86], where quantum state reconstruction using generative models, both RBMs and RNNs is combined with a tensor-network paradigm. Similarly, in Ref. [98], fully connected DNNs were used for denoising data and dealing with state-preparation and measurement (SPAM) errors. In Ref. [97], a CNN was trained on simulated data (with noise) and proved able to reconstruct two-qubit states directly from data, outperforming a standard Stokes reconstruction.

However, such demonstrations are usually on simple states, with limited error models, and/or do not fully ensure that the reconstructed states are physical. For example, Ref. [87] considers Greenberger-Horne-Zeilinger (GHZ) states, which only contain four nonzero elements in the density matrix. In Refs. [98] and [97], the Hilbert-space dimensions are restricted to six and four, respectively. Even then, techniques to include prior knowledge, such as the properties of quantum states or background noise, need to be explored. In Ref. [97], noise is handled by adding it to the simulated training dataset and properties of a quantum state are enforced using a similar idea to our proposal in Ref. [99] independently. In Ref. [87], where GHZ states are reconstructed using a Transformer neural network, some reconstructed states have fidelities exceeding unity, which indicates the lack of quantum-mechanical constraints on the state description. In an experimental two-qubit reconstruction with the RBM ansatz, an improvement was observed when the variational ansatz was restricted to physical states, but this added costs during learning [100].

Furthermore, many of the approaches discussed so far cater specifically to qubit-based tomography. For continuous-variable (CV) quantum systems, which currently are attracting much attention for implementation of quantum computing [101–108], special adaptations are required, as in, e.g., Ref. [84], where RBMs were adapted for CV systems, but required an exhaustive search of all possible configurations to train. Lastly, the reconstruction techniques usually either use DNNs to reconstruct a single state where data from one experiment is enough, or require training datasets [97,98]. We show in Ref. [99] adaptions that allow the same DNN to both reconstruct states from scratch or perform single-shot reconstructions by mapping data to the space of density matrices in a general way.

Our motivation here is thus to address some of the problems discussed above and realize a unified framework that can

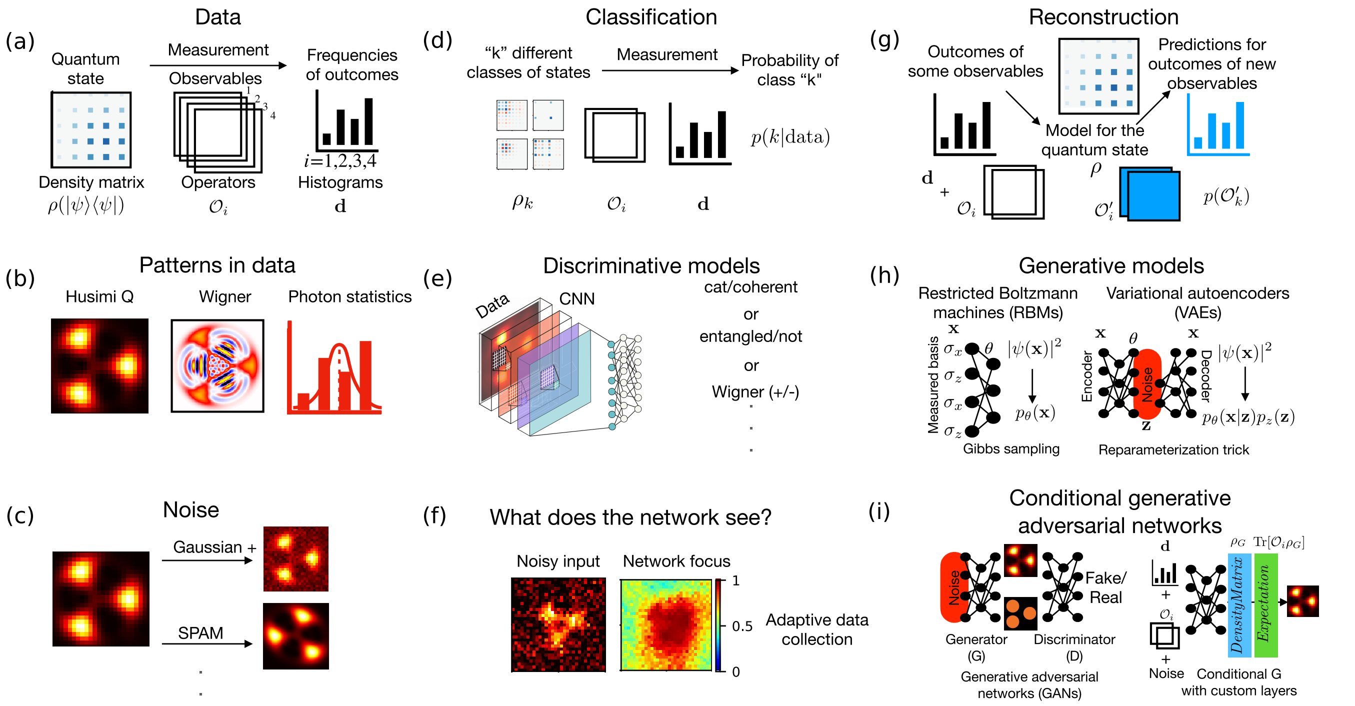

FIG. 1. An outline of the topics discussed in this work. (a) The data required for quantum state tomography consists of the frequencies of measurement outcomes from observables represented as Hermitian operators. The aim is to reconstruct a description of the quantum state—usually a density matrix or the wave function. (b) Measurements of quasi-probability distributions (Wigner or Husimi Q) reveal interesting visual features in the data. Similarly, the histograms of measurement statistics, e.g., the photon-number distribution, can have patterns. Such features and patterns can be used for classification or reconstruction. (c) Several types of noise can corrupt the state or the data. Some types of noise, e.g., white noise, can be reduced by more data collection. Other types of noise, e.g., state-preparation-and-measurement (SPAM) noise, are more difficult to handle. (d) Classification tasks attempt to assign a label to data, classifying it according to its properties, e.g., if it is a Schrödinger-cat state, has Wigner negativity, or is an entangled quantum state. (e) Neural networks can be trained for classification of states or their properties. (f) Once it has been trained, analyzing how a neural network determines the class of a state can help to focus on the most important features in the data. This can be leveraged for adaptive data collection. (g) Reconstruction of quantum states connects to generative modeling tasks, where the goal is to learn the underlying probability distribution of the data to sample new data from it. In quantum state reconstruction, we aim to learn an underlying model, usually in the form of a wave function or density matrix that can generate statistics for any measurement operator. (h) Neural-network methods can be used for estimating or approximating underlying probability densities explicitly using restricted Boltzmann machines (RBMs) or variational autoencoders (VAEs). Training RBMs is not straightforward due to sampling requirements. This is resolved in VAEs using a reparameterization trick that allows gradient-based backpropagation for training. (i) Generative adversarial networks (GANs) provide a density-estimation technique, where we do not explicitly define the density nor require the reparametrization trick for training. We combine ideas from VAEs and GANs to propose a new quantum state tomography technique with conditional GANs—the QST-CGAN. Our QST-CGAN method allows for explicit estimation of the density matrix and computation of measurement statistics using two custom layers—*DensityMatrix* and *Expectation*.

flexibly work with different types of quantum data in various settings. Our contribution is a general method that allows the use of neural networks to capture patterns in data and explicitly generate a density-matrix description as an intermediate representation inside the network. This idea is inspired by developments in density estimation with neural networks [109], more specifically, the VAE architecture [11], which learns an underlying complex data distribution using a simple latent noise space to generate new data. We augment the latent space to be a full quantum state description (the density matrix) conditioned on inputs that are both the data samples and the operators that define the measurements. Using this conditioning allows us to have a very general technique that can further handle known noise—we simply add the noise as an input variable. We consider the role of loss functions for reconstruction and motivate our idea of using a learnable loss in the form of a neural network based on the idea of GANs [12,20,21]. Our QST-CGAN technique thus combines concepts from VAEs and GANs, as illustrated in Fig. 1(i). In this paper, we present details of the implementation and results for noisy reconstruction, reconstruction of mixed states, and reconstruction from reduced data.

This paper is organized as follows. In Sec. II, we briefly discuss the quantum state tomography and state discrimination problems in the context of generative and discriminative modeling. In order to demonstrate our methods, we consider optical quantum states as examples. The various types of data from optical quantum states that will be used throughout the paper are presented in Sec. III, including possible sources of noise. In Sec. IV, we describe details of the neural-network architectures and training methods. In particular, we discuss the custom layers that we introduce for reconstruction here

TABLE I. List of abbreviations (in alphabetical order) used in this paper.

Full name	Abbreviation
Accelerated projected gradient	APG
Compressed sensing	CS
Conditional generative adversarial network	CGAN
Continuous variable	CV
Convolutional neural networks	CNNs
Deep neural networks	DNNs
False positive rate	FPR
Generative adversarial networks	GANs
Gradient-weighted class activation mapping	Grad-CAM
Greenberger-Horne-Zeilinger	GHZ
Informationally complete	IC
Integral probability metrics	IPMs
Iterative maximum-likelihood estimation	iMLE
Kullback-Leibler	KL
Machine learning	ML
Matrix product state	MPS
Maximum-likelihood estimation	MLE
Neural networks	NNs
Positive-operator-valued measures	POVMs
Projected gradient descent	PGD
Quantum state discrimination	QSD
Quantum state tomography	QST
Quantum state tomography with conditional generative adversarial network	QST-CGAN
Receiver-operating-characteristic	ROC
Recurrent neural networks	RNNs
Restricted Boltzmann machines	RBMs
State preparation and measurement	SPAM
Tensor network	TN
True positive rate	TPR
Variational autoencoders	VAEs

and in Ref. [99]. Then, we present the results for the classification task in Sec. V A, where we also analyze the impact of noise on the classification performance. In Sec. V B, we show the performance of the QST-CGAN on noisy data and the role played by various loss functions in the reconstruction. Finally, we conclude in Sec. VI and discuss, in Sec. VII, further possibilities and potential for development of the techniques presented here. In Table I, we list all the abbreviations used throughout the paper for easy reference.

II. BACKGROUND

In this section, we set the stage for the paper by providing an overview of the problems of quantum state discrimination (QSD) and quantum state tomography (QST). We then discuss generative and discriminative modeling in machine learning, which is related to these problems. We compare different neural-network approaches to such modeling to motivate our choice of methods in this paper for tackling QST and QSD.

A. Quantum state discrimination

The task in QSD is to classify an unknown state ρ as being one of a given finite ensemble of states $\{\rho_i\}$, from which states are chosen with probabilities $\{p_i\}$ such that $\sum_i p_i = 1$ [110]. The classification is done by performing measurements on ρ, typically positive-operator-valued measures (POVMs) $\{\mathcal{O}_i\}$, designed such that observing the outcome i, which occurs with probability $p_i = \mathrm{tr}(\mathcal{O}_i\rho)$, corresponds to the state being ρ_i.

The problem of QSD can thus often be rephrased as finding the optimal measurement for discriminating between the $\{\rho_i\}$. In case the states to be discriminated between are not orthogonal, perfect single-shot QSD is not possible. The optimal measurement should then instead maximize the probability of guessing the state correctly [111]. Note that the nonorthogonality of quantum states, which prevents perfect QSD, does not have a classical analog; it cannot be explained by merely assuming overlapping probability distributions [110,112].

If repeated state preparation and measurement is possible, adaptive measurement schemes, where new measurements are chosen based on the results of previous measurements, may be optimal. In this paper, we will consider such a situation, where we can make repeated measurements and collect statistics for various POVMs. However, our aim in this paper is not to construct highly optimized complex POVMs or adaptive schemes, but to show that a neural network can learn to perform QSD well when working with limited measurement data from standard, simple measurements of complex optical quantum states. Insights gleaned from the neural-network performance could then be used to minimize the number of simple measurements needed in experiments to classify states with high certainty. Furthermore, rapid state classification could help find a good starting point and parametrization for full quantum state reconstruction. Previous work has shown that neural networks can distinguish thermal and coherent light sources with few measurements [41]; here, we present a general framework for applying such techniques to arbitrary measurements and states. Note that we do not only distinguish between two types of states, but between many types of states at the same time.

B. Quantum state tomography

The goal of QST is more ambitious than that of QSD: to fully characterize an unknown quantum state, usually by obtaining its density matrix ρ. A physical density matrix is Hermitian, positive semidefinite, and has unit trace. In an N-dimensional Hilbert space, $N^2 - 1$ real numbers have to be estimated from POVM outcomes to completely determine a general ρ. This can be seen clearly from the Cholesky decomposition

$$\rho = T^\dagger T, \tag{1}$$

which is extensively used in reconstruction methods to ensure positivity and Hermiticity. The matrix T is lower-triangular with complex-valued entries except on the diagonal, where the entries are real-valued.

The measurement data used for reconstruction of ρ consists of single-shot outcomes from POVMs $\{\mathcal{O}_i\}$. By repeating the measurement on identically prepared quantum states, we can gather statistics. The frequencies d_i of various measurement outcomes is proportional to the expectation value $\mathrm{tr}(\mathcal{O}_i\rho)$ and forms our data $\mathbf{d}$. The reconstruction problem can therefore be stated as an inversion problem [113]

$$\mathbf{d} = A\rho_f, \tag{2}$$

where the sensing matrix A is given by the choice of measurement operators and ρ_f is the flattened density matrix.

The invertibility of Eq. (2) depends on the set of measurement operators. A set of measurement operators that enables inversion, and thus allows the complete characterization of the state, is called informationally complete (IC) [114]. For a state in an N-dimensional Hilbert space, up to $\sim N^2$ POVMs may be needed for IC (and each measurement needs to be repeated multiple times to gather the statistics). However, with some *a priori* knowledge of the state, e.g., that ρ is low rank or that certain elements of ρ are zero, the measurements can be cleverly selected and their number reduced.

Reconstructing ρ from $\mathbf{d}$ is thus an estimation problem, which can be approached in many ways. Common reconstruction techniques include linear inversion [115,116], maximum-likelihood estimation (MLE) [62,117–119], and Bayesian methods [120–122]. Linear inversion, while being straightforward, can fail due to noise in the data or a high condition number of A [113] and produce unphysical entries in the density matrix, e.g., negative diagonal elements [123]. Therefore statistical inference techniques such as MLE or Bayesian estimation are preferred. Such methods give an estimate ρ' for the density matrix by optimizing the likelihood function

$$L(\rho'|\mathbf{d}) = \prod_i [\mathrm{tr}(\rho'\mathcal{O}_i)]^{d_i}. \tag{3}$$

In case of continuous-variable outputs, where d_i is a real number, appropriate binning is necessary to apply MLE [124]. Alternatively, the mean squared error between the output and the expected value can be minimized [73].

Although MLE guarantees a physical ρ', it does not provide any error bars to quantify the uncertainty in the estimate. Recently, it has also been argued that MLE is not optimal and is an *inadmissible* estimator for common metrics such as fidelity, mean-squared error, and relative entropy [125]. Bayesian methods for QST, on the other hand, can quantify the uncertainty in the parameters of the density matrix using a prior probability distribution over different states $\pi(\rho)$ [120,121]. The initial prior $\pi_0(\rho)$ should be uniform, or as uninformative as possible, and is updated by applying the Bayes theorem using the likelihood $L(\rho'|\mathbf{d})$ to give a posterior $\pi_f(\rho) \propto L(\rho|\mathbf{d})\pi_0(\rho)$. The best estimate of the underlying state is given as the mean over all states ρ_μ, defined by the posterior distribution π_f weighted by the likelihood computed from observed data:

$$\rho_\mu = \int \rho\pi_f(\rho)d\rho. \tag{4}$$

Other examples of methods to optimize the likelihood function and obtain a density matrix estimate include diluted MLE [123], compressed sensing (CS) [126] and projected gradient descent [119,127,128]. The CS methods are motivated by simple parameter-counting arguments: we should only require $O(rN)$ measurements, with r being the (low) rank of the density matrix [129]. Examples of such low-rank states, common in experiments, are pure quantum states corrupted by local noise processes. Recently, other modifications of CS have been proposed and demonstrated experimentally for adaptive tomography [130,131], which only require the a priori information of the density-matrix dimension (an improvement over CS, which requires an a priori guess of r).

However, a good ansatz or model for the state can reduce the effort for reconstruction. If we consider classes of quantum states having particular properties or symmetries, we can write their descriptions with fewer parameters than the $N^2 - 1$ required for a general density matrix. Matrix-product-state (MPS) [69,132] and tensor-network (TN) tomography [133,134] are methods that find efficient ansätze for states using MPSs or TNs, and permutationally invariant tomography [135,136] exploits permutational symmetries of the density matrix. Some other improvements assume a noise model, e.g., additive gaussian noise [73], and therefore these techniques are often restricted to specific situations, lacking versatility.

A different formulation from the above techniques comes from the idea of projected gradient descent (PGD) [119,127,128]. In such methods, a cost function is constructed that distinguishes between model-predicted data and the true data to apply gradient-based optimization to find the best estimate for the model (the density matrix). The benefit of the PGD technique is that it quickly converges to the MLE state in a wider variety of scenarios, even when the problem is ill-conditioned. The PGD method also sets up this notion of a cost function, thereby translating the QST problem into an optimization problem. Here, we made use of a fast MATLAB implementation of the accelerated projected-gradient method for MLE (APG-MLE) from Ref. [119]. We adapted this code to our examples with optical quantum states and used it to benchmark some of our results.

Neural-network-based reconstruction methods have also shown significant promise. In such approaches, neural networks are either used as an ansatz for the state to obtain probabilities of measurement outcomes [84,95,96], or to directly estimate ρ [97]. However, a general framework to study quantum state reconstruction using standard feed-forward neural networks is missing. In this paper, we present a framework that allows any standard neural network to be used for quantum state discrimination and reconstruction by adapting the generative and discriminative modeling framework from machine learning to QSD and QST.

C. Discriminative and generative modeling

Quantum state discrimination and reconstruction can be related to discriminative and generative tasks in machine learning. Consider a data space $\mathbf{S}$ from which we obtain samples $\mathbf{x}$ of a random variable $\mathbf{X}$. The samples can be classified as having one of k different labels y. A dataset can thus consist of a collection of pairs $\{\mathbf{x}, y\}$.

A discriminative model attempts to predict the class label y for a data point $\mathbf{x}'$, i.e., finding the correct conditional probability $p(y|\mathbf{x}')$. We loosely interpret this as identifying whether a data point belongs to one of k possible data distributions $p^{[k]}_{\mathtt{data}}$.

A generative model aims to generate new samples $\mathbf{x}'$ that are similar to the observed data, which is assumed to be drawn from a data distribution defined by a probability density $p_{\mathtt{data}}(\mathbf{x})$. In general, real-world data distributions can be very complex, making it a hard problem to model them in a way that is both easy to compute and expressive enough to capture subtleties of the data. In Sec. II D, we discuss how deep neural networks are used to tackle such challenging distributions for generative and discriminative tasks.

The ideas of discriminative and generative modeling can be connected to QSD and QST in the following way. First, we identify the data space $\mathbf{S}$ with the space of measurement outcomes for operators $\{\mathcal{O}_i\}$. The outcomes can be collected either as single shots or average values; we denote the collected outcomes by $\mathbf{d}$. The expectation value $\langle \mathcal{O}_i \rangle = \mathrm{tr}(\mathcal{O}_i \rho)$ replaces the classical expectation value

$$\mathrm{E}[\mathbf{X}] = \int \mathbf{x} p_{\mathtt{data}}(\mathbf{x})\, d\mathbf{x}. \tag{5}$$

Thus ρ takes the role of a probability density function for the quantum system. If the data comes from one of k different quantum states $\rho^{[k]}$, we can assign it a label y. Our data set is then formed by pairs $\{\mathbf{d}, y\}$.

The discrimination task of assigning one of the k labels to some observed data $\mathbf{d}'$ is QSD. Reconstruction of ρ can be considered a generative modeling task, where we aim to generate outcomes $\mathbf{d}'$ of new measurement operators $\{\mathcal{O}'_i\}$ after having observed some results of POVM measurements. To fulfill that task, we either need to obtain ρ directly or find some parametrization of ρ that lets us calculate $\langle \mathcal{O}'_i \rangle = \mathrm{tr}(\mathcal{O}'_i \rho)$.

Just like complicated classical data distributions $p_{\mathtt{data}}$, ρ can depend on many parameters and be difficult to estimate. However, efficient parametrizations of the quantum state using matrix-product states [69,132], tensor networks [133,134], or neural networks [77,78,83,84,86–88,97,98,100] have reduced data and computation costs for quantum-state reconstruction. In this paper, we provide a general method to obtain ρ as the output of neural networks, allowing the conversion of any neural-network architecture into a generative model for QST. Our ideas are applicable to any parametrization of ρ.

D. Neural networks as discriminative and generative models

Neural networks can approximate any function arbitrarily well [74]. They can be treated as functions that map an input space to a target space:

$$f(\theta) : \mathbf{S} \to \mathbf{T}, \tag{6}$$

where θ are parameters that are learned from training on (labelled) data samples $\{\mathbf{x}, y\}$.

To use neural networks for discriminative tasks (classification) is fairly straightforward. In this case, the output $f(\mathbf{x};\theta)$ of the network is interpreted as the conditional probability $p(y|\mathbf{x})$ [10]. Then, by constructing a loss function that quantifies the total error of predictions on a training set, we can optimize the parameters θ to minimize the classification error.

Using neural networks as generative models is not as simple as mapping input data to target labels. Since a standard feed-forward neural network is a deterministic function $f(\mathbf{x};\theta)$, it cannot be sampled to generate new data $\mathbf{x}'$. Early schemes used to circumvent this problem were neural networks with stochastic outputs, e.g., restricted Boltzmann machines (RBMs). Later, deterministic feed-forward neural networks were adapted to give stochastic outputs for generative tasks; examples include variational autoencoders (VAEs) and generative adversarial networks (GANs). Below, we briefly discuss these methods to motivate our choice of using the conditional variant of GANs for quantum state reconstruction, and to show how our architecture also has connections to the other models.

1. Restricted Boltzmann machines

Restricted Boltzmann machines [137–139] are stochastic neural networks that can represent arbitrary data distributions. An RBM consists of visible ($\mathbf{v}$) and hidden ($\mathbf{h}$) units, which give stochastic binary outputs $\mathbf{v}, \mathbf{h} \in \{0, 1\}$. In single evaluations of the RBM, the states of the hidden units h_j are updated to 1 if the probability

$$p(h_j = 1|\mathbf{v}) = g\left(b_j + \sum_i w_{i,j} v_i\right) \tag{7}$$

is greater than a random number uniformly distributed between 0 and 1 (sampled in each update step). Here g is the sigmoid activation function and $\{b_j, w_{i,j}\}$ are parameters determining the interaction between different units. A visible unit v_i is similarly updated depending on the states of the hidden units and another parameter a_i.

The result of updating the RBM units iteratively in this way from a random initial state is that the states of the visible units converge to a Boltzmann distribution

$$p(\mathbf{v};\theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h};\theta)}, \tag{8}$$

where $Z(\theta) = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h};\theta)}$ is the partition function and the energy is given by

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i\in\text{visible}} a_i v_i - \sum_{j\in\text{hidden}} b_j h_j - \sum_{i,j} v_i h_j w_{ij}, \tag{9}$$

parametrized by $\theta = a, b, w$. To train an RBM is to find parameters θ, which make the probability distribution $p(\mathbf{v};\theta)$ mimic the data distribution $p_{\mathtt{data}}$, as measured by some statistical divergence, e.g., the Kullback-Leibler (KL) divergence. After training, new data points can then be generated by sampling $p(\mathbf{v};\theta)$. Since standard RBMs only output binary-valued data, continuous-valued data needs to be handled either in a binary encoding or by using variants like Gaussian-Bernoulli RBMs [140].

Although RBMs have been around for a long time, it was only recently that effective techniques for training them, e.g., contrastive divergence [139,141], were found and enabled them to play a significant role in the initial success of image processing with deep neural networks. These training methods have later been successfully applied to QST with RBMs [142]. However, RBMs are still not straightforward to train and are less flexible than feed-forward or convolutional neural networks. In particular, the partition function $Z(\theta)$ can be difficult to compute since it involves a sum over an exponential number of states [143]. Furthermore, the sampling methods can have convergence issues for typical high-dimensional problems. These issues have stimulated the development of standard feed-forward neural networks converted for generative modeling.

2. Variational autoencoders

Variational autoencoders are an early example of an adaptation of standard feed-forward neural networks to generative modeling. The idea of VAEs is to generate new data by sampling from a latent space $\mathbf{Z}$ and mapping it to the data space $\mathbf{S}$,

$$\mathbf{Z} \xrightarrow{\texttt{Generator}} \mathbf{S}. \tag{10}$$

The latent space is used to define the data distribution, parameterized by θ, as the marginal of a joint distribution $p_\theta(\mathbf{x}, \mathbf{z})$ over the data and latent variables [11,89,144]:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}. \tag{11}$$

The latent variable model $p_\theta(\mathbf{x}, \mathbf{z})$ can be specified by using some prior noise distribution $p_z(\mathbf{z})$, assuming the following factorization representing an infinite mixture model:

$$p_\theta(\mathbf{x}) = \int p_z(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z}. \tag{12}$$

In VAEs, a neural network g acts as a stochastic decoder to map the latent space to data:

$$p_\theta(\mathbf{x}|\mathbf{z}) = p[\mathbf{S}|g_\theta(\mathbf{z})]. \tag{13}$$

Even if the factors in Eq. (12) are simple, e.g., Gaussians, their mixture can be very expressive and thus capture complex data distributions.

However, the marginal $p_\theta(\mathbf{x})$ is typically intractable due to the integral in Eq. (11). Finding θ by some gradient-based optimization is thus not feasible. The intractable nature of the marginal stems from the intractability of the posterior

$$p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x}, \mathbf{z})}{p_\theta(\mathbf{x})}. \tag{14}$$

In VAEs, this posterior is approximated using a stochastic encoding in an encoder neural network e, parameterized by ϕ, that maps the data space to the latent space:

$$p_\theta(\mathbf{z}|\mathbf{x}) \approx q_\phi(\mathbf{z}|\mathbf{x}) = p[\mathbf{Z}|e_\phi(\mathbf{x})]. \tag{15}$$

The VAE architecture thus closely resembles that of an autoencoder—a neural network that finds a compressed representation of data by encoding it in a latent space and reconstructing it back from there,

$$\mathbf{S} \xrightarrow{\text{encoding}} \mathbf{Z} \xrightarrow{\text{decoding}} \mathbf{X}. \tag{16}$$

In general, VAEs assume $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ to be Gaussians specified by g_θ and e_ϕ. The encoder network processes an input $\mathbf{x}$ to give the mean and covariance for a multidimensional Gaussian, which is sampled to obtain latent vectors $\mathbf{z}$. The decoder then generates new data $\mathbf{x}'$ from another multidimensional Gaussian with the mean and covariance determined by the sampled noise vector.

To obtain the parameters (θ, ϕ), we want to maximize $\ln(p_\theta(\mathbf{x}))$, but since it is intractable, the variational approach maximizes the evidence lower bound or minimizes the loss

$$E_{\mathbf{x}\sim p_{\texttt{data}}}\big[-E_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \ln(p_\theta(\mathbf{x}|\mathbf{z})) + \mathrm{KL}(\mathrm{q}_\theta(\mathbf{z}|\mathbf{x}), \mathrm{p_z}(\mathbf{z}))\big]. \tag{17}$$

However, training such a variational model comes with its own challenges due to the stochastic nature of the encoder and decoder. Even using a reparametrization trick making backpropagation-based training work on VAEs, several critical issues leave VAEs susceptible to generate samples that do not match the data distribution well. In image-generation tasks, this leads to blurry images as the Gaussian mixtures, used for their simplicity, are not the best for representing natural data distributions.

3. Generative adversarial networks

Generative adversarial networks [12] and their conditional variant, conditional GANs (CGANs) [21], solve the problem of approximating data distributions in a different way than RBMs and VAEs. In the GAN framework, a standard feed-forward neural network G, with parameters θ_G, generates new data using noise vectors $\mathbf{z}$:

$$\mathbf{x}' = G(\mathbf{z}; \theta_G). \tag{18}$$

The network G is trained by letting a second neural network, the discriminator D, evaluate the outputs from G. Unlike in a VAE, the second neural network does not map the input space to the latent space. Instead, the discriminator directly trains the generator to find the map from the latent noise space to data.

The discriminator D is a standard classifier network, parametrized by θ_D, that takes an input $\mathbf{x}'$ and outputs a probability $D(\mathbf{x}'; \theta_D)$ that $\mathbf{x}'$ comes from the data distribution. The parameters $\{\theta_G, \theta_D\}$ are optimized in an alternating fashion until the generator produces outputs that the discriminator cannot distinguish from samples of the real dataset, i.e., both $\mathbf{x}' \sim p_{\texttt{data}}$ and $\mathbf{x} \sim p_{\texttt{data}}$. In each optimization step, θ_D is first updated to maximize

$$E_{\mathbf{x}\sim p_{\texttt{data}}}[\ln{(D(\mathbf{x}; \theta_D))}] + E_{\mathbf{z}\sim p_z}[\ln(1 - D(G(\mathbf{z}; \theta_G); \theta_D))]. \tag{19}$$

Then, θ_G is updated to minimize

$$E_{\mathbf{z}\sim p_z}[\ln(1 - D(G(\mathbf{z}; \theta_G); \theta_D))]. \tag{20}$$

When the training is completed, new samples can be generated from G using noise vectors $\mathbf{z}$.

A standard GAN can only generate samples randomly according to the data distribution. However, we can modify the inputs to G and D by adding a conditioning variable $\mathbf{c}$ to guide the output. This leads to the CGAN architecture, where

$$G : \mathbf{z}|\mathbf{c} \rightarrow \mathbf{x}, \tag{21}$$

$$D : \mathbf{x}|\mathbf{c} \rightarrow \Pr(\mathbf{x} \sim p_{\texttt{data}}). \tag{22}$$

The optimization of parameters for the CGAN networks follows the same procedure as for the GAN, i.e., maximizing Eq. (19) and minimizing Eq. (20).

The CGAN architecture provides a very flexible method for modeling complex conditional maps between different spaces. The flexibility stems from using the discriminator network, instead of a fixed loss function, as an evaluator of the generator performance. Conditional GANs have been used successfully in many types of generative tasks, e.g., generating images [145,146], converting edges to images, converting day to night pictures [21], etc. In this paper, we use CGANs for QST, but we also borrow ideas from VAEs for this task.

III. DATA

In this section, we define the data that we use for testing our methods of classification and reconstruction. We consider eight classes of optical quantum states, which we define in Sec. III A. The data for these states is given by measurements consisting of applying coherent displacements followed by sampling of the photon number distribution for the resulting state, as we explain in Sec. III B. We consider six types of noise, described in Sec. III C, that can distort the data.

A. Optical quantum states

Optical quantum states are states of photons, i.e., of bosonic fields. In general, such states live in an infinite-dimensional Hilbert space, but we can obtain a finite-dimensional description by introducing a cutoff on the energy of the state. In the Fock basis for a single bosonic mode, a harmonic oscillator, the state is written as

$$|\psi\rangle = \sum_{n=0}^{N-1} c_n |n\rangle, \tag{23}$$

where n represents photon number, N is the size of the Hilbert space, and c_n are complex-valued amplitudes such that $\sum |c_n|^2 = 1$. Pure and mixed states in this Hilbert space are represented as $N \times N$ density matrices ρ.

Throughout this paper, we use a Hilbert-space cutoff of $N_c = 32$, except for some specific examples and demonstrations. We restrict the maximum photon number of the various states to $\lesssim 16$ to avoid artifacts due to truncation once the displacements are applied to these states.

Below, we define the various types of states used in this paper. The first three are well-known, basic classes of quantum optical states. The following four are from bosonic codes, i.e., states that are designed for quantum error correction. For these latter states, we adopt the definitions from Ref. [147], where $\mu = \{0, 1\}$ denotes whether the state encodes logical 0 or 1. Finally, we also use random states as noise for representing mixed states.

1. Fock states

The Fock states `fock` are the eigenstates of the Fock basis,

$$|\psi_{\texttt{fock}}\rangle = |n\rangle. \tag{24}$$

We consider Fock states with photon number $1 \leqslant n \leqslant 16$.

2. Coherent states

Coherent states `coherent` are displaced vacuum states, characterized by the complex displacement amplitude α:

$$|\psi_{\texttt{coherent}}(\alpha)\rangle = |\alpha\rangle = D(\alpha)|0\rangle, \tag{25}$$

where $D(\alpha) = \exp(\alpha a^\dagger - \alpha^* a)$ is the displacement operator and a ($a^\dagger$) is the annihilation (creation) operator of the bosonic mode. The parameter α gives the position of the state in phase space. We consider $10^{-6} \leqslant |\alpha| \leqslant 3$ to keep the mean photon number $|\alpha|^2$ well below the Hilbert-space cutoff.

3. Thermal states

Thermal states `thermal` are mixed states where the photon number distribution follows super-Poissonian statistics:

$$\rho_{\texttt{thermal}}(n_{\text{th}}) = \sum_{n=0}^{N-1} p(n)|n\rangle\langle n|, \tag{26}$$

where the probability distribution for the photons is given by

$$p(n) = \frac{1}{n_{\text{th}}+1}\left(\frac{n_{\text{th}}}{n_{\text{th}}+1}\right)^n, \tag{27}$$

where n_{th} is the mean photon number. We consider thermal states with $n_{\text{th}} \in [0, 16]$.

4. Num states

`Num` states are a specific set of bosonic-code states, consisting of superpositions of a few Fock states, numerically optimized (hence the name `num`) for quantum error correction, and characterized by their average photon number $\bar{n} \in \{1.562, 2.696, 2.770, 4.149, 4.336\}$ [147,148]. The logical states for each code are orthogonal; for $\bar{n} = 1.562$, they are

$$\left|\psi_{\texttt{num}}^{\mu=0}(1.562)\right\rangle = \frac{1}{\sqrt{6}}\left[(7-\sqrt{17})^{\frac{1}{2}}|0\rangle + (\sqrt{17}-1)^{\frac{1}{2}}|3\rangle\right], \tag{28}$$

$$\left|\psi_{\texttt{num}}^{\mu=1}(1.562)\right\rangle = \frac{1}{\sqrt{6}}\left[(9-\sqrt{17})^{\frac{1}{2}}|1\rangle + (\sqrt{17}-3)^{\frac{1}{2}}|4\rangle\right]. \tag{29}$$

5. Binomial states

Binomial states `bin` are bosonic-code states constructed from a superposition of Fock states weighted by the binomial coefficients [147,148]:

$$\left|\psi_{\texttt{bin}}^{\mu}\right\rangle = \frac{1}{\sqrt{2^{N+1}}}\sum_{m=0}^{N+1}(-1)^{\mu m}\sqrt{\binom{N+1}{m}}|(S+1)m\rangle. \tag{30}$$

Here the parameter N plays a similar role as α in coherent states, determining the size of the state. For the parameter S, we use integers in the range [1,10]. Together with the Hilbert-space cutoff N_c, this determines a maximum value for N. We use $2 \leqslant N \leqslant N_c/(S+1) - 1$.

6. Cat states

Cat states `cat` are bosonic-code states consisting of superpositions of coherent states, with the simplest example being $(|\alpha\rangle \pm |-\alpha\rangle)$ up to a normalization. In general, we can define cat states, parametrized by an integer S and a complex displacement α, as projections given by

$$\left|\psi_{\texttt{cat}}^{\mu}\right\rangle = \frac{1}{\mathcal{N}}\Pi_{(S+1)\mu}|\alpha\rangle, \tag{31}$$

where $\mathcal{N}$ is a normalization. The projections are on even or odd Fock states given by

$$\Pi_r = \sum_{m=0}^{\infty} |2m(S+1)+r\rangle\langle 2m(S+1)+r|, \qquad (32)$$

with the variable $r \in \{0, 1, 2, ..., 2S+1\}$. The parameter S corresponds to the number of photon-loss errors that the code can correct for. A simpler formulation for large α, i.e., $2|\alpha|\sin(\frac{\pi}{S+1}) \gg 1$, is an equal superposition of the $2(S+1)$ coherent states $\{|\alpha e^{i\frac{\pi}{S+1}k}\rangle\}_{k=0}^{2S+1}$ around a circle of radius $|\alpha|$. We use $S \in \{0, 1, 2\}$ and $|\alpha| \in [1, 3]$.

7. Gottesmann-Kitaev-Preskill states

Gottesmann-Kitaev-Preskill states `gkp` are bosonic-code states defined on a square grid in phase space [108,147,149]. The ideal `gkp` states can be seen as a superposition of vacuum states displaced to the points of this grid:

$$\left|\psi^{\mu}_{\text{gkp(ideal)}}\right\rangle = \sum_{n_1,n_2\in\mathcal{I}} \mathcal{D}\left(\sqrt{\frac{\pi}{2}}(2n_1+\mu)\right)\mathcal{D}\left(i\sqrt{\frac{\pi}{2}}n_2\right)|0\rangle \qquad (33)$$

with the integers $n_1, n_2 \in \{-\infty, ..., -2, -1, 0, 1, 2, ..., \infty\}$ forming the grid.

However, a finite `gkp` state limits the lattice and adds a Gaussian envelope to make the state normalizable, thus parametrizing the state with a real parameter $\Delta \in [0, 1]$ as

$$\left|\psi^{\mu}_{\text{gkp(finite)}}\right\rangle = \sum_{\alpha\in\mathcal{K}(\mu)} e^{-\Delta^2|\alpha|^2} e^{-i\text{Re}[\alpha]\text{Im}[\alpha]}|\alpha\rangle, \qquad (34)$$

where the complex amplitudes α are calculated from the grid $\mathcal{K}(\mu) = \sqrt{\frac{\pi}{2}}(2n_1+\mu)) + i\sqrt{\frac{\pi}{2}}n_2$ with some finite cutoff for n_1, n_2. We use $n_1, n_2 \in \{-20, 20\}$ and $\Delta \in [0.2, 0.5]$.

8. Random states

Random states are mixed states generated using the QuTiP [150,151] function `rand_dm` by choosing a density (proportion of nonzero elements) for the density matrix ρ_{random}. The elements of ρ_{random} are sampled from a uniform distribution, ensuring that the density matrix is physical (Hermitian, positive semidefinite, and with unit trace). We choose the density 0.8 for all tasks in this paper and allow ρ_{random} to be full-rank mixed states.

B. Measurements

Measurements on optical states are usually performed with a displace-and-measure technique. Applying a coherent displacement of amplitude β and measuring the photon number distribution gives the generalized Q function [152]

$$Q_n^{\beta} = \text{tr}(|n\rangle\langle n|D(-\beta)\rho D^{\dagger}(-\beta)), \qquad (35)$$

From the generalized Q function we can easily obtain other quasiprobability distributions describing the state, e.g., the Husimi Q function (photon field quadratures)

$$Q(\beta) = (1/\pi)Q_0^{\beta} \qquad (36)$$

and the Wigner function [153] (photon parity)

$$W(\beta) = (2/\pi)\sum_n (-1)^n Q_n^{\beta}. \qquad (37)$$

In Fig. 2, we plot the Q_n^{β} functions for a `binomial` state to illustrate the different types of data. We also show how combining the various levels of the generalized Q function leads to the Wigner function.

In this paper, we mostly consider classification and reconstruction of optical quantum states based on Husimi-Q-function data, but our methods can also be used with Wigner-function data (as we show when reconstructing a state from experimental data in Ref. [99]), generalized-Q-function data, or data from any other observables.

In Fig. 3, we plot Wigner functions and Hinton plots of the density matrices for representative examples of all classes of states defined in Sec. III A above.

C. Noise

Noise is an inevitable factor in most experiments. Thus, methods for state classification and reconstruction should be made sufficiently robust against various types of noise. In this subsection, we define the different types of noise that we use to test our neural network based classification and reconstruction.

Noise can enter the problem at different stages. First, the preparation of the state to be classified or reconstructed could have errors that lead to a slightly different state, $\rho \rightarrow \rho_{\text{noisy}}$ (state-preparation errors). Second, the measurement protocol could have errors due to calibration such that we are not measuring exactly what we sought out to measure, $\{\mathcal{O}_i\} \rightarrow \{\mathcal{O}_i^{\text{noisy}}\}$ (measurement errors). Lastly, there can be errors in the data collection, e.g., errors incurred during amplification of the signal or photon shot noise, which corrupts the data, $\mathbf{d} \rightarrow \mathbf{d}_{\text{noisy}}$ (data errors).

The state-preparation and measurement (SPAM) errors can be systematic and thus hard to correct. Recently, deep neural networks have been demonstrated to be effective in learning such errors and correcting them [98] by training a supervised model to correct the data $\mathbf{d}_{\text{noisy}} \rightarrow \text{DNN} \rightarrow \mathbf{d}$. The neural network is thus used as a sophisticated filter to denoise experimental data, which can be agnostic to the underlying SPAM noise. In this paper, during reconstruction, we do not train our networks to correct SPAM errors; we only deal with specific errors on a case-by-case basis. But for classification, we show that the neural network approach is robust against the various types of SPAM and data errors defined below.

1. Mixed states

In many experiments, thermal and other environmental noise will affect the quantum state. We model this noise by considering mixed states [see Fig. 4(a)]

$$\rho_{\text{mixed}} = (1-\zeta)\rho + \zeta\rho_{\text{random}}, \qquad (38)$$

with $\zeta \in [0, 0.5]$. In the classification task, the correct label for such a mixed state is defined to be that of the class that ρ belongs to. In the reconstruction task, the aim would not be to

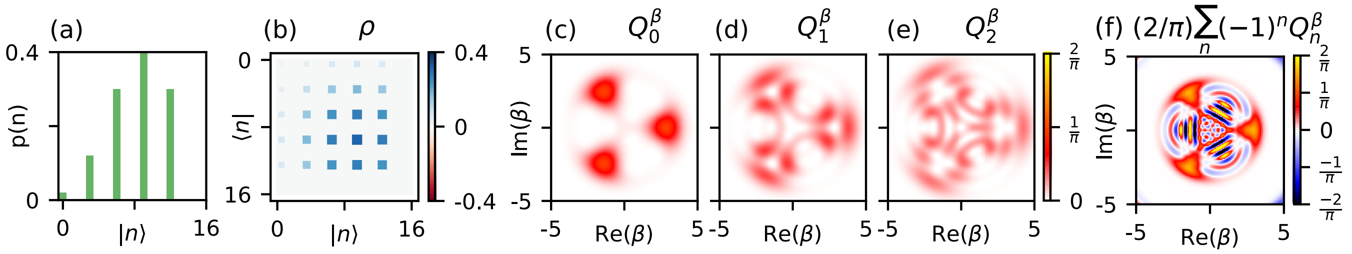

FIG. 2. A `binomial`$(S = 2, N = 5, \mu = 0)$ state and data generated from a displace-and-measure calculation using QuTiP [150,151] within a 200×200 grid. (a) The photon occupation probabilities, i.e., the diagonal elements of the density matrix ρ. (b) A Hinton plot of ρ, where blue (red) denotes that the real part of the density-matrix element is positive (negative). The size and the shade of each square is determined by the absolute value of the density-matrix element. [(c),(d),(e)] The generalized Q function, Q_n^β, for $n = 0, 1, 2$. (f) The corresponding Wigner function computed using the different Q_n^β as $(2/\pi)\sum_n(-1)^n Q_n^\beta$. Note that even when choosing a Hilbert-space cutoff of 100 for this demonstration, the corners in the Wigner-function plot have spurious nonzero values at large displacements $\beta \approx \pm 5 \pm 5i$. To mitigate such effects, larger cutoffs are required for states that have a high photon number or we need to restrict the computation to smaller values of β. Other methods of computing the Wigner function from ρ do not suffer such problems even with a cutoff of 16 for this specific example. QuTiP provides several such implementations and we use one of them, the numerically stable Clenshaw method, to compute Wigner functions in the rest of the paper.

reconstruct ρ, but to reconstruct ρ_{mixed}, since that is the actual state created in the experiment.

2. *Convolution with Gaussian noise during amplification*

In a measurement scheme, which uses linear amplification detectors, one of the effects of noise is modelled by considering additional bosonic modes coming from the amplifier channel [154]. The Husimi Q function in the presence of such linear noise channels [see Fig. 4(b)] is a convolution

$$Q_{\mathrm{noisy}}(\beta') = \int_\beta P_h(\beta'^* - \beta^*)Q(\beta), \tag{39}$$

where $P_h(\beta')$ is the Glauber-Sudarshan P function [155] of the noise mode. While at optical frequencies the noise mode is nearly in the vacuum state, such that $Q_{\mathrm{noisy}}(\beta') \sim Q(\beta')$, at microwave frequencies, the noise mode is in a thermal state. In this case,

$$P_h(\beta) = \frac{1}{\pi n_{\mathrm{th}}} \exp\left(-\frac{|\beta|^2}{n_{\mathrm{th}}}\right). \tag{40}$$

Therefore, the effect of noise is simply applying a Gaussian convolution with the variance n_{th}. Note that such a noise is also interpreted as detection efficiency error with the reduced detection efficiency $\eta = 1/(1 + n_{\mathrm{th}})$. We consider such noise during reconstruction tasks by allowing it as an input, which is easily estimated in experiments, e.g., the detector efficiency or thermal photons in the amplification channel.

3. *Photon loss*

If the optical quantum state is created in a lossy resonator, photons may leak out from this resonator before the measurement of the state is completed. We model such photon loss [see Fig. 4(c)] by letting the original state evolve for some time τ according to the master equation

$$\dot{\rho} = -\frac{i}{\hbar}[H, \rho] + \gamma\mathcal{L}[a]\rho, \tag{41}$$

where $H = \hbar\omega a^\dagger a$ is the free resonator Hamiltonian, ω is the resonator frequency, γ is the photon loss rate, and $\mathcal{L}[a]\rho = a\rho a^\dagger - \frac{1}{2}a^\dagger a\rho - \frac{1}{2}\rho a^\dagger a$. Similar to the case of mixed states

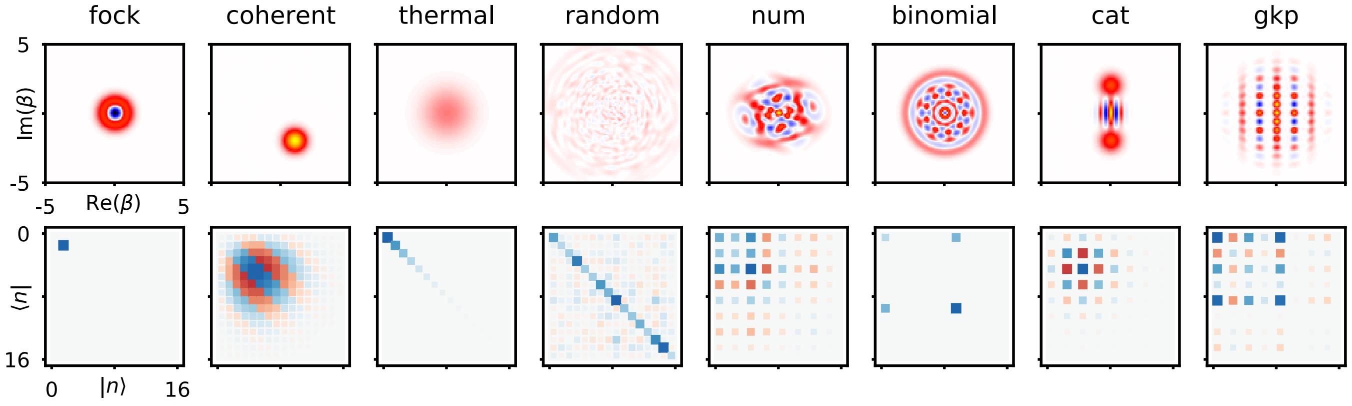

FIG. 3. Representative examples from each class of optical quantum states considered in this paper. In the top row, we plot the Wigner function for the states, using the same scaling as Fig. 2(f). In the bottom row, we show the values of the density-matrix elements for each state as Hinton plots similar to Fig. 2(b). We can see that the Wigner functions and density matrices have characteristic patterns that a neural network can learn and use for classification or reconstruction.

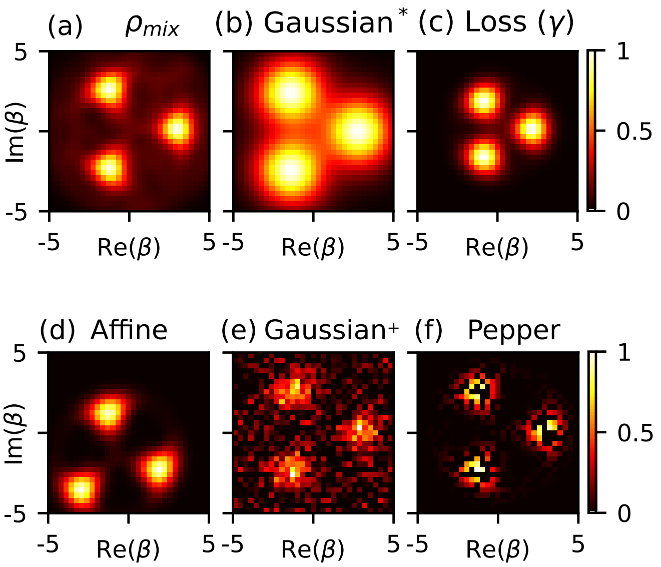

FIG. 4. The effect of various types of noise on the measurement data (Husimi Q functions) for the state in Fig. 2. We normalize the data to [0, 1] by dividing with the maximum value and will use this color scheme throughout the paper to represent such rescaled data. (a) Mixed states with $\zeta = 0.5$ and density 0.8 for ρ_{random}. (b) Convolution with Gaussian noise with $n_{\text{th}} = 3$. (c) Photon loss with $\gamma\tau$ such that 50% of the average initial photons have been lost. (d) Affine transformation with rotation $\theta = 100^{\circ}$, shear $\Omega = 5^{\circ}$, and $\Delta x = \Delta p = 1.61$ (5 pixels). (e) Additive Gaussian noise with standard deviation $\sigma = 0.2$. (f) Pepper noise setting 50% of the data points to zero.

in Sec. III C 1, in the classification task, the correct label is defined to be that of the class that $\rho(t = 0)$ belongs to, while in the reconstruction task, such a noise is not necessarily an error as the aim is to reconstruct $\rho(t = \tau)$.

4. *Affine transformations*

An affine transformation is a geometric transformation that can be represented as a composition of a linear transformation and a translation. In two-dimensional (2D) images, it preserves lines and parallelism, but allows for effects such as rotations, displacements, reflections, scaling, and shearing. Our motivation for this type of noise is that such effects can mimic SPAM errors, e.g., poorly calibrated displacement pulses, squeezing, and rotations of the state. We therefore consider rotations, displacements, scaling, and shearing to distort the training data (2D images of Husimi Q or Wigner functions), see Fig. 4(d).

If (x, p) represent the position and momentum values in the phase space, i.e., [Re(β), Im(β)], the affine transformation $(x, p) \rightarrow (X, P)$ can be parametrized by the scaling factors (s_x, s_y), rotation angle θ, the shear Ω, and linear displacements Δx, Δp as

$$X = s_x x \cos(\theta) - s_y p \sin(\theta + \Omega) + \Delta x, \tag{42}$$

$$Y = s_x x \sin(\theta) + s_y p \cos(\theta + \Omega) + \Delta p. \tag{43}$$

We use the TensorFlow [156] implementation for data augmentation that applies such transformations, with the values of the parameters randomly selected within a certain range for each image augmentation: $\theta \in [0, 180^{\circ}]$; $\Omega \in [0, 5^{\circ}]$; $(\Delta x, \Delta p) \in [-2, 2]$ such that the pixels of the images are shifted up to 20% of the image size. The range for scaling the image (zoom) is set to 0.2 to allow shrinking or expanding the images within a factor [0.8, 1.2] of the original size. We also allow the images to be flipped horizontally and vertically. The data augmentation described here is only used in the classification task.

5. *Additive Gaussian noise*

Measuring the expectation value of a quantum observable often requires repeated measurements to find the average value with good precision. Thus, a limited number of measurements will reduce the precision. Moreover, the precision can also be reduced by binning of measurement results from nearby points in the phase space. We model these types of uncertainty in the data by adding randomly sampled values from a Gaussian distribution $\mathcal{N}$ with zero mean and standard deviation σ to each data point as

$$d_{\text{noisy}} = d + \mathcal{N}(0, \sigma). \tag{44}$$

See Fig. 4(e) for an example.

6. *Pepper noise*

Salt-and-pepper noise represents a corruption of data where the signal changes drastically at a few points. We use pepper noise [see Fig. 4(f)] to represent dead pixels or missing data by selecting a random proportion of data points and setting them to zero.

IV. METHODS

In this section, we present the details of how we use deep neural networks for the two tasks—classification (quantum-state discrimination) and reconstruction (obtaining the density matrix) using the data discussed in Sec. III. Three different neural-network architectures are considered: `Classifier`, `Generator`, and `Discriminator`. We provide the methods and parameters for training and evaluation of the networks that we have used to obtain our results in this paper (Sec. V) and in Ref. [99].

The selection of the neural-network architecture [157], optimizers [158,159], and other non-trainable hyperparameters is a challenging task called hyperparameter tuning [160]. In our paper, we have not used any specific methods for hyperparameter tuning. Instead, we choose our network architectures and hyperparameters inspired by existing successful implementations from the machine-learning community and manually tweaked them ourselves by trying different combinations.

A. Classification

The problem of quantum state discrimination can be considered as a classification task, a task for which deep neural networks have shown impressive results. The input data **d** consists of observed frequencies for some measurement, which is related to the probabilities of outcomes of observables. The output is a label ∈

{`fock`, `coherent`, `thermal`, `cat`, `bin`, `num`, `gkp`}. The neural network we use for the classification is a standard convolutional neural network, which we train by minimizing cross-entropy loss using backpropagation.

1. Input and output data

Since we consider optical quantum states, the data, e.g., the Husimi Q or Wigner function of the state, can be rearranged into an image on a grid determined by the real and imaginary parts of the displacements β. Our training dataset is generated by randomly constructing states from the seven classes discussed in Secs. III A 1–III A 7, adding noise in the form of random mixed states (see Secs. III A 8 and III C 1) and then calculating the Husimi Q functions of the resulting states for the fixed set of β values evenly spaced in a 32×32 grid with $\beta \in [-5, 5]$.

We use 43 762 states for training and 8670 states for testing. The input values are normalized to the range [0, 1] by dividing each data instance with the maximum value. In the training phase, affine transformations (see Sec. III C 4) and additive Gaussian noise (see Sec. III C 5) with σ randomly selected between [0, 0.05] are applied to the data. The addition of noise has a dual purpose—preventing overfitting and mimicking the effects of measurement noise. In the testing phase, we consider the impact of different types of noise separately.

The output labels are encoded in a 7-dimensional vector using a one-hot encoding, $\{t_i\}$ with $t_i \in \{0, 1\}$ and $t_i = 1$ denotes that the input state has been labeled as belonging to the class i.

Note that the full generalized Q function (see Sec. III B) could be represented as a multichannel image $n \times n \times N_c$, where $n \times n$ is the grid of β values and N_c is the photon-number cutoff. Similarly, we can just input the flattened data vector $\mathbf{d}$ for other types of measurements that cannot be seen as an image. However, for data in such form, using convolutional layers in the neural network would not make much sense, since there may not be any spatial correlations in the data.

2. Network architecture

The `Classifier` network, illustrated in Fig. 5 and detailed in Table II, is a convolutional neural network (CNN). Its first six layers consists of blocks of convolution [162] layers that extract geometric features from the input image. After the first six layers, the output is flattened and fed through two fully connected layers that output a 7-dimensional vector for each input image.

We use the activation function “LeakyReLU” [163] for all layers except the final output. The final output layer has 7 neurons, with outputs $\{y_i\}$, one for each class. We apply a softmax activation to these outputs,

$$\text{softmax}(\mathbf{y})_i = \frac{\exp(y_i)}{\sum_j y_j}, \tag{45}$$

to normalize the outputs such that they can be interpreted as the probability of the input data belonging to one of the seven classes. We assign the predicted label for the input state to the output that has the highest probability.

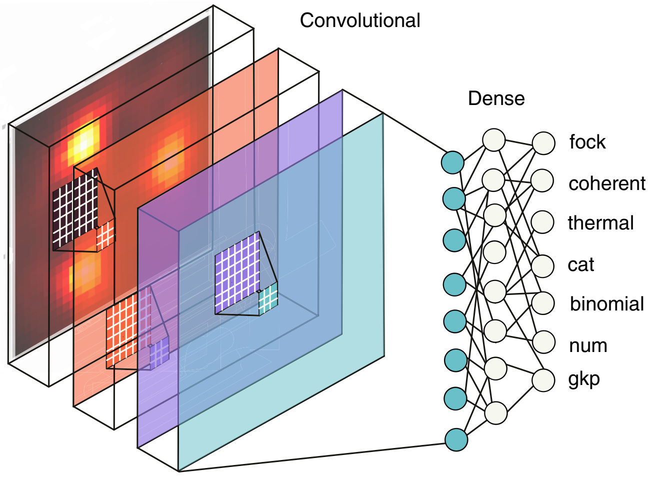

FIG. 5. Sketch of the `Classifier` network, which classifies optical quantum states from Husimi Q data (input image on the left). The blocks represents convolution operations, where filters (exemplified by boxes connecting one layer to another) extract features from the image. We use six such convolutional layers in our architecture (we only show three here). The extracted features are fed to the first of three fully connected layers. The outputs of the last layer are converted to a classification label. For the parameters used, see Table II.

3. Training

The parameters of the `Classifier` network are trained by minimizing the average cross-entropy loss between the predicted probabilities softmax(y_i) in Eq. (45) and the one-hot encoded target labels t_i, defined as

$$\text{cross-entropy}(\mathbf{t}, \mathbf{y}) = -\sum_i t_i \ln\,[\text{softmax}(\mathbf{y})_i]. \tag{46}$$

We use the gradient-based optimizer Adam [11] with a learning rate $l = 0.0002$ and exponential decay rates for first and second moment estimates, $m_1 = 0.5$, $m_2 = 0.5$, to minimize the cross-entropy loss.

TABLE II. Definitions, shapes, and number of trainable parameters for the layers of the `Classifier` network. We denote the convolution layers as Conv2D(f, k, s) where f, k, and s represent the filter size, kernel size, and strides, respectively. After each convolution layer and the first dense layer, the activation function LeakyReLU is used. A full implementation of the code as a TensorFlow model can be found in Ref. [161].

Layer	Output shape	No. of Parameters
Conv2D (32, 3, 1)	30, 30, 32	288
Conv2D (32, 3, 1)	28, 28, 32	9216
Conv2D (32, 5, 2)	14, 14, 32	9216
Conv2D (64, 3, 1)	12, 12, 64	18 432
Conv2D (64, 3, 1)	10, 10, 64	36 864
Conv2D (64, 5, 2)	5, 5, 64	36 864
Dense	512	524 800
Dense	256	131 328
Dense, output y_i	7	1799
Total parameters		768 807

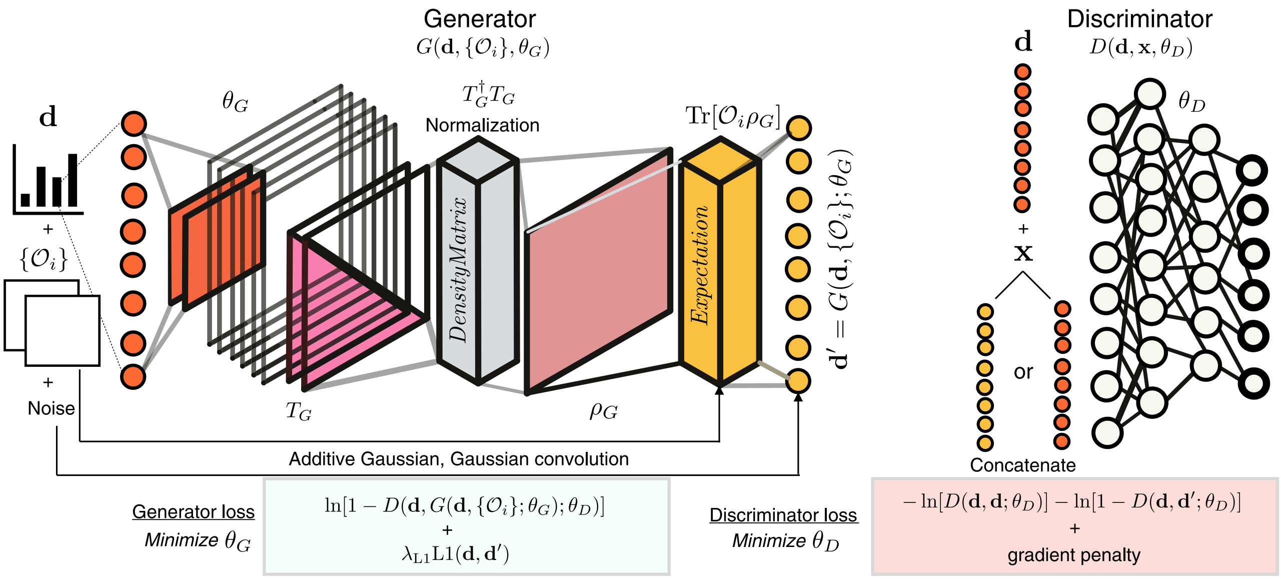
Generator
$G(\mathbf{d}, \{\mathcal{O}_i\}, \theta_G)$
$\mathbf{d}$
+
$\{\mathcal{O}_i\}$
+
Noise
θ_G
$T_G^\dagger T_G$
Normalization
DensityMatrix
$\mathrm{Tr}[\mathcal{O}_i \rho_G]$
Expectation
$\mathbf{d}' = G(\mathbf{d}, \{\mathcal{O}_i\}; \theta_G)$
T_G
ρ_G
Additive Gaussian, Gaussian convolution
Generator loss
Minimize θ_G
$\ln[1 - D(\mathbf{d}, G(\mathbf{d}, \{\mathcal{O}_i\}; \theta_G); \theta_D)]$
+
$\lambda_{\mathrm{L1}} \mathrm{L1}(\mathbf{d}, \mathbf{d}')$
Discriminator
$D(\mathbf{d}, \mathbf{x}, \theta_D)$
$\mathbf{d}$
+
$\mathbf{x}$
or
Concatenate
θ_D
Discriminator loss
Minimize θ_D
$-\ln[D(\mathbf{d}, \mathbf{d}; \theta_D)] - \ln[1 - D(\mathbf{d}, \mathbf{d}'; \theta_D)]$
+
gradient penalty

FIG. 6. Sketch of the `Generator` G and `Discriminator` D neural networks adapted for quantum state reconstruction. The two inputs to G are the measurement statistics $\mathbf{d}$ and the observables $\{\mathcal{O}_i\}$. The input $\mathbf{d}$ is taken as a flattened vector, which is reshaped to a $16 \times 16 \times 2$ matrix after the first layer of G. Then, after successive transpose convolution operations, we obtain a $32 \times 32 \times 2$ matrix. This intermediate output is converted into a lower-triangular matrix with real elements on the diagonal to obtain a Cholesky decomposition form, T_G, that can yield a valid density-matrix representation ρ_G. The expectation values of the observables are then computed using the Born rule $\mathrm{tr}(\mathcal{O}_i \rho_G)$. In the last layer, any known source of noise is added to the outputs. The inputs to D are a concatenation of $\mathbf{d}$ with either the generated data from G or $\mathbf{d}$ itself. The output is interpreted as a similarity score between the inputs (the score is $\sim$1 if they match, i.e., for inputs $\sim\mathbf{d}$). The weights of the two networks, θ_G and θ_D are updated alternatingly to minimize their respective loss functions. For the details of all the parameters, see Table III and Table IV.

During training, we apply the dropout regularization technique [164], where the output of a random fraction of neurons is ignored at each step of optimization, to prevent overfitting. We use 40% dropout after the second, fourth, and sixth convolutional layers, and after the first dense layer. To further prevent overfitting, we also add a small (additive) Gaussian noise ($\sigma = 0.005$, see Sec. III C 5) after the second and fourth convolutional layers.

B. Reconstruction

We now show how a standard neural network can be used to reconstruct the density matrix ρ of a quantum state by adding custom layers to a generative model. The standard formulation of a generative model with feed-forward neural networks (see Sec. II D 3) is a map between a latent space and the data space. Our data $\mathbf{d}$ consists of single shots or average values of measurement outcomes for operators $\{\mathcal{O}_i\}$. We construct a `Generator` network parametrized by weights θ_G, that first estimates a density matrix ρ_G. We then use a custom *Expectation* layer that can generate the statistics for new measurements $\mathbf{d}'(d_i' = \mathrm{tr}(\mathcal{O}_i' \rho_G)$:

$$\{\mathbf{d}, \{\mathcal{O}_i\}\} \xrightarrow[G(\mathbf{d}, \{\mathcal{O}_i\}; \theta_G)]{\texttt{Generator}} \rho_G \xrightarrow[\mathrm{tr}(\mathcal{O}' \rho_G)]{\textit{Expectation}} \{\mathbf{d}'\}. \qquad (47)$$

The `Generator`-network formulation, depicted in Fig. 6, resembles a VAE (see Sec. II D 2), but rather than modeling the data distribution using a parametrization with a mixture of Gaussians, we instead use the straightforward parametrization given by the estimated density matrix itself, ρ_G. The mapping between the latent space of measurement operators $\{\mathcal{O}_i'\}$ and the outcomes is simply $\mathrm{tr}(\mathcal{O}_i' \rho_G)$, which is the data generation map.

TABLE III. Definitions, shapes, and number of trainable parameters for the layers of the `Generator` network. We denote the transpose convolution layers as Conv2D-T(f, k, s) where f, k, and s represent the filter size, kernel size, and strides respectively. After the first dense layer and the first three Conv2D-T layers, the activation function LeakyReLU is used. Instance normalization is used between the first two Conv2D-T layers. The output from the last Conv2D-T layer passes through two custom neural network layers: a *DensityMatrix* layer generating ρ_G, and an *Expectation* layer generating expectation values. A full implementation of the code as a TensorFlow model can be found in Ref. [161].

Layer	Output shape	No. of Parameters
Dense	512	524 288
Reshape	16, 16, 2	0
Conv2D-T (64, 4, 2)	32, 32, 64	2048
Instance normalization	32, 32, 64	128
Conv2D-T (64, 4, 1)	32, 32, 64	65 536
Instance normalization	32, 32, 64	128
Conv2D-T (32, 4, 1)	32, 32, 32	32 768
Conv2D-T (2, 4, 1)	32, 32, 2	1024
DensityMatrix	32, 32	0
Expectation	4096	0
Total parameters		625 920

TABLE IV. Definitions, shapes, and number of trainable parameters for the layers of the `Discriminator` network. The activation function LeakyReLU is used for all layers except the final output layer. A full implementation of the code as a TensorFlow model can be found in Ref. [161].

Layer	Output shape	No. of Parameters
Concatenate	2048	0
Dense	128	1 048 704
Dense	128	16 512
Dense	64	8256
Dense	64	4160
Total parameters		1 077 632

Below, we define the input and output data for the `Generator` network. Then, we show the details of the network architecture with our customized layers that regularize the intermediate output ρ_G to a valid density matrix and generates the correct output. Finally, we discuss the training methods used to optimize the parameters of the `Generator` network. The first training method focuses on minimizing the least-squares and cross-entropy loss between the expected output and generated output. The second method learns a more sophisticated loss function in the form a second, trainable neural network, a `Discriminator`, also illustrated in Fig. 6. This second training method is inspired by the idea of CGANs [12,21], which we use for quantum state tomography (QST-CGAN) [99].

1. Input and output data

The input data for reconstruction are the measurement statistics $\mathbf{d}$ and the operators $\{\mathcal{O}_i\}$ that were measured. Similar to the classification task, we consider the Husimi Q function in a 32×32 grid with $\beta \in [-5, 5]$. The measurement operators $\mathcal{O}_i$ are 32×32 complex-valued matrices. Therefore the input data for a single reconstruction is a combination of the flattened data vector $\mathbf{d}$ (1×1024) and the set of operators $\{\mathcal{O}_i\}$ $(1 \times 1024 \times 32 \times 32)$. Note that it is easy to change the parameters in the data or the neural-network architecture to allow arbitrary phase-space grid sizes and Hilbert-space cutoffs; the fact that they are both set to 32 in most examples here does not have any special significance.

The training data for a single reconstruction thus requires only these 1024 data points (real-valued numbers) and the 1024 operators (complex-valued matrices) as the input for each reconstruction. We consider noise on a case-by-case basis during training (described in Sec. IV B 3 below).

The output of the neural network is a (1×1024) vector representing the expectation values for the measurements $\{\mathcal{O}_i\}$. Inside the `Generator`, the full density matrix of the state is estimated as a $1 \times 32 \times 32$ complex-valued matrix ρ_G determined by the outputs of an intermediate *DensityMatrix* layer.

In this way, we allow for a flexible architecture, which can reconstruct a single state with inputs shaped as $(1 \times 1024, 1 \times 1024 \times 32 \times 32)$ for $(\mathbf{d}, \{\mathcal{O}_i\})$ or allow multiple states as the input simply by concatenating the inputs. For example, to reconstruct 10 states simultaneously with 1024 measurements each, we simply feed the network a batch of data points as $(10 \times 1024, 10 \times 1024 \times 32 \times 32)$.

In this paper, we only consider single reconstructions, so our inputs will always be of the shape $1 \times n$ for the data $\mathbf{d}$ and $1 \times n \times N_c \times N_c$ for the measurements, where n is the number of measurement settings and N_c is the Hilbert-space cutoff. Note that we allowed the most general description of the measurement setting in the inputs as the full operator descriptions $\{\mathcal{O}_i\}$. We could also use alternative ways to specify the measurement settings, e.g., a set of complex displacements β_i, and redefine our *Expectation* layer to use those β values. In the case of qubit tomography, these measurement settings can be replaced with a set of single-qubit measurement operators such as $[Z, X, X, Z, \ldots]$.

2. Network architecture

Our `Generator` network G is a modified version of the standard $G(\mathbf{z}; \theta)$ formulation (see Sec. II D 3), where we first consider the conditional form $G(\mathbf{z}|(\mathbf{d}, \{\mathcal{O}_i\}); \theta)$. The conditioning variable is our data and the measurement settings, represented as a vector and a set of matrix operators, respectively. Then, inspired by the *pix2pix* architecture [21], we remove the random noise $\mathbf{z}$ and just consider the data and measurement operators as inputs to define $G(\mathbf{d}, \{\mathcal{O}_i\}; \theta)$ as the `Generator`.

The full architecture, detailed in Table III and depicted in Fig. 6, begins with a fully connected dense layer, which receives the flattened data vector $\mathbf{d}$ as input. The output of this layer is reshaped to a $16 \times 16 \times 2$ tensor. This layer converts the input into a matrix with two channels that can be upsampled into the density matrix. The next layers are three blocks of two-dimensional transpose convolution operations (Conv2D-T) and instance normalizations [165] such that the final output is moulded to an estimate of the density matrix ρ_G. All the layers described so far use LeakyReLU activation, except the final Conv2D-T layer, whose outputs are fed to a custom *DensityMatrix* layer.

The *DensityMatrix* layer converts the output of the final Conv2D-T layer to a valid density matrix. This output is two matrices $(32 \times 32 \times 2)$, which are combined into one 32×32 complex-valued matrix, T_G. The upper triangular part of T_G and the imaginary part of the diagonal are set to zero to obtain the Cholesky decomposition of a Hermitian matrix [Eq. (1)]. Finally, we divide the resulting matrix by its trace to obtain a valid density matrix. Therefore, the custom *DensityMatrix* layer can convert the real-valued outputs of any standard neural network to a Hermitian, positive-semidefinite matrix with unit trace.

The final layer is another custom one, called *Expectation*. It takes as input $\{\mathcal{O}_i\}$ during training (the other part of the input to the `Generator`) and outputs the expected values for measurement outcomes for each component of $\mathbf{d}'$ as

$$d'_i = \mathrm{tr}(\mathcal{O}_i \rho_G). \tag{48}$$

The last two layers, *DensityMatrix* and *Expectation*, do not contain any trainable parameters.

The `Discriminator` network used to train the generator is detailed in Table IV. This network receives two inputs: the data $\mathbf{d}$ and the generated statistics $\mathbf{d}'$, and begins by concatenating the two. The concatenated input is then passed through four dense layers, with the final layer having 64 neurons. All the layers of the discriminator use the LeakyReLU activation, except the final layer, whose outputs are interpreted as a measure of the similarity between $\mathbf{d}'$ and $\mathbf{d}$. Note that the dimensions, or even the shape, of the final output layer can be arbitrary. The outputs simply need to be interpret as a similarity score between $\mathbf{d}$ and $\mathbf{d}'$, which should be $\sim$1 if $\mathbf{d} \sim \mathbf{d}'$ and $\sim$0 otherwise. We were inspired by the *Patch*GAN idea [21] for our `Discriminator` that motivates penalties at the scale of patches in the input. We have also concurrently found during the course of our work, that similar ideas were effectively demonstrated for x-ray tomography with promising results for denoising [166].

3. Training

The training for reconstruction can be done in two ways—either we reconstruct a single state or we reconstruct a set of different states using the same `Generator` network. This flexibility comes from our formulation of the `Generator` network and reshaping of the data to find a map from data space to the set of density matrices. In this paper, we only show how to perform single reconstructions, but in Ref. [99], we show how the same `Generator` network can perform single-shot reconstructions for many different states.

For each reconstruction in this paper, we only consider a single state ρ and the data from measurements of several operators on ρ as the inputs and outputs (see Sec. IV B 1). We train the `Generator` network to minimize a loss metric that gives some measure of how the reconstructed statistics $\mathbf{d}'$, calculated from an underlying ρ_G, differ from the data $\mathbf{d}$. If d_i are the frequencies of measurements $\mathcal{O}_i$ and $d_i' = \text{tr}(\mathcal{O}_i \rho_G)$ are the computed probabilities from the generated density matrix, then maximizing the log-likelihood in Eq. (3) amounts to minimizing the cross-entropy loss between observed frequencies $\mathbf{d}$ and $\mathbf{d}'$:

$$\text{cross-entropy}(\mathbf{d}, \mathbf{d}') = -\sum_i d_i \ln[\text{tr}(\mathcal{O}_i \rho_G)]. \tag{49}$$

However, the cross-entropy loss assumes discrete-valued data, i.e., single-shot outputs of POVMs, whereas in many cases we may be looking at continuous-variable outputs instead.

If we consider the data to be the expectation values of some continuous-valued observable, e.g., the homodyne current, metrics such as the mean squared error

$$\text{L2}(\mathbf{d}, \mathbf{d}') = \frac{1}{q} \sum_i (d_i - d_i')^2, \tag{50}$$

where q is the number of data points, are more suitable. For such continuous-valued data, the error in measurement can be assumed normally distributed with variance σ_i^2. Under this assumption, minimizing the L2 loss maximizes the likelihood

$$L(\rho'|\mathbf{d}) = \prod_i \left[\frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(d_i - \mu_i)^2}{2\sigma_i^2} \right) \right], \tag{51}$$

where we consider the mean for each measurement outcome as the expectation value $\mu_i = \text{tr}(\rho' \mathcal{M})$ for some observable M.

Speaking more generally, the loss function uses some metric to measure the distance between two probability distributions P and Q. Such metrics can be divided into two major classes: ϕ divergences and integral probability metrics (IPMs) [167,168]. The first are of the form

$$D_\phi(P||Q) = \int \phi\left(\frac{\partial P}{\partial Q} \right) dQ \tag{52}$$

where ϕ is some convex function $\phi : R_{\geqslant 0} \to R_{\geqslant 0}$, while the latter are defined as

$$D_{\text{IPM}}(P, Q) = \sup_{g \in \mathcal{G}} \left| \int g\, dP - \int g\, dQ \right|, \tag{53}$$

where the class of functions $\mathcal{G}$ parametrizes some notion of distance.

In the deep-learning community, the study of such metrics in generative modeling is an area of active research [169,170]. It has been shown that the choice of loss function can greatly impact the quality of image reconstruction [171]. There are several recent attempts to gain better understanding of the role of different loss functions in GAN performance, e.g., using the Wasserstein metric [169] or IPMs [172].

Since the best choice of loss function is far from clear, we train the `Generator` network to minimize several different loss functions between predicted Husimi Q values and observed data. We first use the well-known L1, L2, and cross-entropy loss functions, as well as the KL divergence. The latter two are closely related, and belong to the class of ϕ-divergences. The L1 loss is both a ϕ-divergence and an IPM.

Beyond these well-known loss functions, GANs allow for more complex loss functions to be learned. In our QST-CGAN architecture, we train the `Generator` using the `Discriminator` network combined with L1 loss, to minimize

$$\begin{aligned} &\ln\left[1 - D(\mathbf{d}, G(\mathbf{d}, \{\mathcal{O}_i\}; \theta_G); \theta_D)\right] \\ &+\lambda_{\text{L1}}[G(\mathbf{d}, \{\mathcal{O}_i\}; \theta_G) - \mathbf{d}], \end{aligned} \tag{54}$$

with the L1 loss coefficient $\lambda_{\text{L1}} \in \{0, 1, 10, 100\}$. The discriminator loss function maximizes Eq. (19) by minimizing

$$\begin{aligned} &-\ln\left[D(\mathbf{d}, \mathbf{d}; \theta_D)\right] - \ln\left[1 - D(\mathbf{d}, \mathbf{d}'; \theta_D)\right] \\ &+ \lambda_\Delta E[(||\Delta_{\mathbf{x}} D(\mathbf{x}; \theta_D)||_2 - 1)^2], \end{aligned} \tag{55}$$

where the last term is a gradient penalty [173] with weight $\lambda_\Delta = 10$. We combined the inputs to the `Discriminator` as the vector $\mathbf{x}$.

Therefore, in each training iteration, we alternatively update the generator and discriminator weights using backpropagation with the help of some gradient-based optimizer. Since the choice of hyperparameters, e.g., optimizer or learning rate, can significantly affect the rate of convergence, we try to find settings that enable a fast convergence for all loss functions. To make a fair comparison, we keep the same parameters for optimization for all loss functions. We use the Adam optimizer with the exponential decay rates for first and second moment estimates as $m_1 = 0.5$, $m_2 = 0.5$. We also use an exponentially decaying learning rate as a function of iteration number i,

$$l(i) = l_0 C^{\frac{i}{s}} \tag{56}$$

with initial learning rate $l_0 = 0.0002$, decay constant $C = 0.96$, and $s = 1000$ steps.

V. RESULTS

In this section, we characterize the performance of our Classifier and Generator networks in various settings. We first check the performance of the Classifier network, including some types of noise, in Sec. V A 1. We then study, in Secs. V A 2 and V A 3, the impact of photon loss and additive Gaussian noise on classification performance. Finally, in Sec. V A 4, we analyze, which parts of the data the Classifier bases its decision on. This provides information that can help reduce the number of measurements needed in an experiment or guide an adaptive scheme for tomography.

For reconstruction, we first investigate, in Sec. V B 1, the result of using different loss functions, including the Discriminator network, to train the Generator network. We compare the performance of the Generator against maximum-likelihood-based reconstruction algorithms—iterative MLE (iMLE) [174] and the "superfast" APG-MLE [119] (see Sec. II B)—under additive Gaussian noise in Sec. V B 2. In Sec. V B 3, we show how to tackle Gaussian convolution noise. Then, we show the results of reconstruction for mixed states in Sec. V B 4 and finally, in Sec. V B 5, demonstrate how few data points are needed for the network to reconstruct a state well.

A. Classification

1. Confusion matrix

The performance of the Classifier network on a test set is shown as a confusion matrix in Fig. 7(a). The test set consists of ∼1200 different instances of each of the seven classes in Secs. III A 1–III A 7, with noise in the form of state mixing (see Sec. III C 1) applied with $\zeta \in [0, 0.5]$ and density 0.8.

The accuracy of the classification (number of correct classifications divided by the total number of classifications) on the whole test set is 98.6% . For a validation set with the same states as the test set, but where we have added noise in the form of affine transformations (see Sec. III C 4) and additive Gaussian (see Sec. III C 4 with $\sigma \in [0, 0.05]$) on top of the state-mixing noise, the accuracy of the Classifier remains very high, 97.7% .

It is clear from the confusion matrix in Fig. 7(a) that the class which presents challenges for the network is cat. All other classes are correctly identified in virtually every case, but the cat states are misclassified in about 9% of the cases. In these cases, the network misidentifies the cat states as all other classes except thermal, with the most common mislabellings being coherent, fock, and binomial. The reverse misidentification, where a state is misclassified as cat, occurs for about 1% of the binomial states.

A few examples of misclassifications are shown in Fig. 7(b), where we consider pure cat states with low values of α. These examples demonstrate that there are parameters for which states from different classes are very similar. For example, a cat($\alpha = 4, S = 0$) state ρ and a binomial($S = 1, N = 16$) state ρ' have a fidelity

$$F(\rho, \rho') = \left[\mathrm{tr}\left(\sqrt{\sqrt{\rho}\rho'\sqrt{\rho}}\right)\right]^2 \tag{57}$$

greater than 0.99. Note that the fidelity F reduces to the squared overlap $|\langle\psi|\psi'\rangle|^2$ for pure states. Similarly, the fidelity of cat($\alpha = 3, S = 4$) and fock(10) is greater than 0.996. It is thus not surprising that the network found some states hard to classify. A human quantum physicist would likely have made the same misclassifications from the data in Fig. 7(b).

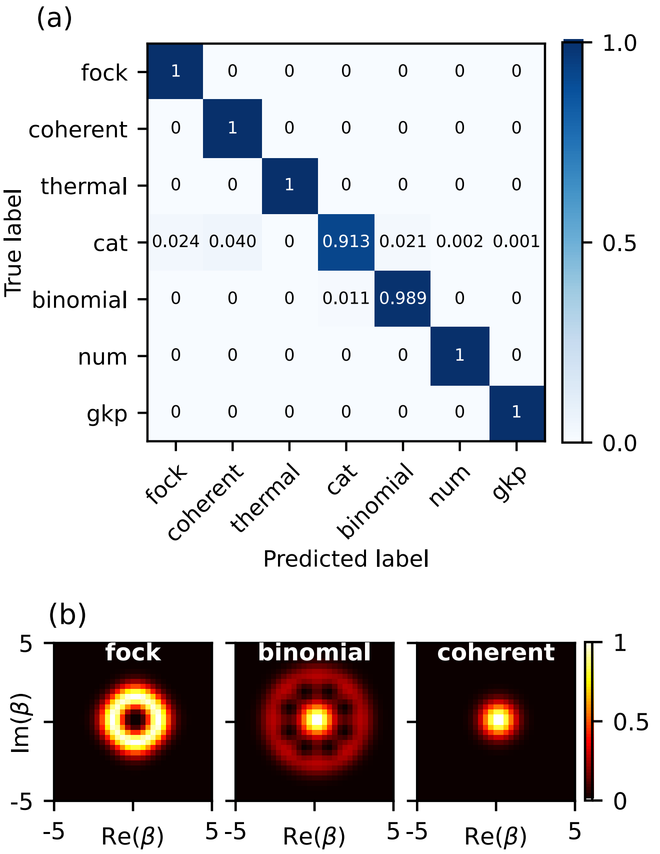

FIG. 7. Performance of the Classifier. (a) Confusion matrix demonstrating the performance of the Classifier on a test dataset containing 8760 states (∼1200 different instances of each class). The prediction counts are normalized to show the true labels versus the predictions made by the Classifier. (b) Husimi Q functions for three examples of pure cat states that the Classifier does not classify correctly. For each state, the incorrectly assigned label is shown. The states are, from left to right, cat($\alpha = 1, S = 1, \mu = 1$), cat($\alpha = 2, S = 3, \mu = 0$), and cat($\alpha = 1, S = 3, \mu = 0$). For certain parameters, different states have a high overlap in fidelities and the measurement data, making classification challenging. The Classifier tries to find the best label according to relevant patterns in the data.

2. Recognizing cat states with photon loss

We now investigate the performance of the Classifier network in the presence of photon loss (see Sec. III C 3). In Fig. 8(a), we show how well the Classifier manages to recognize a set of cat($\alpha, S = 0$) states, with $\mu = 0$ and

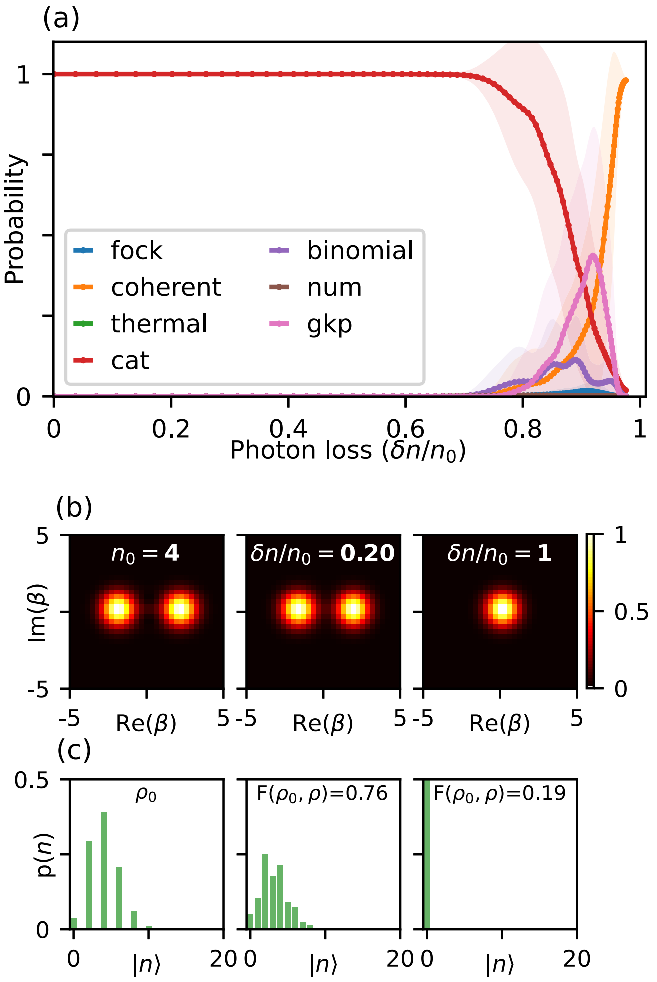

FIG. 8. Classifier performance for cat states with photon loss. (a) Softmax probabilities (solid lines) predicted by the Classifier for the labels of the seven classes. The shaded regions show one standard deviation from the mean. The dataset consists of 100 cat states ($|\alpha| \in [2, 3]$, $S = 0$, $\mu = 0$) with photon loss, quantified by the proportion of photons lost $\delta n/n_0$ starting from the initial mean photon number n_0. (b) Husimi Q functions for one of the cat states in the dataset with, from left to right, 0 %, 20 %, and 100 % of photons lost with respective fidelities 0.76 and 0.19 for the states with photon loss. It is not straightforward to assert from just the Husimi-Q data when a cat state stops being a "cat" as it still possesses cat-like features (two coherent blobs) even after losing 20 % of the initial photons. (c) The photon-number occupation probabilities for the states in (b). Note that the occupation probability for the vacuum state is ~1 in the right panel, but we set the limits of the y axis to 0.5 for better distinguishability.

$|\alpha| \in [2, 3]$, as more and more photons are lost. Before any photons are lost, the softmax probabilities for different labels show that the Classifier assigns the highest probability to the label cat. After ~70 % of the photons have been lost, the probability of the state being classified as a cat decreases and the labels coherent and binomial become equally probable. It is an interesting question whether these probabilities reflect the characteristics of the state in such a way that it could be used as a starting point for reconstruction. When almost all the photons are lost, the classification label is always coherent.

Even though we did not include any photon-loss noise during the training phase, the Classifier is still able to identify cat states after many photons have been lost. It should be noted that once photons have been lost, it is not certain that the state can be considered a cat state anymore. A distinctive feature of cat states is the interference between the coherent states making up the superposition state. This interference results in zero probability of odd photon numbers in the state [see the left panel in Fig. 8(c)]. Once photon loss starts acting on the state, these occupation probabilities become nonzero [see the middle panel in Fig. 8(c)], but the Classifier network can still identify general features leading it to classify the data with the label cat. However, once more photons have been lost, the state ceases to be a cat state and is classified as a coherent state.

We note that the results presented here for classification under photon loss may be different if the network is trained on data in the form of Wigner functions (see Sec. III B) instead of Husimi Q functions. The Wigner function for cat states has characteristic interference fringes, some with negative values, between the coherent-state blobs. These features are not clearly seen in the Husimi Q function; it only takes very small nonzero values ($\sim 10^{-4}$) between the two coherent blobs in a cat($\alpha = 2, S = 0, \mu = 0$) state. Another approach to identify the lossy cat states better would be to train a classifier to distinguish cat states and mixtures of coherent states from the Husimi Q function. Just like the Classifier, this does not require explicitly specifying criteria fr what is or is not a cat, but works in the spirit of "Software 2.0" [175]— replacing explicit programming with learning from data.

3. Classification in the presence of additive Gaussian noise

Next, we test the performance of the Classifier in the presence of additive Gaussian noise. As explained in Sec. III C 5, this type of noise models uncertainty in the data due to averaging over a limited number of measurements and binning of data. In Fig. 9(a), we plot the classification accuracy as a function of the standard deviation σ of the added Gaussian noise (see Sec. III C 5). The dataset is the same as that in Fig. 7, but with the Gaussian noise added. In Fig. 9(b), we show an example of how the Gaussian noise impacts a cat state in the dataset.

The accuracy of the predictions from the Classifier remains high until $\sigma \approx 0.05$ and then decreases gradually. However, even at $\sigma = 1$, the accuracy is almost 25%, clearly better than $\sim 1/7$, which is what one would obtain for a random guess among the seven classes. At these high levels of noise, the network can still correctly classify up to ~60% of the fock states, ~30% of the coherent states, and ~55% of the cat states in the test set. However, at such a high level of noise, the Classifier almost always predicts the label as one of fock, coherent, or cat. Hence accuracy is not the best indicator of performance in all scenarios.

Therefore, in addition to the accuracy, we also quantify the Classifier performance by considering the receiver-operating-characteristic (ROC) curve [176,177]. The ROC curve for a binary classification problem is a plot of the true positive rate (TPR, the ratio between correctly classified positive labels and the number of real positive labels) versus the

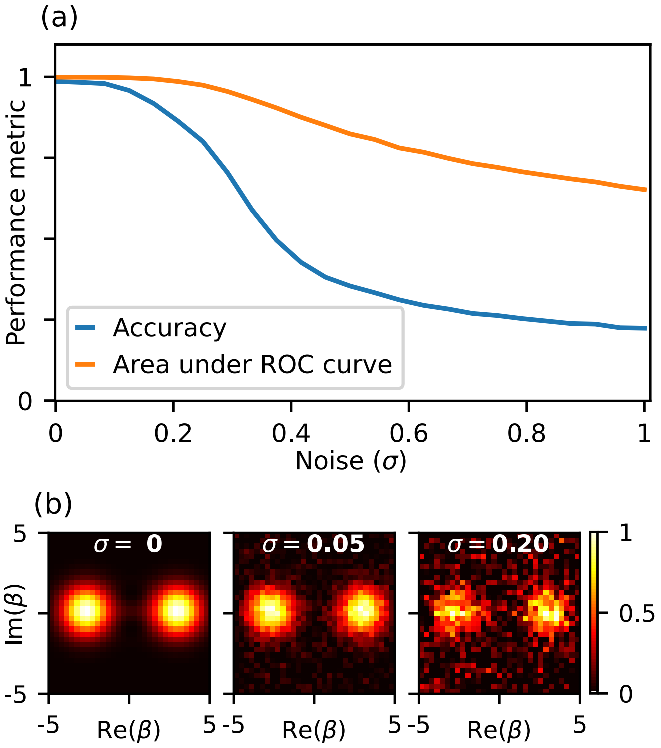

FIG. 9. `Classifier` performance in the presence of additive Gaussian noise. (a) Accuracy (blue) and area under the ROC curve (orange) as a function of the level of additive Gaussian noise. The noise is added to the dataset used in Fig. 7. (b) Effect of additive Gaussian noise on a cat($\alpha = 2, S = 0$) state. The distortion of the state starts to become significant between $\sigma = 0.05$ and $\sigma = 0.2$, which is when the accuracy of the `Classifier` starts to drop below 90%.

false positive rate (FPR, the ratio between false predictions of positive labels and true predictions of negative labels). The area under the ROC curve gives an indication of the discriminative power of the classifier: the area is 1 for perfect classification and 0.5 for random guesses. For our multiclass problem, we use the one-vs-rest strategy in Scikit-learn [178] to calculate the area under the ROC curve. The result, averaged over all classes, is shown in Fig. 9(a). The area under the ROC curve shows a behavior similar to the accuracy, but indicates a somewhat better performance than the latter does.

4. What does the network see?

The `Classifier` network is a highly nonlinear function that maps data to a label. In order to find the patterns in the data that the network uses to determine the label, we apply gradient-weighted class activation mapping (Grad-CAM) [179]. The Grad-CAM method works by fixing a class label and finding the gradients of the score for this target class (before the softmax activation) with respect to the last convolution layer of the network. This is a form of backpropagation that allows us to construct a heatmap of numbers showing, which pixels of the input image influence the output the most.

In Fig. 10(a), we show three examples of noisy input data, from which we calculate Grad-CAM heatmaps, shown in Fig. 10(b). These heatmaps are then used in Fig. 10(b) to show the parts of the noise-free input data that contribute the most to the classification. Affine transformations (see Sec. III C 4) and additive Gaussian noise (see Sec. III C 5) with $\sigma = 0.2$ have been applied to the input data to simulate an experiment with SPAM errors and little averaging. We chose to only show the parts of the data where the heatmap has high values (exceeding 0.9), to demonstrate that, even in the presence of significant noise, the `Classifier` makes its decision based on the data in the regions that contain the important patterns characterizing the state. A nonmachine-learning way to achieve similar results would be to hand-craft an algorithm that can clean noisy data and detect the regions with a high signal using some boundary-finding algorithm. However, instead of hand-crafting solutions for each type of state and noise, our trained `Classifier` can easily adapt to a variety of different scenarios.

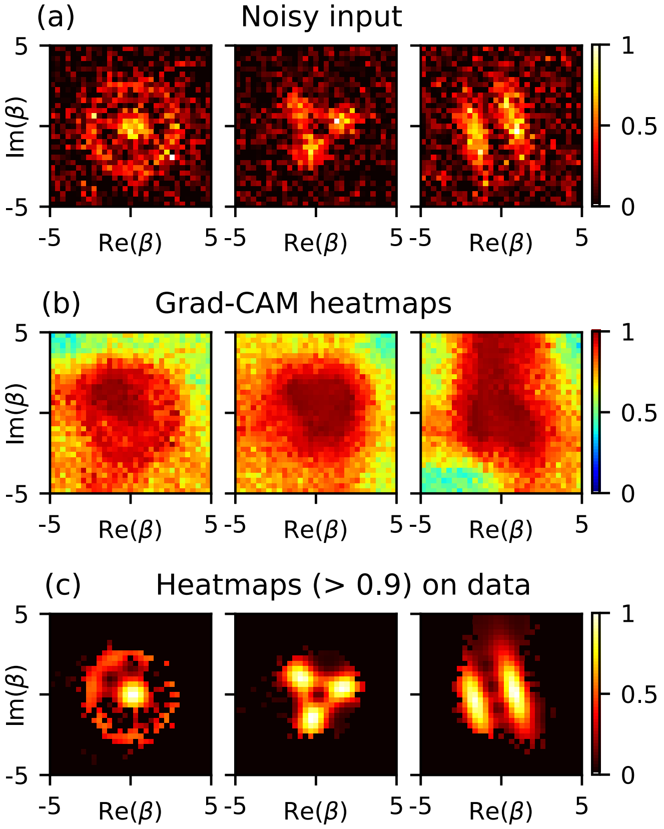

FIG. 10. Using Grad-CAM to highlight the regions of the input data that the `Classifier` considers most important for predicting a label. (a) Input data for three states (from left to right: `binomial`, `num`, and `gkp`) with affine transformations and a constant additive Gaussian noise ($\sigma = 0.2$) for all values of β after normalizing the data to the interval [0, 1]. (b) Heatmaps, normalized to the interval [0, 1], constructed with Grad-CAM from the data in (a), showing which parts of the data the `Classifier` focusses on. (c) The areas of the data (without the additive Gaussian noise) that appear in the focus when we only show the regions for which the Grad-CAM signals exceed 0.9.

The Grad-CAM results suggest an interesting possibility for adaptive tomography: using Grad-CAM during the data collection in an experiment to identify regions that are important and then sample from these regions more that from other places. In this way, our Classifier network can identify specific POVMs (defined by displacements β) that give the most useful data for discriminating optical quantum states.

B. Reconstruction

1. *Impact of loss metric*

We first investigate how the choice of loss function affects the performance of our neural-network reconstruction method. In Fig. 11, we compare the impact of different loss metrics used to train the Generator network. For each loss function, we train the network using the same data. We show the reconstruction fidelity for data from a binomial state reconstructed with six different methods (see Sec. IV B 3). In Fig. 11(a), we show results for the QST-CGAN with various weights λ_{L1} of the L1 loss term in Eq. (54). For all values of λ_{L1}, including $\lambda_{\mathrm{L1}} = 0$, corresponding to pure Discriminator loss, the reconstruction fidelity converges to unity. The convergence is faster with L1 loss added than without it, but a large weight on the L1 part of the loss function leads to worse performance than a moderate weight. The best performance is seen for $\lambda_{\mathrm{L1}} = 1$, when the network converges to the correct reconstruction in a little more than 100 iterations, i.e., 100 updates of our estimate for the density matrix.

The MLE methods, shown in Fig. 11(b), also converge to unit fidelity, but do so using two orders of magnitude more iterations than the best QST-CGAN. However, note that the specific MLE implementation can affect the actual time needed for each iterative step and therefore the total reconstruction time. Similarly, the neural-network architectures will affect the actual training time. Therefore we only compare the fidelities for intermediate states each time any method updates the density-matrix estimate.

In Figs. 11(c)–11(f), we plot the results of training the Generator using the cross-entropy, KL-divergence, L1, and L2 loss functions, respectively. In all cases, the reconstruction fidelity converges to close to unity. The Generator trained with cross-entropy loss [Fig. 11(c)] displays the fastest convergence, on par with the best QST-CGAN. Training with KL-divergence loss [Fig. 11(d)] gives almost as good results. The L1 [Fig. 11(e)] and L2 [Fig. 11(f)] loss functions result in slower convergence, but still perform better than iMLE for the example considered here. We note that the L1 and L2 loss functions lead to a wider distribution of the number of iterations required for convergence for the same data than any of the other methods.

To ensure that the results in Fig. 11 were not particular to the state used as input data there, we also show the results of reconstruction of a cat state in Fig. 12. The results in Fig. 12(a) are similar to those in Fig. 11(a): the QST-CGAN always converges to unit fidelity, and it does so the fastest when L1 loss is added to the Discriminator loss with weight $\lambda_{\mathrm{L1}} = 1$. The main difference to Fig. 11(a) is that the convergence with pure Discriminator loss is considerably faster in Fig. 12(a) and is almost as fast as when L1 loss is added. Just as in Fig. 11, the iMLE method, shown in Fig. 12(b), converges to unit fidelity about two orders of magnitude slower than the best QST-CGAN. The APG-MLE method from Ref. [119] is faster than our iMLE implementation, but requires more iterations to converge for this example.

The plots in Figs. 12(c)–12(f) show the results of training the Generator using the cross entropy, KL-divergence, L1, and L2 loss functions, respectively. Whereas these methods all eventually lead to close to unit fidelities for the reconstruction in Fig. 11, here they all sometimes fail and end up in a state giving reconstruction fidelity zero instead. In the cases where they do end up at unit fidelity, the convergence is approximately as fast as in Fig. 11, perhaps somewhat faster for the L2 loss in Fig. 12(f).

In the cases where the standard loss functions lead the Generator to reconstruct a state with fidelity zero, the reconstructed state is a cat state, shown in the inset of Fig. 12(d), orthogonal to the cat state, shown in the inset of Fig. 12(b), that provides the data. The two cat states have the same α and virtually indistinguishable Husimi Q functions. The only difference between the two is that the correct state has nonzero values in a narrow line along $\mathrm{Im}(\beta) = 0$ between the two prominent lobes in the Husimi Q function, while the orthogonal state has nonzero values at two narrow lines along $\mathrm{Im}(\beta) \approx \pm 0.5$ instead. The differences between the two states are more clearly seen if one plots their Wigner functions instead. We consider reconstruction from Wigner-function samples in Sec. V B 4 and from experimental data in Ref. [99].

For the KL divergence in Fig. 12(d) and the L1 loss in Fig. 12(e), the Generator network seems to start moving towards one of the two cat states and then eventually converge to that state. However, for the L2 loss in Fig. 12(e), there are some runs where the Generator network reconstructs an orthogonal state (with very low fidelity to the target), but then corrects and jumps to the correct state within a few iterations. In the specific case of a cat state, the orthogonal state is reached by applying the photon annihilation operator a to the correct state. It remains to be explored if the Generator network learns to represent quantum states in a way that it can apply such nontrivial quantum operations to find the correct state from an initially incorrect prediction.

In our attempts to tune the hyperparameters of the training, we have noticed that higher values of the parameters m_1 and m_2 for the Adam optimizer removes the behavior seen in Figs. 12(c)–12(f). Instead, for these values the Generator always finds the correct state and not its orthogonal counterpart, similar to how the QST-CGAN in Fig. 12(a) always converges to the correct state. Another way to achieve this convergence could be sampling more around $\mathrm{Re}(\beta)$ to focus on the data that distinguishes the two cat states. In any case, it is noteworthy that the Generator network could possibly apply nontrivial steps to quickly reconstruct the state while the iMLE converges in small steady steps.

To summarize the results in Figs. 11 and 12, the main finding is that the best QST-CGAN reaches unit fidelity orders of magnitude faster than MLE methods. The QST-CGAN performs best when its loss function is an approximately equal mix of L1 loss and the trained Discriminator loss. Further tuning of hyperparameters leads to even better performance in some cases. Among the standard loss functions, cross-entropy loss and KL divergence lead to somewhat better performance for the Generator network than did L1 and L2 loss. However,

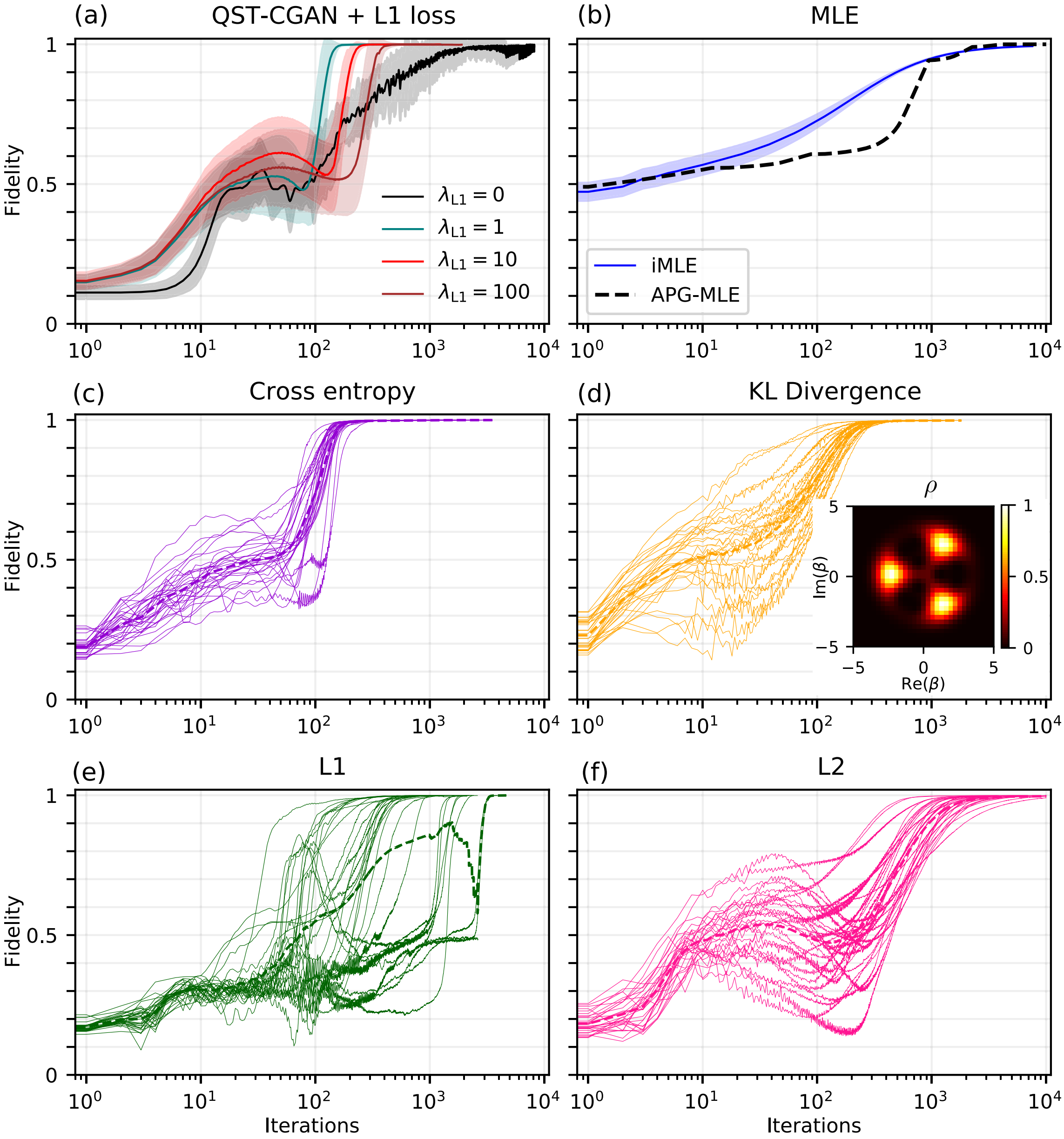

FIG. 11. The effect of the loss function on reconstruction of ρ using the Generator. We use the data from the Husimi Q function of a binomial($S = 2, N = 4, \mu = 0$) state [inset in panel (d)] in a 32×32 grid and repeat the reconstruction with random initializations of the network weights and starting estimate of ρ for iMLE. In each iteration, the weights of the Generator or Discriminator networks are updated using a single step of the Adam optimizer. The learning-rate schedule and optimization hyperparameters are set to the same values (see Sec. IV B 3) for all loss functions in order to achieve a fair comparison. However, further tuning of the parameters for each type of loss function could possibly give better results. (a) The reconstruction fidelity as a function of iterations for QST-CGAN with various weights of the L1 loss set by the λ_{L1} parameter [see Eq. (54)]. In each of a total of 30 runs, the weights of the Generator and Discriminator are randomly initialized. The solid lines show the mean and the shaded regions shows one standard deviation from the mean. (b) The performance of MLE methods on the same data. We repeat the reconstruction 30 times and show the mean fidelity (solid-blue line) and one standard deviation from the mean (shaded region) for iMLE. For the APG-MLE (dashed-black line), we use the default initialization scheme ("bootstrap") from Ref. [119], which initializes the starting density matrix via conjugate gradients using a line search. The plot for APG-MLE shows the improvement of fidelity including the steps during the initialization scheme. There is no deviation from the mean for APG-MLE, since there is no explicit randomization involved in the reconstruction. [(c),(d),(e),(f)] Reconstruction fidelities using standard loss functions for the Generator: cross-entropy [see Eq. (49)], KL divergence, L1, and L2 [see Eq. (50)]. We show all 30 runs for each loss function with the dashed line showing the mean.

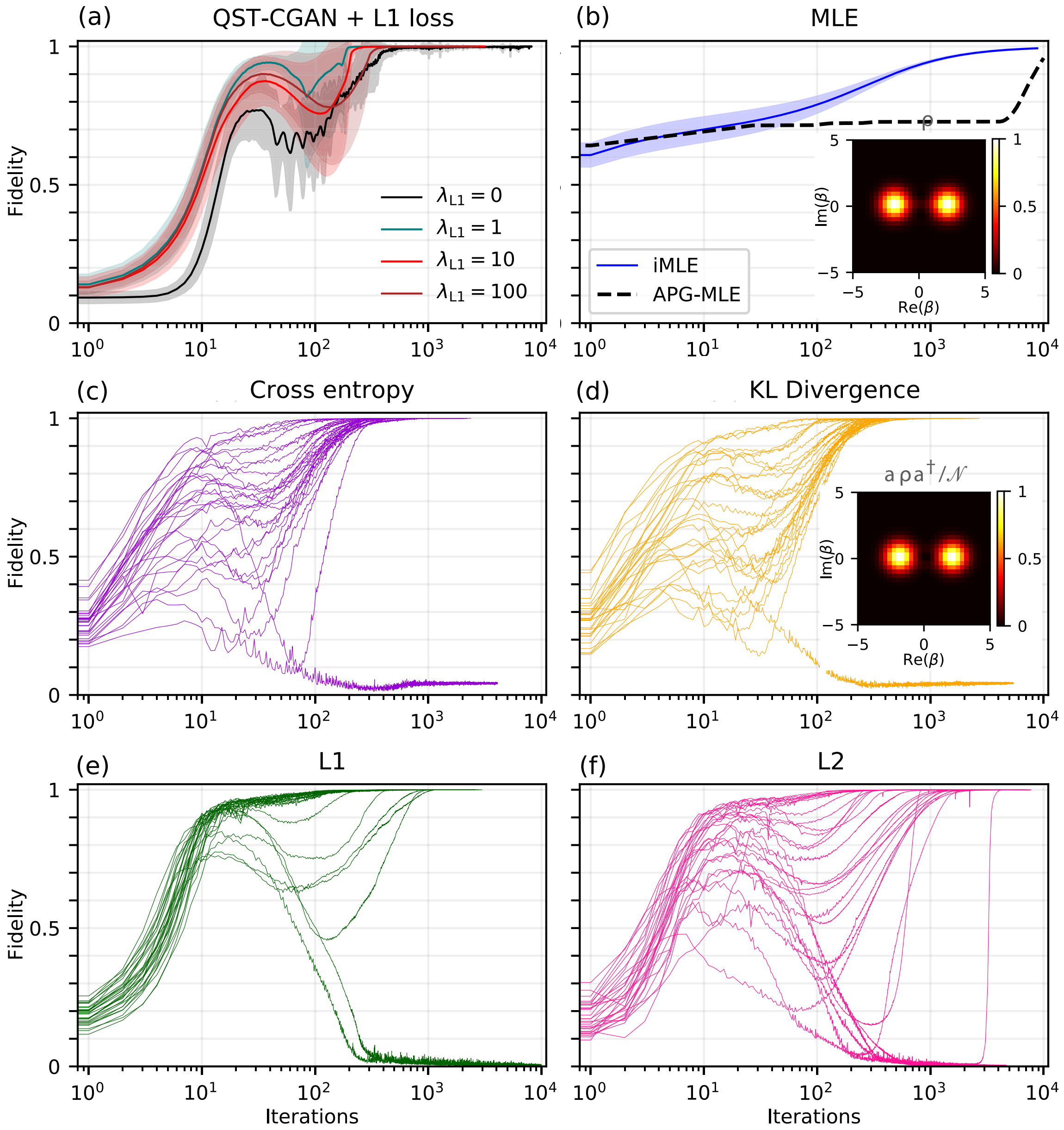

FIG. 12. The effect of the loss function on reconstruction of a `cat`$(\alpha = 2, S = 0, \mu = 0)$ state from Husimi-Q-function data [inset in panel (b)]. All hyperparameters, number of runs, and meanings of solid lines and the shaded regions in the plots are the same as for Fig. 11. (a) Performance of the QST-CGAN with various weights of the L1 loss. (b) Reconstruction fidelities for MLE methods on the same data. [(c)–(f)] Reconstruction performance with standard loss functions. The inset in (d) shows the Husimi Q function of a `cat` state orthogonal to the one in the inset in (b), which was used to produce the data. The state in (d) is constructed by applying the photon annihilation operator a to the original state in (b).

as we will see in the following subsections, there are other situations, e.g., when the reconstruction is performed in the presence of noise, where these losses give better performance and where a different value for λ_{L1} may be more suitable for the QST-CGAN. The fact that different situations seem to require different loss functions is an important argument in favour of the flexibility of the `Discriminator` loss, which can adapt to the situation, allowing the QST-CGAN to perform well in a more general setting.

2. Reconstruction in the presence of additive Gaussian noise

We now compare how different loss functions affect the neural-network performance in the presence of additive Gaussian noise (see Sec. III C 5). In Fig. 13, we show representative results of reconstructing a `binomial` state from Husimi-Q-function data where Gaussian noise has been added. For each β in the Husimi Q function of the state, we add a random value sampled from a zero-mean Gaussian with standard deviation $\sigma = 0.05$. Before adding the noise, we rescale the data to the

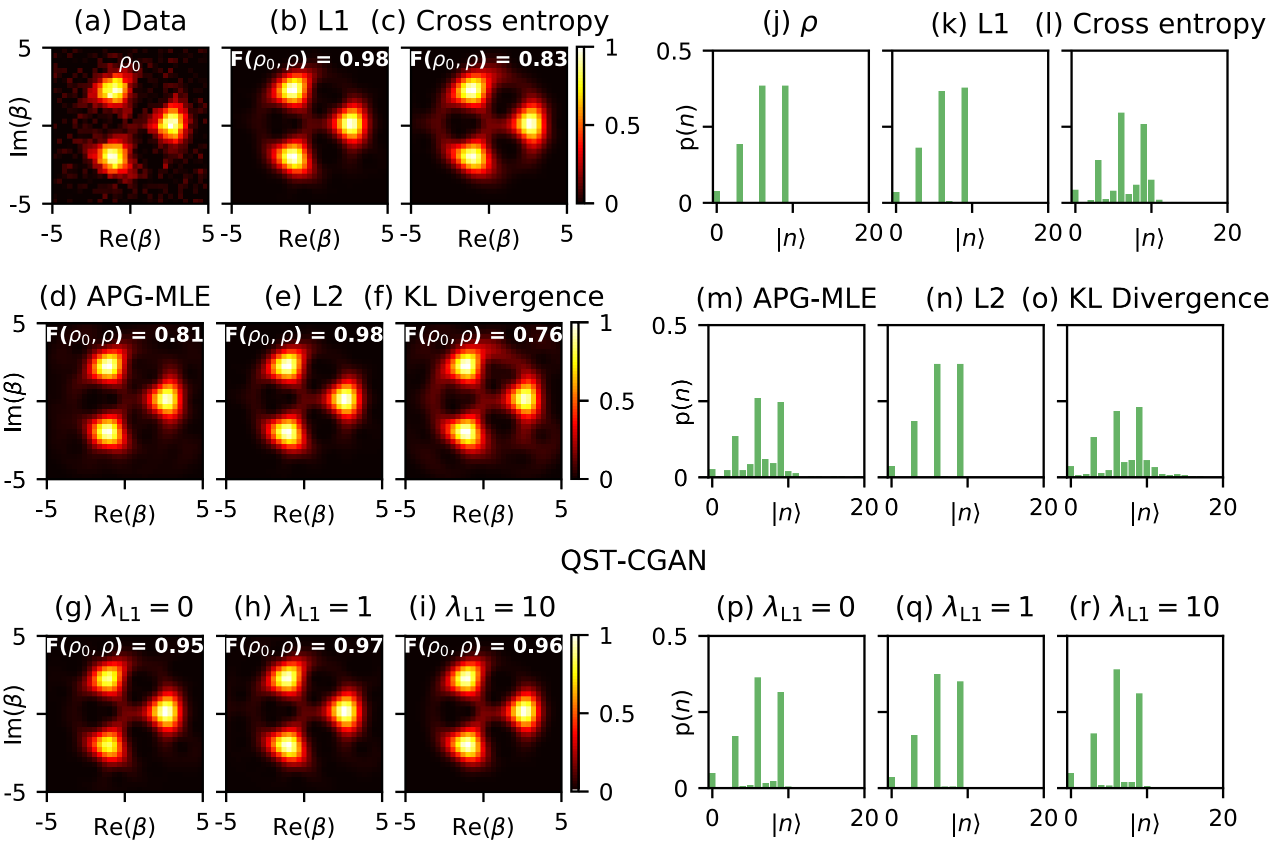

FIG. 13. Reconstruction of a binomial($S = 2, N = 4, \mu = 0$) state in the presence of additive Gaussian noise. (a) The Husimi Q function of the state after addition of Gaussian noise at each β. The random noise is drawn from a standard normal distribution with $\sigma = 0.05$ and added after the data has been normalized to the range [0, 1]. [(b)–(f)] Reconstructed Husimi Q functions, without noise added by the *GaussianNoise* layer, using standard loss functions for the Generator: L1, cross entropy, L2, and KL divergence, respectively. (d) Reconstructed Husimi Q function using APG-MLE. [(g)–(i)] Reconstructed Husimi Q functions using our QST-CGAN with three different weights of the L1 loss set by λ_{L1}. (j) Photon-number occupation probabilities for the data without noise added. [(k)–(r)] Photon-number occupation probabilities extracted from the reconstructed density matrices corresponding to the Husimi Q functions in (b), (c), (d), (e), (f), (g), (h), and (i), respectively. In all reconstructions using neural networks, the hyperparameters for learning were kept the same. For each method, including APG-MLE, the calculations were stopped after 10 000 iterations.

range [0, 1] by dividing it with the maximum value of the Husimi Q function.

To enable the neural network to learn the state underlying the noisy data, we augment the Generator output with the known noise by introducing a *GaussianNoise* layer. This layer applies the same type of noise to the reconstructed data by sampling from a Gaussian with $\sigma = 0.05$ at each gradient-descent step of the Adam optimization. Note that at each step the noise added has the same variance, but differs due to the random sampling. The application of this method in practice requires knowing the type of noise, and its variance, in the experimental setup, but we believe this is feasible to extract.

It might also be possible to simply let the neural network learn the noise. However, applying backpropagation techniques for training requires calculation of gradients with respect to the parameters. The automatic differentiation methods usually employed for gradient calculation in neural networks are not straightforward to apply when such stochastic noise layers are present in the networks. Nevertheless, methods such as the reparametrization trick [11] can still make it possible to learn the noise. However, we have not explored this possibility further in this paper.

Looking at the reconstructed Husimi Q functions in Figs. 13(b)–13(i), it appears that the Generator with L1 or L2 loss and the QST-CGAN with $\lambda_{L1} = \{0, 1, 10\}$ outperform the Generator with cross entropy or KL divergence loss, and clearly outperform the APG-MLE implementation. However, a small difference in the appearance of the Husimi Q function does not necessarily mean that two states are similar (compare the orthogonal states depicted in the insets of Fig. 12). We therefore plot, in Figs. 13(j)–13(r), the photon-number occupation probabilities corresponding to the noiseless data and the reconstructions in Figs. 13(b)–13(i). The noiseless data has nonzero probabilities for 0, 3, 6, and 9 photons. This is only reproduced well by the Generator with L1 or L2 loss and the QST-CGAN with $\lambda_{L1} = 1$. The QST-CGAN with $\lambda_{L1} = 1$ also reproduces the equal probabilities of 6 and 9 photons in the data better than the QST-CGAN with $\lambda_{L1} = \{0, 10\}$.

To further investigate how different loss functions affect the neural-network performance in the presence of additive

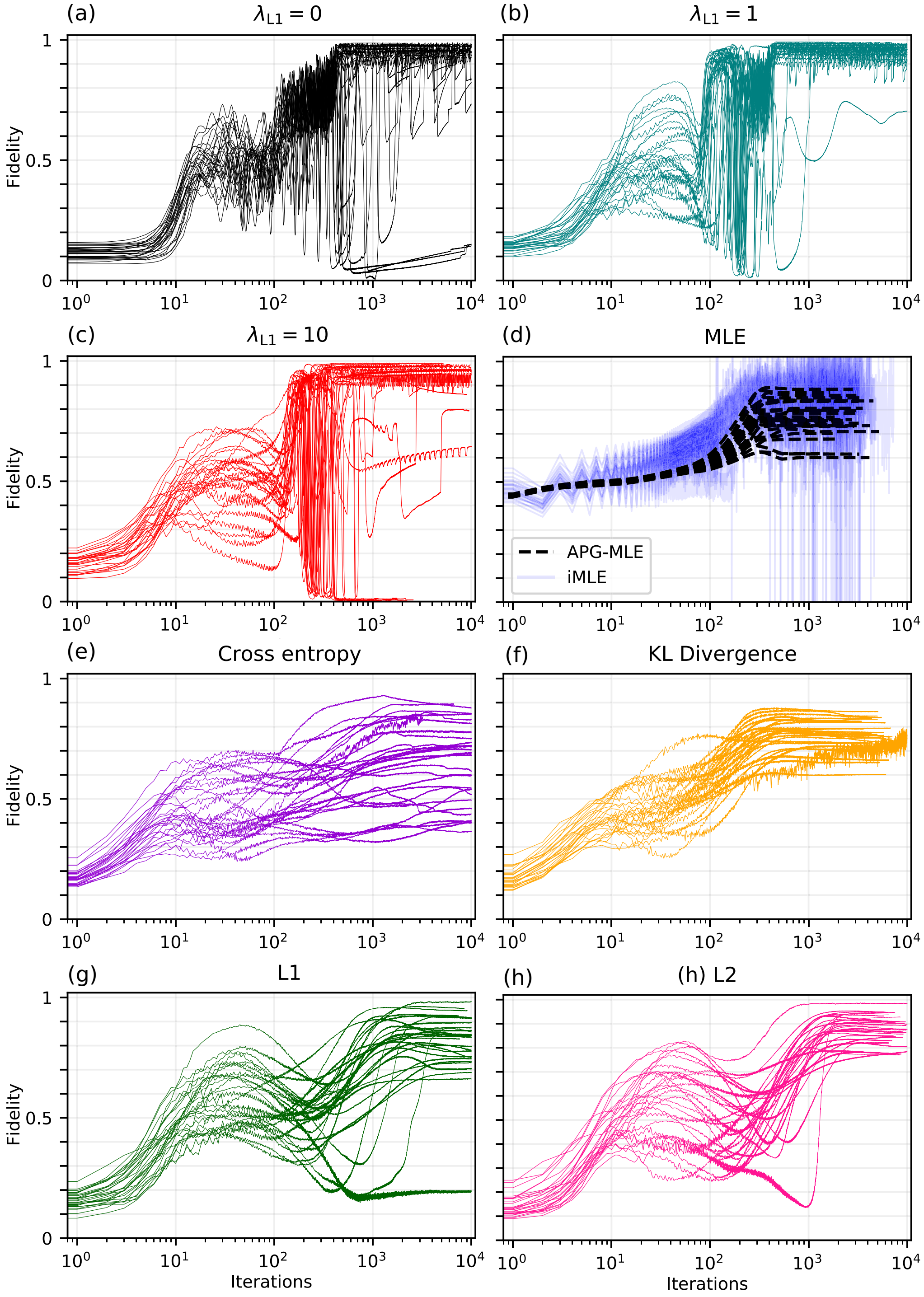

FIG. 14. The effect of the loss function on reconstruction of the `binomial` state in Fig. 13 with 30 different realizations of the additive Gaussian noise. We show all runs for each loss function. The injected noise has the same variance $\sigma = 0.05$ in each run, but is sampled anew for each run. [(a)–(c)] Reconstruction fidelity for the QST-CGAN with various weights of the L1 loss. (d) Reconstruction fidelity for MLE methods. [(e)–(h)] Reconstruction fidelity obtained by training the `Generator` using standard loss functions: cross entropy, KL divergence, L1, and L2. In all the neural-network-based reconstructions, we use the same hyperparameters for training in order to have a fair comparison.

Gaussian noise, we plot, in Fig. 14, how the reconstruction fidelity develops, for all reconstruction methods, as a function of the number of iterations for 30 different realizations of the noise in the data (the same `binomial` state as in Fig. 13). The average reconstruction fidelities and standard deviations are summarized in Table V, where we exclude the reconstructions when the state converges to an orthogonal state [see, e.g., Fig. 14(c) for $\lambda_{L1} = 10$]. Note that we have not tuned the hyperparameters for training, which can lead to improvements for all methods.

TABLE V. Mean and standard deviations for fidelities F reached for reconstruction of the `binomial` state in Fig. 13 in the presence of additive Gaussian noise ($\sigma = 0.05$). We consider 30 different sets of noise for each type of loss function. The full trajectories for the fidelities, as each method iteratively updates the estimate of the density matrix, are shown in Fig. 14.

Loss	Mean F	Std (F)
QST-CGAN ($\lambda_{L1} = 0$)	0.85	0.24
QST-CGAN ($\lambda_{L1} = 1$)	0.95	0.05
QST-CGAN ($\lambda_{L1} = 10$)	0.93	0.07
Cross entropy	0.65	0.15
KL-Divergence	0.76	0.06
L1	0.81	0.14
L2	0.87	0.05
APG-MLE	0.76	0.07

However, it is clear from Fig. 14 that the average reconstruction fidelities alone do not give a complete picture of the performance. Looking at the best runs for each method, we see that the QST-CGANs, the iMLE, and the `Generator` with L1 or L2 loss all are able to reach fidelities very close to 1, while the `Generator` with cross entropy or KL-divergence loss never ends up above fidelity 0.9. Looking at the spread of results, we see that the iMLE is very unstable, while the QST-CGAN and the `Generator` with L1 loss have a small number of runs ending up at a very low fidelity. The `Generator` with L2 loss appears to produce high-fidelity reconstructions with the greatest consistency. We note that the remaining instability in the QST-CGAN performance likely could be remedied by regularization, e.g., higher L1 or L2 penalties on the weights of the neural network. However, to make comparisons fair across the paper, we opted for not changing any such hyperparameters.

Finally, we can compare how fast the different methods reach high reconstruction fidelity. Here, we see the same trend in Fig. 14 as in Figs. 11 and 12: the QST-CGAN is faster than the `Generator` trained with standard loss functions, and the fastest QST-CGAN is the one with $\lambda_{L1} = 1$.

To summarize the results in Figs. 13 and 14, the QST-CGAN and the `Generator` trained with L1 or L2 loss outshone the other methods when reconstructing a state in the presence of additive Gaussian noise. For the `Generator` trained with standard loss functions, the results were thus different from the noiseless case in Sec. V B 1, when training with cross entropy or KL-divergence loss gave better results. When comparing the best QST-CGAN, the one trained with $\lambda_{L1} = 1$, to the best `Generator` trained with standard loss functions, the one trained with L2 loss, we find similar performance in terms of reconstruction fidelities, but the QST-CGAN is faster to reach a good reconstruction.

The errors in reconstruction using the cross entropy and KL-divergence loss are expected, because these loss functions, similar to iMLE, assume the incorrect likelihood [Eq. (3)] for the data, which does not include the Gaussian error model of Eq. (51). The QST-CGAN reconstruction performs better than these methods since it has the flexibility to learn an appropriate loss function. We only provide the overall objective of making the reconstructed statistics similar to the data.

The reason for the good performance using the L2 loss can be further understood from arguments presented in a recent work on image denoising—*Noise2Noise* [180]. There, the authors note that the expectation value for the loss function when using L2 loss remains unchanged if the targets are replaced by random numbers distributed such that their expectation value matches the target. The crucial insight is that "the training targets of a neural network can be corrupted with zero-mean noise without changing what the network learns". Similarly, the L1 loss recovers the median values of the targets and is thus not affected by outliers.

As a final remark, we note that an important factor to consider is that fidelity may not be the best metric to compare results, since sometimes completely random quantum states can have a high fidelity with a desired state [181]. Moreover, for continuous-variable quantum states such as the optical states considered here, several states with high overlap can have very different characteristics [181,182].

3. *Reconstruction in the presence of Gaussian convolution noise*

The additive Gaussian noise discussed in the preceding section models statistical errors due to a low signal-to-noise ratio. Such noise can be reduced (averaged out) by taking more measurements. However, in many cases we have other types of noise that can corrupt the data. Removing such noise in the context of image processing constitutes an inverse problem that is often difficult or ill-posed and requires regularization techniques [183].

We now show how to deal with one such type of noise: Gaussian convolution noise (see Sec. III C 2) due to a linear amplification channel [154]. In such a setup, a background noise, which usually is easy to estimate, corrupts our signal via a convolution operation. Similar to Sec. V B 2, we consider this known background noise as an input to the `Generator` and augment the `Generator` with a *GaussianConv* layer such that the `Generator` output is convoluted with the noise in the same way as the data. This noise layer is not learned, but fixed to the predetermined background noise. The addition of the noise layer forces the `Generator` network to learn a density matrix ρ' that can generate similar statistics as the data after convolution with the background noise.

In Fig. 15(a), we show the results of reconstructing a single-photon Fock state from the Husimi-Q-function data after convolution with a background noise arising due to the amplification channel being in a thermal state. In the simulations considered in preceding sections, we used a 32×32 grid of measurements. However, this coarse grid led to numerical aberrations in the convolution operation. For the present section, we therefore considered an 81×81 grid instead. The results show that the underlying single-photon state is reconstructed perfectly with unit fidelity by a QST-CGAN with $\lambda_{L1} = 10$ despite the presence of significant noise.

However, since the inverse problem can be ill-posed, it is also possible to obtain a result that reconstructs the data well without getting the underlying state right. In Fig. 15(b), we show the one such reconstruction using a `binomial` state in the presence of the same Gaussian convolution noise as in

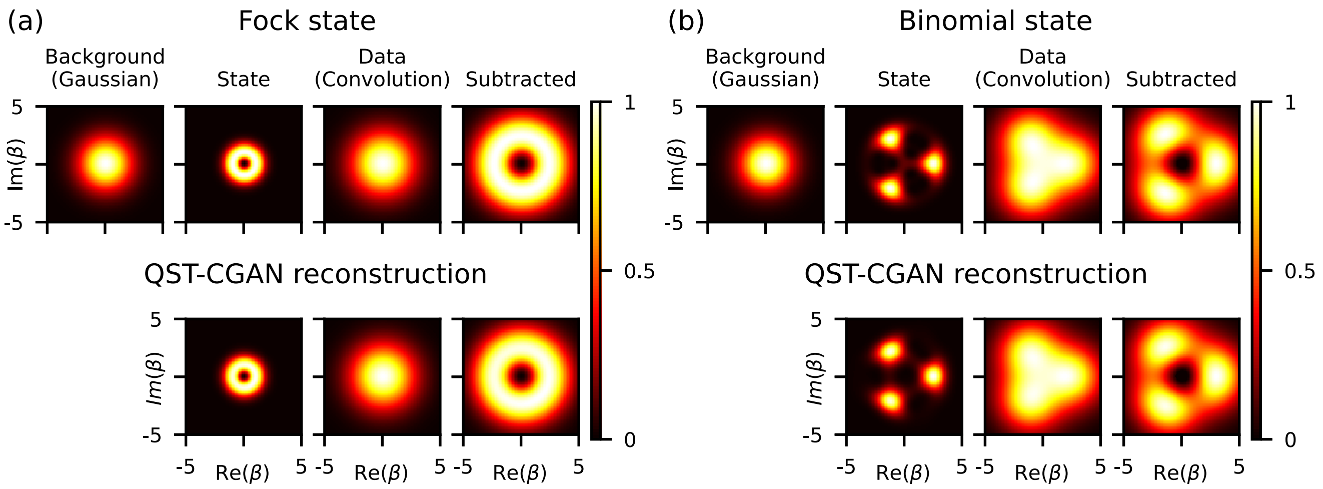

FIG. 15. Reconstruction in the presence of Gaussian convolution noise. We assume that the noisy amplification channel has a thermal noise with $n_{\text{th}} = 5$ photons (see Sec. III C 2). This noise is convoluted with the data from the Husimi Q function in (a) a fock($n = 1$) state and (b) a binomial($S = 2, N = 4$) state. We had to use an 81 × 81 grid for the data since we noticed that the convolution operation leads to effects such as loss of radial symmetry in (a) when using a grid of 32 × 32. The image obtained by subtracting the background from the convoluted data shows the symmetries of the state. We show the reconstruction of the underlying states in by the QST-CGAN ($\lambda_{\text{L1}} = 10$) where the Generator output is convoluted with the same background noise. We recover the underlying state from the *DensityMatrix* layer of the Generator. Note that even though the convoluted outputs and subtracted counts match the data well in both cases, the fidelity between the underlying reconstructed state itself and the true underlying state is 1 for (a), but only 0.45 in (b).

Fig. 15(a). The reconstructed density matrix gives rise to measurement statistics that match the measured (simulated) data exactly. However, the state itself is incorrect with a fidelity of just 0.45. We note that the symmetries of the state are captured in the reconstruction, but due to the convolution operation, the information of the exact state is lost and the inversion is not unique.

4. Reconstruction of mixed states

So far, we have only considered pure quantum states, where the density matrix has rank $r = 1$. However, in real experiments, we will almost always be dealing with mixed states. Such states may be harder for a neural network to handle, since they do not admit as compact a representation as a pure state, which can be written $\rho = |\psi\rangle\langle\psi|$. In this section, we therefore discuss how the QST-CGAN method performs for mixed states with rank $r > 1$.

In a realistic experiment, it is reasonable to assume that the mixed state will have a dominant part, e.g., a target state, which decoheres due to photon loss. Figure 16 shows results for the QST-CGAN reconstruction on a mixed state with a cat state (the same state as in Fig. 12) being the dominant component. The figure shows that the QST-CGAN can reconstruct such a mixed state easily for ranks up to $r = 4$ with close to unit fidelity ($\geqslant$.99). For ranks 1 and 2, the QST-CGAN method converges almost two orders of magnitude faster than iMLE. As the rank increases, both QST-CGAN and iMLE show a slower convergence. Although we did not run the iMLE for enough iterations to be certain, the increase in the number of iterations required for convergence appears to be greater for iMLE than for the QST-CGAN. Similar trends are seen for the APG-MLE method.

To explore further for higher ranks, we consider in Fig. 17 the reconstruction of a full-ranked ($r = 32$) thermal state with a mean photon number $n_{\text{th}} = 1$. Here, the iMLE method converges very fast, almost instantaneously, while the QST-CGAN requires several hundred iterations. Although both methods reconstruct the state with a high fidelity $\geqslant 0.99$, the photon-number populations of the reconstructed state do not exactly match the expected super-Poissonian distribution for thermal states for the higher photon numbers (the tail of the distribution), neither for iMLE nor for QST-CGAN. The Husimi Q function of the reconstructed states match well in Figs. 17(c) and 17(d), but the Wigner functions for the iMLE and QST-CGAN methods do not match the smooth Wigner function for the thermal state in Figs. 17(f) and 17(g). However, changing the input data for reconstruction to the Wigner function (displaced parity measurements; see Sec. III B) can lead to a better reconstruction as we discuss in Fig. 18. For the QST-CGAN, this is as simple as changing the input measurement operators to the Generator from projections on the coherent state (Husimi Q) to displaced parity operators (Wigner).

Having explored reconstruction performance for low- and high-rank density matrices, we next turn to intermediate rank. In Fig. 18, we consider a random density matrix of rank 11 and show its reconstruction from both types of input data (Husimi Q and Wigner). Here, the difference between using the two types of data becomes clear. With the QST-CGAN method, we obtain a reconstruction fidelity of 0.8 using Husimi Q function and ~0.99 when using the Wigner function. In Fig. 18(h), we can clearly see that details of the Wigner function for the true state in Fig. 18(e) are not captured when we take Husimi Q as our data, even though the reconstructed Husimi Q function in Fig. 18(g) matches perfectly with the data in Fig. 18(d). However, there are big differences in how different ML models

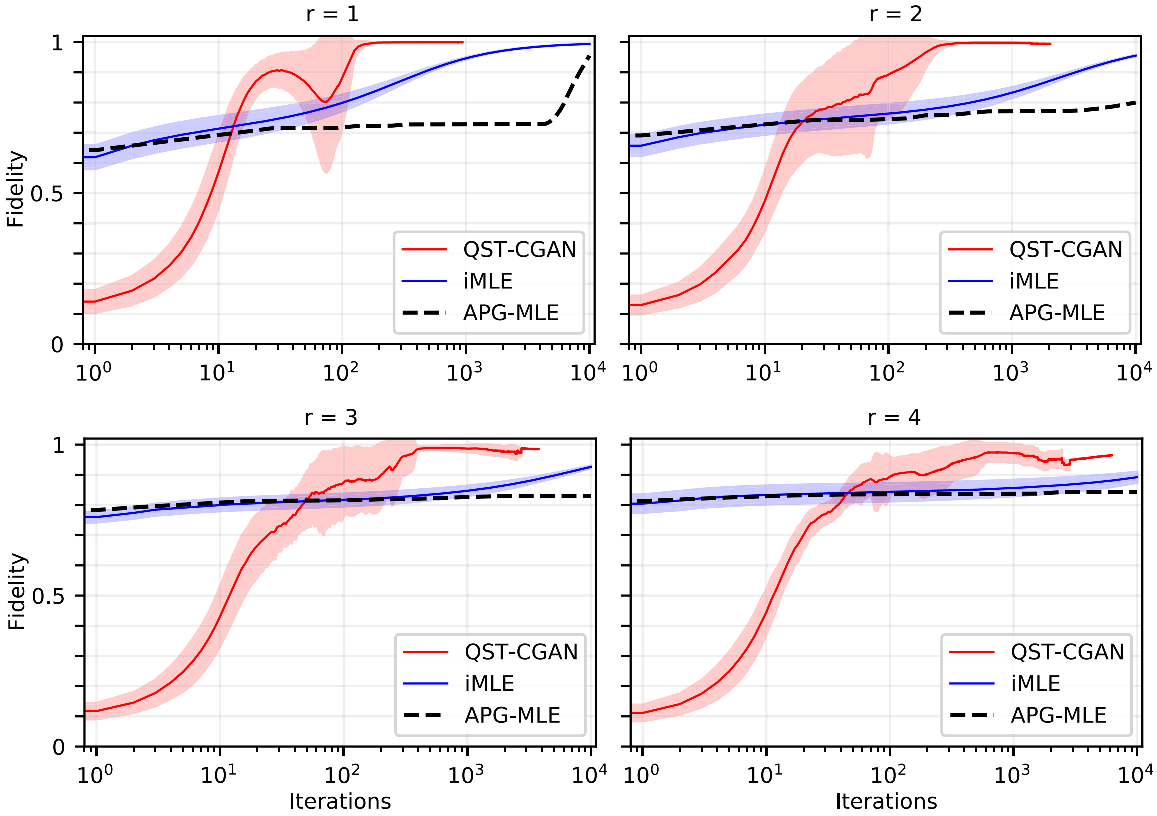

FIG. 16. Reconstruction of a mixture $\rho = 0.8\texttt{cat} + [0.2/(r-1)]\sum_{n=0}^{r-2}\texttt{fock}(n)$ of $\texttt{cat}(\alpha = 2, S = 0, \mu = 0)$ (the state used in Fig. 12) and $\texttt{fock}(n)$ states, where $r \geqslant 2$ denotes the rank. For $r = 1$, the state is just $\texttt{cat}(\alpha = 2, S = 0, \mu = 0)$. The input data is the Husimi Q function of ρ measured in a 32×32 grid. The solid lines show the mean and the shaded regions show one standard deviation from the mean for the QST-CGAN (red) and iMLE (blue) over 15 reconstructions for each of ranks $r = 1, 2, 3, 4$. The dashed black lines show the fidelities for states given by APG-MLE using the default "bootstrap" initialization. The APG-MLE method does not have any randomization and therefore does not have a standard deviation from the mean. In each repetition, we use the same data, but start from a different random initial state for iMLE and random weights for the QST-CGAN. We choose the weight $\lambda_{L1} = 1$ for the QST-CGAN and keep all other training hyperparameters the same as used for previous results and described in Sec. IV B 3. The QST-CGAN runs are stopped when they have converged on a reconstructed state.

perform, within the limitations of the data; there is no silver bullet. In this particular case, we can argue that the Husimi Q represents a convolution over the Wigner function [184] and therefore using it for reconstruction could be ill-posed [183].

5. Data reduction

In Ref. [99], we show that for a particular pure cat state, the QST-CGAN method requires much fewer data points, ∼100, for reconstruction than the iMLE method, which requires more than 10,000. In Ref. [185], it is argued how homodyne tomography can be IC when the number of independent quadratures measured is equal to the dimension N of the density matrix. More specifically, IC requires N quadratures to be measured, each of which can be discretized into $2N - 1$ bins. Therefore, a full-rank density matrix of dimension N requires $O(N^2)$ measurements in the phase space for IC. However, for low-rank states the data requirements can scale as $O(rN)$ [186]. These arguments suggest that for states described with density matrices of dimension $N = 32$, thousands of measurements are required for IC when considering full-rank states. However, for low-rank or rank-1 pure states, the number of data points for IC could be much smaller ($\propto 32r$). Note that the IC limit does not necessarily specify which measurements are important and give maximum information; the limit also depends on the density-matrix dimensions, which we can set to have different cut-offs for optical quantum states. Our QST-CGAN approach consistently required only ∼100 measurements for reconstruction of pure ($r = 1$) states using a random set of measurement settings.

In this section, we benchmark the QST-CGAN performance further by testing how much data it needs to reconstruct states of higher rank. In general, a density matrix of size $N \times N$ with full rank $r = N$ is specified by $N^2 - 1$ real numbers. The number of parameters that needs to be determined during reconstruction is significantly reduced if the state is pure or if we have some prior information about the state [185]. For example, if we know that the state that we are reconstructing is a thermal state, then even if the density matrix is full rank ($r = N$), we only need to estimate a single parameter, the mean photon number n_{th}, to reconstruct the state. Such priors can thus make it easy to reconstruct the state with data from only a few measurements. Similarly, the analyticity of the Husimi Q function makes it possible to

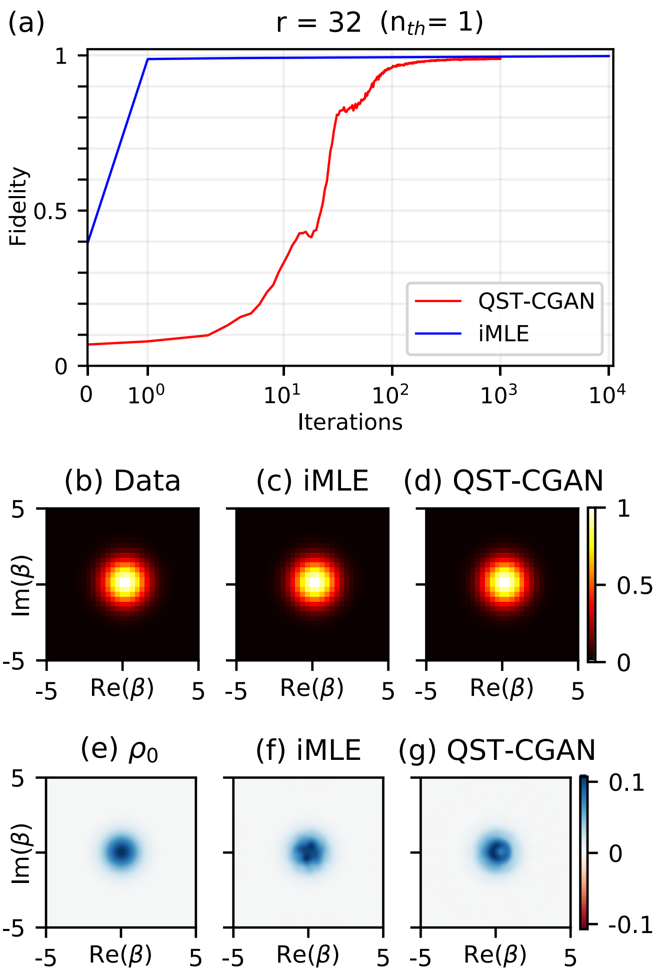

FIG. 17. Reconstruction of a thermal state with mean photon number $n_{\mathrm{th}} = 1$. The QST-CGAN method uses the weight $\lambda_{\mathrm{L1}} = 1$; all other training hyperparameters are kept the same as for previous results and described in Sec. IV B 3. (a) The fidelity of the reconstructed state as a function of the number of iterations for iMLE (blue) and QST-CGAN (red). [(b)–(d)] Reconstructed data compared to the data used for obtaining the underlying density matrix. We use the Husimi Q function measured in a 32×32 grid as the input. [(e)–(g)] Wigner function of the underlying thermal state compared to the Wigner functions obtained from the reconstructions. The reconstructed Wigner functions do not match the smooth nature of the Wigner function obtained from the underlying state.

apply other reconstruction methods, e.g., Lagrange interpolation [187], which sample from the Husimi Q function and obtain the exact density matrix without requiring any iterations. Similarly, Ref. [188] reconstructed a state description in the Fock basis using Wigner-function-overlap measurements and semidefinite programming, requiring less data.

In Fig. 19, we show how reconstructing a rank-4 state from a random selection of 256 points of the Husimi Q function fails for iMLE, which gets stuck. The convergence of iMLE is not guaranteed since there could be steps, which strictly reduce the likelihood, producing cycles where the method does not improve its estimate of the density matrix [123]. The QST-CGAN, on the other hand, reconstructs the state almost perfectly, as shown in Fig. 19(d). However, the QST-CGAN requires more than 1000 iterations to converge, which is more than what was needed when it reconstructed the same state using all data [see Fig. 16(d)].

In Fig. 20, we show the reconstruction fidelity with QST-CGAN and iMLE for mixed states of rank $r = 2, 3, 4$ as the number of measurements (data points; values of the Husimi Q function at different β) is reduced. We choose the β values at random inside the circle $|\beta| = 5$. We note that there could be better ways to choose points to sample the Husimi Q function, e.g., the so-called Padua points discussed in Ref. [187].

The QST-CGAN clearly outperforms the iMLE in terms of the amount of data needed for reconstruction in Fig. 20. The QST-CGAN reaches fidelity close to unity with somewhat less than 100 data points, around 100 data points, and a little more than 100 data points, for states of rank 2, 3, and 4, respectively.

This slow growth in the number of data points needed appears consistent with previous results showing that reconstruction of low-rank states in the best case can be done with $\propto rN$ data points [129]. Meanwhile, the iMLE cannot reconstruct ρ even when given a large number of data points. However, this is not just due to the lack of information, but also due to the random selection of the data points themselves.

Although the results here do not establish any bounds on the minimum number of data points necessary for the QST-CGAN method to reconstruct a quantum state, they show that the QST-CGAN approach can perform much better than conventional reconstruction methods when data is scarce. An intuitive explanation of this is that since neural networks are universal function approximators, the `Generator` network might learn to find an approximation for the state in terms of a few parameters, e.g., the mean photon number for a thermal state, and estimate it better. However, the theoretical underpinnings of the QST-CGAN performance for few data points needs to be explored further, which is beyond the scope of this paper.

VI. CONCLUSION

We have shown how deep neural networks can assist in the characterization of quantum states. The states we considered here were optical quantum states, including bosonic error correction codes, but our methods are general and should be applicable also to systems with qubits.

A. Classification

We first showed how a neural network with convolutional layers followed by a dense layer can discriminate and classify several different types of quantum states with near-perfect accuracy. The input to the network was measurement data from phase-space descriptions of the states. The rare few misclassifications could be explained by the existence of parameter ranges where states from different classes are extremely similar and the problem of classification thus becomes ill-defined.

We further demonstrated the robustness of this classification method against two prominent noise sources—additive Gaussian noise and single-photon loss. For the former, the network performance remained almost perfect until the standard deviation of the added noise reached as high values as 20% of the largest input data values. For the latter noise, we showed a

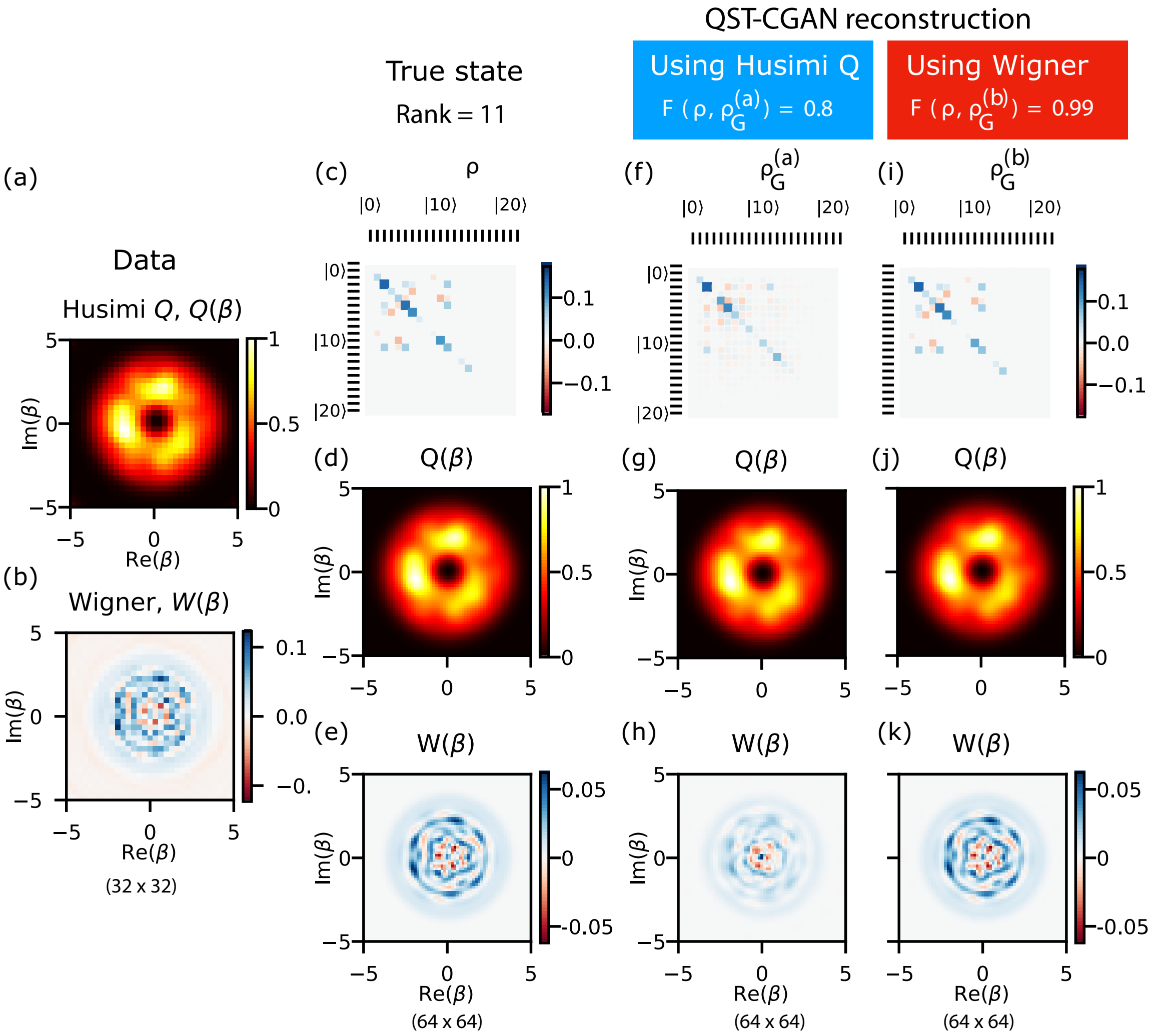

FIG. 18. Reconstruction with a QST-CGAN of a random mixed state of rank 11 from Husimi Q and Wigner functions sampled on a 32×32 grid. [(a),(b)] Data used for the reconstruction: the Husimi Q and Wigner functions. (c) Hinton plot (see Fig. 3) of the underlying density matrix. [(d),(e)] Husimi Q and Wigner functions for a 64×64 grid computed from the underlying state to show finer features not present in the data that is fed to the neural network. [(f)–(h)] Reconstruction results using the Husimi-Q-function data in (a) as input for the QST-CGAN. [(i),(j),(k)] Reconstruction results using the Wigner-function data in (b) as input for the QST-CGAN.

specific example where the network could identify a `cat` state even after it had lost 70% of its initial photons.

By using the Grad-CAM method to extract and visualize which parts of the input phase space that the neural network bases its classification decision on, we proposed a simple adaptive technique for tomography that could significantly reduce the data-collection time for an experiment. Since the neural network learns the characteristic features of the states it is set to classify and can be trained to be robust against simple noise sources in the data, we can deploy it online at the initial stages of an experiment for guided data-collection during tomography.

B. Reconstruction

We next introduced, here and in Ref. [99], a density-matrix-estimation technique using a combination of ideas from VAEs and GANs: the QST-CGAN method. This method uses custom neural network layers that convert the output of any standard neural network into valid density matrices using the Cholesky decomposition. Therefore, we can convert any neural-network architecture into a variational map from input data to a density matrix. Following this scheme, we constructed a custom `Generator` network that maps input data to a density matrix and computes statistics for measurement operators.

By training the `Generator` network using gradient-based methods, we showed that the density matrix for the underlying state can be easily reconstructed. Instead of using a standard straight-forward loss function that requires an assumption on the likelihood for the data, we used a second `Discriminator` neural network to help train the `Generator`. Our choice of this adversarial training framework was motivated by an analysis of how standard loss functions, e.g., L2 or KL-divergence loss, perform for different states and noise in the data. We found that some of these standard loss functions resulted in good performance in the absence of noise, while other loss functions gave better performance in the presence of certain

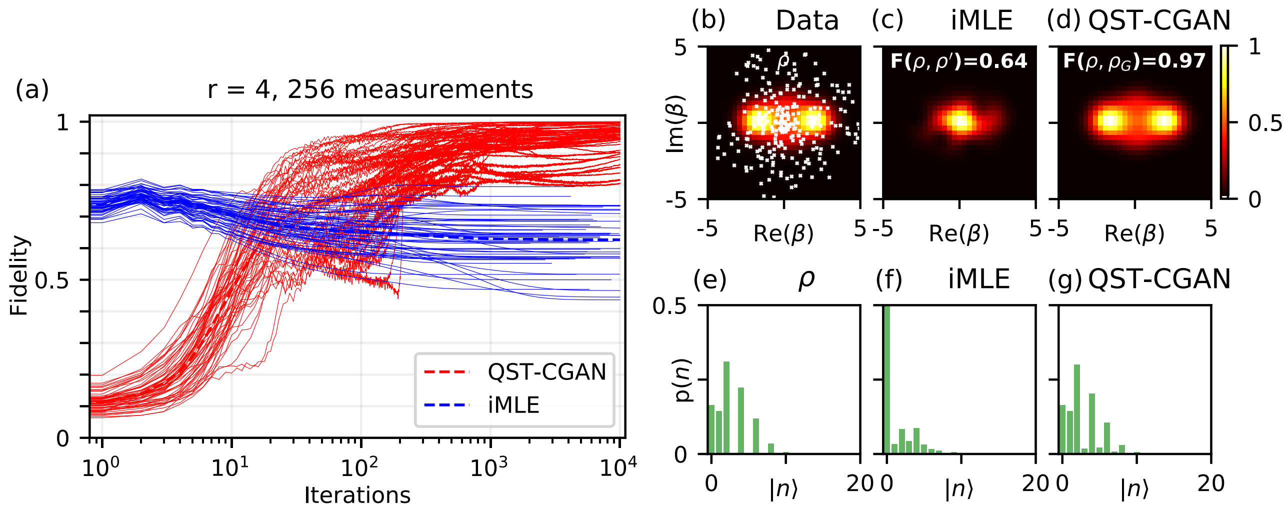

FIG. 19. Reconstruction of a rank-4 mixture of $\texttt{cat}(\alpha = 2, S = 0, \mu = 0)$ and $\texttt{fock}(n)$ states (the mixture is constructed using the same formula as in Fig. 16) from input data consisting of 256 points of the Husimi Q function (instead of the full $32 \times 32 = 1024$ points used in preceding sections). (a) Reconstruction fidelity of the QST-CGAN (with $\lambda_{\mathrm{L1}} = 1$; red) and iMLE (blue) where we plot all trajectories for 36 different reconstructions and show the mean with dashed lines. In each reconstruction, we randomly selected a set of β values from the phase space and reconstruct the state where the QST-CGAN weights and the initial state for iMLE are reinitialized randomly. [(b)–(d)] Comparison between the data and the reconstructed Husimi Q function given by iMLE and QST-CGAN for one selection of input data points (white points in the left panel). [(e)–(g)] Photon-number occupation probabilities from the (reconstructed) density matrices in (b).

types of noise, but none of the loss functions led to a consistently good performance in a general setting. However, we showed that the QST-CGAN method is flexible and can easily adapt to different noise, states, or measurement settings. We ascribe this flexibility to the ability of the `Discriminator` to learn a loss function suited to the situation at hand.

We showed that the QST-CGAN-based reconstruction can be up to two orders of magnitude faster than MLE methods, counted in the number of iterations required for reconstruction. Although the actual time for each iteration in the QST-CGAN can depend on the design of the neural networks, this presents a significant advantage for data postprocessing during tomography. We also note that the neural network based method seems to be performing nontrivial operations during reconstruction, e.g., applying a quantum operation to almost instantaneously jump from an orthogonal state to the correct state. This suggests that the neural networks learn to represent the state in a way that is well suited for the problem. Alternatively, the use of the Adam optimization might explain how the neural network based reconstruction is so fast, in a similar way as accelerated gradient-based methods [119].

Having first benchmarked the reconstruction of pure states with no noise, we next considered how the QST-CGAN method can be augmented further to deal with noise in the data. We leveraged the flexibility of having a loss function that combines the `Discriminator` loss with a simple L1 loss, since our objective is simply to make the generated data look like the training data. For the case of additive Gaussian noise of up to 5% of the maximum signal value, our QST-CGAN method performs denoising and reconstruction much better than MLE methods without needing any change in the architecture or loss function. Gaussian convolution noise corresponding to having a thermal state with mean photon number $n_{th} = 5$ in a linear detection scheme was also tackled quite easily. The QST-CGAN only required the expected background noise as input, which was added as special noise layers to the `Generator` network.

Lastly, we showed that the QST-CGAN method clearly outperforms MLE methods also when reconstructing mixed states. The QST-CGAN proved superior not only in terms of how few iterations it needed to reach high reconstruction fidelity, but also in terms of how little input data it required to reconstruct the state well. For a `cat` state, the QST-CGAN required almost two orders of magnitude fewer data points than iMLE (as well as an RBM-based reconstruction shown in Ref. [84]) to achieve high reconstruction fidelity. It has been demonstrated that the iMLE method can become stuck in cycles for some choices of input data, but our QST-CGAN method works well even with random sets of measurements generating the input data for the examples considered.

In conclusion, by connecting ideas of generative and discriminative modeling to quantum state classification and reconstruction, we have attempted to bridge the gap between deep neural networks and quantum information and computing. We have shown how some of the latest ideas from deep learning can be quite easily adapted and applied to quantum-information tasks with just a few tweaks to incorporate the rules of quantum physics. This opens up a wealth of possible applications, as we discuss further below and in Ref. [99].

VII. OUTLOOK

Our paper suggests several practical possibilities in data analysis of quantum experiments. At the same time, it leads to questions regarding the limits of using neural networks for quantum state characterization. It is expected that image-recognition algorithms will be good at distinguishing different optical quantum states from their phase-space data. However, the benefit of using neural networks is their resilience to known types of noise. If we have to classify a "cat" state, it

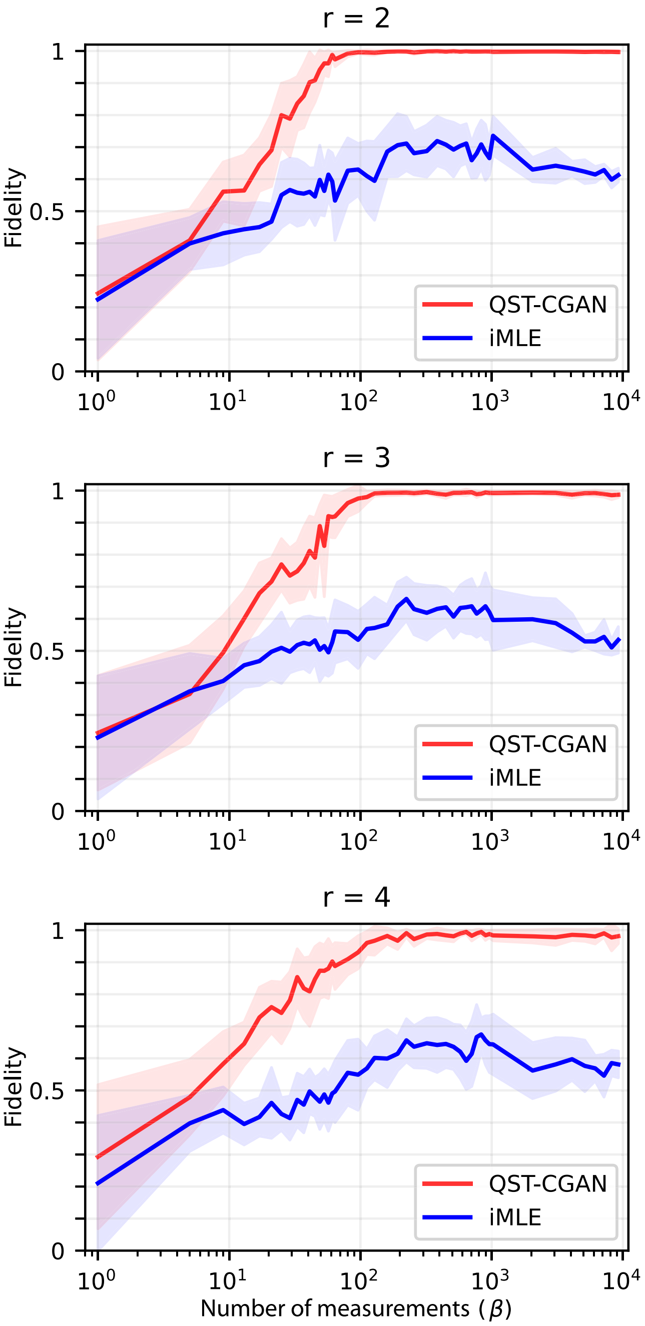

FIG. 20. Reconstruction of a mixture of $\mathtt{cat}(\alpha = 2, S = 0, \mu = 0)$ and $\mathtt{fock}(n)$ states (the mixture is constructed using the same formula as in Fig. 16) with a reduced number of measurements. Reconstruction fidelities for ranks $r = 2, 3,$ and 4 are shown for QST-CGAN (red) and iMLE (blue). The solid lines show the mean and the shaded regions show one standard deviation from the mean. The fidelity shown is the one reached after a certain convergence criterion set by a tolerance value. We choose a tolerance such that if the average fidelity in 100 iterations does not change by 10^{-5} over 5 steps (i.e., 500 iterations) we stop the reconstruction.

is rather easy to see that two lobes and a connecting bridge in the phase space should be a “cat”. But what if, due to noise, the phase-space plots are shifted or rotated? An algorithm that relies on the fixed definition of a cat state will see poor overlap between the definition and the data, and hence cannot recognize the cat even if all the features are present. The neural network method, on the other hand, is implicitly taught the important features that characterize cats and therefore works even in the presence of systematic or random noise.

In the case of reconstruction, we see that the the QST-CGAN method is a very powerful alternative to RBMs. We leverage the universal approximation capabilities of a deep neural network to have a tractable representation of the state by explicitly constructing the full density matrix. Standard loss functions such as fidelity, L1, L2, cross entropy, etc., will always have some shortcomings, since they require an assumption on the underlying likelihood for the data. Instead, with the CGAN framework, we let the loss metric be implicitly defined with the objective of simply making the data look similar to the generated data. However, we have not explored the theoretical underpinnings of using such a learned loss function for reconstruction. This remains to be analysed. Furthermore, we note that automated tuning of the hyperparameters, e.g., learning rates and network architectures, could result in better performance and thus appears relevant to explore.

The future work that leverages these ideas would go in two directions, beyond the suggestions for improvements and tweaks already mentioned in connection with the results. The first is further theoretical analysis of the techniques. For example, it remains to be well understood how the neural network can reconstruct states using much fewer data points than maximum-likelihood methods or possibly perform nontrivial operations during a reconstruction. The second direction is validation with more experimental data and comparison to other standard methods for reconstruction. For example, we have not explored thoroughly how the QST-CGAN method compares with RBM-based approaches for tomography. This would be an interesting comparison since much of the work in QST with machine learning is focused on using RBMs.

Since we ask for the full density matrix during reconstruction, our method cannot directly scale up for very large quantum systems. Even if it is straightforward to replace the density-matrix description with other efficient ansätze, it remains to be answered how to obtain efficient representations such that one does not use the millions of parameters in the deep neural network to estimate a few hundred parameters of the density matrix.

The methods discussed here are ready to be applied to real experiments such that adaptive, online tomography schemes can be designed that can deal with noisy data. The techniques for classification and reconstruction could even be combined: the result of classifying a state with one neural network could be used as a good starting point and parametrization for the training of another network for full quantum state reconstruction. We foresee that our ideas will lead to better techniques for quantum state characterization and bring the power of deep-learning-based tools to the quantum physicist.

ACKNOWLEDGMENTS

We acknowledge useful discussions with Andreas Ekström, Giulia Ferrini, Simone Gasparinetti, Steven Girvin, Marina Kudra, Yong Lu, Florian Marquardt, Fernando Quijandría, and Ingrid Strandberg. We thank Yanming Che for comments on the manuscript. The neural networks were written using Tensorflow [156]. Data generation and visualizations were done with QuTiP [150,151], Scikit-image [189], and Scikit-learn [178]. S.A. and A.F.K. acknowledge support from the Knut and Alice Wallenberg Foundation through the Wallenberg Centre for Quantum Technology (WACQT). C.S.M. acknowledges that the project that gave rise to these results received the support of a fellowship from "la Caixa" Foundation (ID 100010434) and from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant Agreement No. 847648, with fellowship code LCF/BQ/PI20/11760026, and financial support from the Proyecto Sinérgico CAM 2020 Y2020/TCS-6545 (NanoQuCo-CM). F.N. is supported in part by: Nippon Telegraph and Telephone Corporation (NTT) Research, the Japan Science and Technology Agency (JST) the Quantum Leap Flagship Program (Q-LEAP), the Moonshot R&D Grant No. JPMJMS2061, and the Centers of Research Excellence in Science and Technology (CREST) Grant No. JPMJCR1676], the Japan Society for the Promotion of Science (JSPS) [via the Grants-in-Aid for Scientific Research (KAKENHI) Grant No. JP20H00134 and the JSPS-RFBR Grant No. JPJSBP120194828], the Army Research Office (ARO) (Grant No. W911NF-18-1-0358), the Asian Office of Aerospace Research and Development (AOARD) (via Grant No. FA2386-20-1-4069), and the Foundational Questions Institute Fund (FQXi) via Grant No. FQXi-IAF19-06.

[1] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, Rev. Mod. Phys. **91**, 045002 (2019).

[2] J. Shlomi, P. Battaglia, and J.-R. Vlimant, Graph neural networks in particle physics, Mach. Learn.: Sci. Technol. **2**, 021001 (2020).

[3] S. Ravanbakhsh, J. Oliva, S. Fromenteau, L. C. Price, S. Ho, J. Schneider, and B. Poczos, Estimating cosmological parameters from the dark matter distribution, arXiv:1711.02033.

[4] M. A. Aragon-Calvo, Classifying the large-scale structure of the universe with deep neural networks, Mon. Not. R. Astron. Soc. **484**, 5771 (2019).

[5] G. Stein, georgestein/ml-in-cosmology: Machine learning in cosmology (2020).

[6] J. Carrasquilla, Machine learning for quantum matter, Adv. Phys.: X **5**, 1797528 (2020).

[7] J. Schmidt, M. R. G. Marques, S. Botti, and M. A. L. Marques, Recent advances and applications of machine learning in solid-state materials science, npj Comput. Mater. **5**, 83 (2019).

[8] R. Iten, T. Metger, H. Wilming, L. del Rio, and R. Renner, Discovering Physical Concepts with Neural Networks, Phys. Rev. Lett. **124**, 010508 (2020).

[9] R. T. D'Agnolo and A. Wulzer, Learning new physics from a machine, Phys. Rev. D **99**, 015014 (2019).

[10] G. Zhang, Neural networks for classification: A survey, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **30**, 451 (2000).

[11] D. P. Kingma and M. Welling, Auto-encoding variational Bayes, arXiv:1312.6114.

[12] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, in *Advances in Neural Information Processing Systems*, Vol. 3 (Curran Associates, Inc., Montreal, Quebec, Canada, 2014), pp. 2672–2680.

[13] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2016), p. 770.

[14] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2017), p. 1800.

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You only look once: Unified, real-time object detection, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2016), p. 779.

[16] J. Xie, L. Xu, and E. Chen, Image denoising and inpainting with deep neural networks, *Advances in Neural Information Processing Systems* (Curran Associates Inc., Red Hook, NY, USA, 2012), p. 341.

[17] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising, IEEE Trans. Image Process. **26**, 3142 (2017).

[18] C. Tian, L. Fei, W. Zheng, Y. Xu, W. Zuo, and C. W. Lin, Deep learning on image denoising: An overview, arXiv:1912.13171.

[19] L. Xu, J. S. J. Ren, C. Liu, and J. Jia, Deep convolutional neural network for image deconvolution, in *Advances in Neural Information Processing Systems*, Vol. 27, edited by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (Curran Associates, Inc., Red Hook, NY, 2014), pp. 1790–1798.

[20] M. Mirza and S. Osindero, Conditional generative adversarial nets, arXiv:1411.1784.

[21] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, Image-to-image translation with conditional adversarial networks, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2017), pp. 5967–5976.

[22] T. Karras, S. Laine, and T. Aila, A Style-Based Generator Architecture for Generative Adversarial Networks, in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2019), pp. 4396–4405.

[23] Y. Mirsky and W. Lee, The creation and detection of deepfakes: A survey, ACM Comput. Surv. **54**, 1 (2021).

[24] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, Language models are few-shot learners, arXiv:2005.14165.

[25] A. Karpathy, Software 2.0 (2017) https://medium.com/@karpathy/software-2-0-a64152b37c35.

[26] C. Payne, MuseNet (2019) https://openai.com/blog/musenet/.

[27] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, Synthesizing Obama: Learning lip sync from audio, ACM Trans Graph. **36**, 1 (2017).

[28] S. W. Kim, Y. Zhou, J. Philion, A. Torralba, and S. Fidler, Learning to simulate dynamic environments with GameGAN, arXiv:2005.12126.

[29] J. Gottschlich, A. Solar-Lezama, N. Tatbul, M. Carbin, M. Rinard, R. Barzilay, S. Amarasinghe, J. B. Tenenbaum, and T. Mattson, The three pillars of machine programming, in *Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages - MAPL 2018* (ACM Press, New York, 2018), pp. 69–80.

[30] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. R. Nelson, A. Bridgland *et al.*, Improved protein structure prediction using potentials from deep learning, Nature (London) **577**, 706 (2020).

[31] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko *et al.*, Highly accurate protein structure prediction with AlphaFold, Nature **596**, 583 (2021).

[32] R. P. Feynman, Simulating physics with computers, Int. J. Theor. Phys. **21**, 467 (1982).

[33] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).

[34] A. Montanaro, Quantum algorithms: An overview, npj Quantum Inf. **2**, 15023 (2016).

[35] G. Wendin, Quantum information processing with superconducting circuits: A review, Rep. Prog. Phys. **80**, 106001 (2017).

[36] J. Preskill, Quantum Computing in the NISQ era and beyond, Quantum **2**, 79 (2018).

[37] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell *et al.*, Quantum supremacy using a programmable superconducting processor, Nature (London) **574**, 505 (2019).

[38] B. S. Rem, N. Käming, M. Tarnowski, L. Asteria, N. Fläschner, C. Becker, K. Sengstock, and C. Weitenberg, Identifying quantum phase transitions using artificial neural networks on experimental data, Nat. Phys. **15**, 917 (2019).

[39] G. Sentis, M. Guta, and G. Adesso, Quantum learning of coherent states, EPJ Quantum Technol. **2**, 17 (2015).

[40] V. Gebhart and M. Bohmann, Neural-network approach for identifying nonclassicality from click-counting data, Phys. Rev. Research **2**, 023150 (2020).

[41] C. You, M. A. Quiroz-Juárez, A. Lambert, N. Bhusal, C. Dong, A. Perez-Leija, A. Javaid, R. d. J. León-Montiel, and O. S. Magaña-Loaiza, Identification of light sources using machine learning, Appl. Phys. Rev. **7**, 021404 (2020).

[42] C. Harney, S. Pirandola, A. Ferraro, and M. Paternostro, Entanglement classification via neural network quantum states, New J. Phys. **22**, 045001 (2020).

[43] Y.-C. Ma and M.-H. Yung, Transforming Bell's inequalities into state classifiers with machine learning, npj Quantum Inf. **4**, 34 (2018).

[44] M. Krenn, M. Malik, R. Fickler, R. Lapkiewicz, and A. Zeilinger, Automated Search for new Quantum Experiments, Phys. Rev. Lett. **116**, 090405 (2016).

[45] A. A. Melnikov, H. Poulsen Nautrup, M. Krenn, V. Dunjko, M. Tiersch, A. Zeilinger, and H. J. Briegel, Active learning machine learns to create new quantum experiments, Proc. Natl. Acad. Sci. USA **115**, 1221 (2018).

[46] L. O'Driscoll, R. Nichols, and P. A. Knott, A hybrid machine learning algorithm for designing quantum experiments, Quantum Mach. Intell. **1**, 5 (2019).

[47] M. Krenn, M. Erhard, and A. Zeilinger, Computer-inspired quantum experiments, Nat. Rev. Phys. **2**, 649 (2020).

[48] G. Torlai and R. G. Melko, Neural Decoder for Topological Codes, Phys. Rev. Lett. **119**, 030501 (2017).

[49] P. Baireuther, T. E. O'Brien, B. Tarasinski, and C. W. J. Beenakker, Machine-learning-assisted correction of correlated qubit errors in a topological code, Quantum **2**, 48 (2018).

[50] S. Krastanov and L. Jiang, Deep neural network probabilistic decoder for stabilizer codes, Sci. Rep. **7**, 11003 (2017).

[51] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, Reinforcement Learning with Neural Networks for Quantum Feedback, Phys. Rev. X **8**, 031084 (2018).

[52] D. Fitzek, M. Eliasson, A. F. Kockum, and M. Granath, Deep Q-learning decoder for depolarizing noise on the toric code, Phys. Rev. Research **2**, 023230 (2020).

[53] E. Flurin, L. S. Martin, S. Hacohen-Gourgy, and I. Siddiqi, Using a Recurrent Neural Network to Reconstruct Quantum Dynamics of a Superconducting Qubit from Physical Observations, Phys. Rev. X **10**, 011006 (2020).

[54] N. Wittler, F. Roy, K. Pack, M. Werninghaus, A. S. Roy, D. J. Egger, S. Filipp, F. K. Wilhelm, and S. Machnes, Integrated Tool-Set for Control, Calibration, and Characterization of Quantum Devices Applied to Superconducting Qubits, Phys. Rev. Appl. **15**, 034080 (2021).

[55] K. Bharti, T. Haug, V. Vedral, and L.-C. Kwek, Machine learning meets quantum foundations: A brief survey, AVS Quantum Science **2**, 034101 (2020).

[56] G. Torlai, J. Carrasquilla, M. T. Fishman, R. G. Melko, and M. P. A. Fisher, Wave-function positivization via automatic differentiation, Phys. Rev. Research **2**, 032060(R) (2020).

[57] N. Leung, M. Abdelhafez, J. Koch, and D. Schuster, Speedup for quantum optimal control from automatic differentiation based on graphics processing units, Phys. Rev. A **95**, 042318 (2017).

[58] M. Abdelhafez, D. I. Schuster, and J. Koch, Gradient-based optimal control of open quantum systems using quantum trajectories and automatic differentiation, Phys. Rev. A **99**, 052327 (2019).

[59] M. Krenn, J. Handsteiner, M. Fink, R. Fickler, R. Ursin, M. Malik, and A. Zeilinger, Twisted light transmission over 143 km, Proc. Natl. Acad. Sci. USA **113**, 13648 (2016).

[60] G. Mauro D'Ariano, M. G. Paris, and M. F. Sacchi, Quantum tomography, *Advances in Imaging and Electron Physics*, Vol. 128 (Elsevier, Amsterdam, 2003), p. 205.

[61] Y.-x. Liu, L. F. Wei, and F. Nori, Tomographic measurements on superconducting qubit states, Phys. Rev. B **72**, 014547 (2005).

[62] A. I. Lvovsky and M. G. Raymer, Continuous-variable optical quantum-state tomography, Rev. Mod. Phys. **81**, 299 (2009).

[63] S. Ashhab and F. Nori, Qubit-oscillator systems in the ultrastrong-coupling regime and their potential for preparing nonclassical states, Phys. Rev. A **81**, 042311 (2010).

[64] C. M. Caves, I. H. Deutsch, and R. Blume-Kohout, Physical-resource requirements and the power of quantum computation, J. Opt. B: Quantum Semiclassical Opt. **6**, S801 (2004).

[65] 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 (London) **567**, 209 (2019).

[66] Z. Hou, H.-S. Zhong, Y. Tian, D. Dong, B. Qi, L. Li, Y. Wang, F. Nori, G.-Y. Xiang, C.-F. Li, and G.-C. Guo, Full reconstruction of a 14-qubit state within four hours, New J. Phys. **18**, 083036 (2016).

[67] S. Deléglise, I. Dotsenko, C. Sayrin, J. Bernu, M. Brune, J. M. Raimond, and S. Haroche, Reconstruction of non-classical cavity field states with snapshots of their decoherence, Nature (London) **455**, 510 (2008).

[68] G. M. D'Ariano, D. F. Magnani, and P. Perinotti, Adaptive Bayesian and frequentist data processing for quantum tomography, Phys. Lett. A **373**, 1111 (2009).

[69] M. Cramer, M. B. Plenio, S. T. Flammia, R. Somma, D. Gross, S. D. Bartlett, O. Landon-Cardinal, D. Poulin, and Y.-K. Liu, Efficient quantum state tomography, Nat. Commun. **1**, 149 (2010).

[70] S. T. Flammia and Y.-K. Liu, Direct Fidelity Estimation from Few Pauli Measurements, Phys. Rev. Lett. **106**, 230501 (2011).

[71] D. Petz and L. Ruppert, Optimal quantum-state tomography with known parameters, J. Phys. A **45**, 085306 (2012).

[72] T. Baumgratz, D. Gross, M. Cramer, and M. B. Plenio, Scalable Reconstruction of Density Matrices, Phys. Rev. Lett. **111**, 020401 (2013).

[73] J. A. Smolin, J. M. Gambetta, and G. Smith, Efficient Method for Computing the Maximum-Likelihood Quantum State from Measurements with Additive Gaussian Noise, Phys. Rev. Lett. **108**, 070502 (2012).

[74] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. **2**, 303 (1989).

[75] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, Neural Networks **2**, 359 (1989).

[76] A. M. Schäfer and H. G. Zimmermann, Recurrent neural networks are universal approximators, in *International Conference on Artificial Neural Networks* (Springer, New York, 2006).

[77] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, Neural-network quantum state tomography, Nat. Phys. **14**, 447 (2018).

[78] I. Glasser, N. Pancotti, M. August, I. D. Rodriguez, and J. I. Cirac, Neural-Network Quantum States, String-Bond States, and Chiral Topological States, Phys. Rev. X **8**, 011006 (2018).

[79] F. Noé, S. Olsson, J. Köhler, and H. Wu, Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning, Science **365**, eaaw1147 (2019).

[80] A. Rocchetto, S. Aaronson, S. Severini, G. Carvacho, D. Poderini, I. Agresti, M. Bentivegna, and F. Sciarrino, Experimental learning of quantum states, in *Quantum Information and Measurement (QIM) V: Quantum Technologies* (OSA, Washington, D.C., 2019), p. F5A.26.

[81] 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).

[82] F. Flamini, N. Spagnolo, and F. Sciarrino, Visual assessment of multi-photon interference, Quantum Sci. Technol. **4**, 024008 (2019).

[83] G. Carleo, Y. Nomura, and M. Imada, Constructing exact representations of quantum many-body systems with deep neural networks, Nat. Commun. **9**, 5322 (2018).

[84] E. S. Tiunov, V. V. Tiunova (Vyborova), A. E. Ulanov, A. I. Lvovsky, and A. K. Fedorov, Experimental quantum homodyne tomography via machine learning, Optica **7**, 448 (2020).

[85] A. Melkani, C. Gneiting, and F. Nori, Eigenstate extraction with neural-network tomography, Phys. Rev. A **102**, 022412 (2020).

[86] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, Reconstructing quantum states with generative models, Nat. Mach. Intell. **1**, 155 (2019).

[87] P. Cha, P. Ginsparg, F. Wu, J. Carrasquilla, P. L. McMahon, and E.-A. Kim, Attention-based quantum tomography, arXiv:2006.12469.

[88] Z. Cai and J. Liu, Approximating quantum many-body wave functions using artificial neural networks, Phys. Rev. B **97**, 035116 (2018).

[89] D. P. Kingma and M. Welling, An introduction to variational autoencoders, Found. Trends Mach. Learn. **12**, 307 (2019).

[90] S. Mahdizadehaghdam, A. Panahi, and H. Krim, Sparse generative adversarial network, in *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019* (IEEE, Piscataway, NJ, 2019), pp. 3063–3071.

[91] A. Rocchetto, E. Grant, S. Strelchuk, G. Carleo, and S. Severini, Learning hard quantum distributions with variational autoencoders, npj Quantum Inf. **4**, 28 (2018).

[92] C. Zoufal, A. Lucchi, and S. Woerner, Quantum generative adversarial networks for learning and loading random distributions, npj Quantum Inf. **5**, 103 (2019).

[93] L. Hu, S.-H. Wu, W. Cai, Y. Ma, X. Mu, Y. Xu, H. Wang, Y. Song, D.-L. Deng, C.-L. Zou, and L. Sun, Quantum generative adversarial learning in a superconducting quantum circuit, Sci. Adv. **5**, eaav2761 (2018).

[94] S. Lloyd and C. Weedbrook, Quantum Generative Adversarial Learning, Phys. Rev. Lett. **121**, 040502 (2018).

[95] G. Torlai and R. G. Melko, Latent Space Purification via Neural Density Operators, Phys. Rev. Lett. **120**, 240503 (2018).

[96] G. Torlai, B. Timar, E. P. L. van Nieuwenburg, H. Levine, A. Omran, A. Keesling, H. Bernien, M. Greiner, V. Vuletić, M. D. Lukin, R. G. Melko, and M. Endres, Integrating Neural Networks with a Quantum Simulator for State Reconstruction, Phys. Rev. Lett. **123**, 230504 (2019).

[97] S. Lohani, B. T. Kirby, M. Brodsky, O. Danaci, and R. T. Glasser, Machine learning assisted quantum state estimation, Mach. Learn.: Sci. Technol. **1**, 035007 (2020).

[98] A. M. Palmieri, E. Kovlakov, F. Bianchi, D. Yudin, S. Straupe, J. D. Biamonte, and S. Kulik, Experimental neural network enhanced quantum tomography, npj Quantum Inf. **6**, 20 (2020).

[99] S. Ahmed, C. S. Muñoz, F. Nori, and A. F. Kockum, Quantum State Tomography with Conditional Generative Adversarial Networks, Phys. Rev. Lett. **127**, 140502 (2021).

[100] M. Neugebauer, L. Fischer, A. Jäger, S. Czischek, S. Jochim, M. Weidemüller, and M. Gärttner, Neural-network quantum state tomography in a two-qubit experiment, Phys. Rev. A **102**, 042604 (2020).

[101] S. Lloyd and S. L. Braunstein, Quantum Computation over Continuous Variables, Phys. Rev. Lett. **82**, 1784 (1999).

[102] M. Gu, C. Weedbrook, N. C. Menicucci, T. C. Ralph, and P. van Loock, Quantum computing with continuous-variable clusters, Phys. Rev. A **79**, 062318 (2009).

[103] C. Weedbrook, S. Pirandola, R. García-Patrón, N. J. Cerf, T. C. Ralph, J. H. Shapiro, and S. Lloyd, Gaussian quantum information, Rev. Mod. Phys. **84**, 621 (2012).

[104] T. Hillmann, F. Quijandría, G. Johansson, A. Ferraro, S. Gasparinetti, and G. Ferrini, Universal Gate Set for Continuous-Variable Quantum Computation with Microwave Circuits, Phys. Rev. Lett. **125**, 160501 (2020).

[105] J. E. Bourassa, R. N. Alexander, M. Vasmer, A. Patil, I. Tzitrin, T. Matsuura, D. Su, B. Q. Baragiola, S. Guha, G. Dauphinais *et al.*, Blueprint for a scalable photonic fault-tolerant quantum computer, Quantum **5**, 392 (2021).

[106] A. Grimm, N. E. Frattini, S. Puri, S. O. Mundhada, S. Touzard, M. Mirrahimi, S. M. Girvin, S. Shankar, and M. H. Devoret, Stabilization and operation of a Kerr-cat qubit, Nature (London) **584**, 205 (2020).

[107] A. Joshi, K. Noh, and Y. Y. Gao, Quantum information processing with bosonic qubits in circuit QED, Quantum Sci. Technol. **6**, 033001 (2021).

[108] P. Campagne-Ibarcq, A. Eickbusch, S. Touzard, E. Zalys-Geller, N. E. Frattini, V. V. Sivak, P. Reinhold, S. Puri, S. Shankar, R. J. Schoelkopf, L. Frunzio, M. Mirrahimi, and M. H. Devoret, Quantum error correction of a qubit encoded in grid states of an oscillator, Nature (London) **584**, 368 (2020).

[109] G. Papamakarios, Neural density estimation and likelihood-free inference, arXiv:1910.13233.

[110] J. Bae and L.-C. Kwek, Quantum state discrimination and its applications, J. Phys. A **48**, 083001 (2015).

[111] R. Takagi and B. Regula, General Resource Theories in Quantum Mechanics and Beyond: Operational Characterization via Discrimination Tasks, Phys. Rev. X **9**, 031053 (2019).

[112] M. S. Leifer and O. J. E. Maroney, Maximally Epistemic Interpretations of the Quantum State and Contextuality, Phys. Rev. Lett. **110**, 120401 (2013).

[113] C. Shen, R. W. Heeres, P. Reinhold, L. Jiang, Y.-K. Liu, R. J. Schoelkopf, and L. Jiang, Optimized tomography of continuous variable systems using excitation counting, Phys. Rev. A **94**, 052327 (2016).

[114] G. M. D Ariano, P. Perinotti, and M. F. Sacchi, Informationally complete measurements and group representation, J. Opt. B: Quantum Semiclassical Opt. **6**, S487 (2004).

[115] B. Qi, Z. Hou, L. Li, D. Dong, G. Xiang, and G. Guo, Quantum state tomography via linear regression estimation, Sci. Rep. **3**, 3496 (2013).

[116] A. Miranowicz, K. Bartkiewicz, J. Peřina, Jr., M. Koashi, N. Imoto, and F. Nori, Optimal two-qubit tomography based on local and global measurements: Maximal robustness against errors as described by condition number, Phys. Rev. A **90**, 062123 (2014).

[117] Z. Hradil, Quantum-state estimation, Phys. Rev. A **55**, R1561(R) (1997).

[118] K. Banaszek, G. M. D'Ariano, M. G. A. Paris, and M. F. Sacchi, Maximum-likelihood estimation of the density matrix, Phys. Rev. A **61**, 010304(R) (1999).

[119] J. Shang, Z. Zhang, and H. K. Ng, Superfast maximum-likelihood reconstruction for quantum tomography, Phys. Rev. A **95**, 062336 (2017).

[120] R. Blume-Kohout, Optimal, reliable estimation of quantum states, New J. Phys. **12**, 043034 (2010).

[121] C. Granade, J. Combes, and D. G. Cory, Practical Bayesian tomography, New J. Phys. **18**, 033024 (2016).

[122] J. M. Lukens, K. J. H. Law, A. Jasra, and P. Lougovski, A practical and efficient approach for Bayesian quantum state estimation, New J. Phys. **22**, 063038 (2020).

[123] J. Reháček, Z. Hradil, E. Knill, and A. I. Lvovsky, Diluted maximum-likelihood algorithm for quantum tomography, Phys. Rev. A **75**, 042108 (2007).

[124] J. L. E. Silva, S. Glancy, and H. M. Vasconcelos, Quadrature histograms in maximum-likelihood quantum state tomography, Phys. Rev. A **98**, 022325 (2018).

[125] C. Ferrie and R. Blume-Kohout, Maximum likelihood quantum state tomography is inadmissible, arXiv:1808.01072.

[126] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, Quantum State Tomography via Compressed Sensing, Phys. Rev. Lett. **105**, 150401 (2010).

[127] D. Gonçalves, M. Gomes-Ruggiero, and C. Lavor, A projected gradient method for optimization over density matrices, Optim. Methods Softw. **31**, 328 (2016).

[128] E. Bolduc, G. C. Knee, E. M. Gauger, and J. Leach, Projected gradient descent algorithms for quantum state tomography, npj Quantum Inf. **3**, 44 (2017).

[129] A. Kalev, R. L. Kosut, and I. H. Deutsch, Quantum tomography protocols with positivity are compressed sensing protocols, npj Quantum Inf. **1**, 15018 (2015).

[130] D. Ahn, Y. S. Teo, H. Jeong, D. Koutný, J. Rehacek, Z. Hradil, G. Leuchs, and L. L. Sanchez-Soto, Adaptive compressive tomography: A numerical study, Phys. Rev. A **100**, 012346 (2019).

[131] D. Ahn, Y. S. Teo, H. Jeong, F. Bouchard, F. Hufnagel, E. Karimi, D. Koutny, J. Rehacek, Z. Hradil, G. Leuchs, and L. L. Sanchez-Soto, Adaptive Compressive Tomography with No a priori Information, Phys. Rev. Lett. **122**, 100404 (2019).

[132] B. P. Lanyon, C. Maier, M. Holzäpfel, T. Baumgratz, C. Hempel, P. Jurcevic, I. Dhand, A. S. Buyskikh, A. J. Daley, M. Cramer, M. B. Plenio, R. Blatt, and C. F. Roos, Efficient tomography of a quantum many-body system, Nat. Phys. **13**, 1158 (2017).

[133] J. C. Bridgeman and C. T. Chubb, Hand-waving and interpretive dance: An introductory course on tensor networks, J. Phys. A **50**, 223001 (2017).

[134] K. Chabuda, J. Dziarmaga, T. J. Osborne, and R. Demkowicz-Dobrzański, Tensor-network approach for quantum metrology in many-body quantum systems, Nat. Commun. **11**, 250 (2020).

[135] G. Tóth, W. Wieczorek, D. Gross, R. Krischek, C. Schwemmer, and H. Weinfurter, Permutationally Invariant Quantum Tomography, Phys. Rev. Lett. **105**, 250403 (2010).

[136] T. Moroder, P. Hyllus, G. Tóth, C. Schwemmer, A. Niggebaum, S. Gaile, O. Gühne, and H. Weinfurter, Permutationally invariant state reconstruction, New J. Phys. **14**, 105001 (2012).

[137] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, A learning algorithm for Boltzmann machines, Cognit. Sci. **9**, 147 (1985).

[138] P. Smolensky, Information processing in dynamical systems: Foundations of harmony theory, Tech. Rep. (Colorado University at Boulder, Department of Computer Science, 1986), https://apps.dtic.mil/sti/citations/ADA620727.

[139] G. E. Hinton, *A Practical Guide to Training Restricted Boltzmann Machines* (Springer, Berlin, 2012), p. 599.

[140] A. Krizhevsky and G. Hinton, Learning multiple layers of features from tiny images, Tech. Rep. TR-2009 (University of Toronto, Toronto, 2009).

[141] G. E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Comput. **14**, 1771 (2002).

[142] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, 602 (2017).

[143] H. Manukian, Y. R. Pei, S. R. B. Bearden, and M. Di Ventra, Mode-assisted unsupervised learning of restricted Boltzmann machines, Commun. Phys. **3**, 105 (2020).

[144] P. Ghosh, M. S. M. Sajjadi, A. Vergari, M. Black, and B. Schölkopf, From variational to deterministic autoencoders, arXiv:1903.12436.

[145] A. Brock, J. Donahue, and K. Simonyan, Large scale GAN training for high fidelity natural image synthesis, arXiv:1809.11096.

[146] M. Lucic, M. Tschannen, M. Ritter, X. Zhai, O. Bachem, and S. Gelly, High-fidelity image generation with fewer labels, arXiv:1903.02271.

[147] V. V. Albert, K. Noh, K. Duivenvoorden, D. J. Young, R. T. Brierley, P. Reinhold, C. Vuillot, L. Li, C. Shen, S. M. Girvin, B. M. Terhal, and L. Jiang, Performance and structure of single-mode bosonic codes, Phys. Rev. A **97**, 032346 (2018).

[148] M. H. Michael, M. Silveri, R. T. Brierley, V. V. Albert, J. Salmilehto, L. Jiang, and S. M. Girvin, New Class of Quantum Error-Correcting Codes for a Bosonic Mode, Phys. Rev. X **6**, 031006 (2016).

[149] D. Gottesman, A. Kitaev, and J. Preskill, Encoding a qubit in an oscillator, Phys. Rev. A **64**, 012310 (2001).

[150] J. R. Johansson, P. D. Nation, and F. Nori, QuTiP: An open-source Python framework for the dynamics of open quantum systems, Comput. Phys. Commun. **183**, 1760 (2012).

[151] J. R. Johansson, P. D. Nation, and F. Nori, QuTiP 2: A Python framework for the dynamics of open quantum systems, Comput. Phys. Commun. **184**, 1234 (2013).

[152] G. Kirchmair, B. Vlastakis, Z. Leghtas, S. E. Nigg, H. Paik, E. Ginossar, M. Mirrahimi, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf, Observation of quantum state collapse and revival due to the single-photon Kerr effect, Nature (London) **495**, 205 (2013).

[153] J. Weinbub and D. K. Ferry, Recent advances in wigner function approaches, Appl. Phys. Rev. **5**, 041104 (2018).

[154] C. Eichler, D. Bozyigit, and A. Wallraff, Characterizing quantum microwave radiation and its entanglement with superconducting qubits using linear detectors, Phys. Rev. A **86**, 032106 (2012).

[155] P. D. Drummond and C. W. Gardiner, Generalised P-representations in quantum optics, J. Phys. A **13**, 2353 (1980).

[156] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, Tensorflow: A system for large-scale machine learning (2016).

[157] H. Liu, K. Simonyan, and Y. Yang, DARTS: Differentiable architecture search, arXiv:1806.09055.

[158] D. Choi, C. J. Shallue, Z. Nado, J. Lee, C. J. Maddison, and G. E. Dahl, On empirical comparisons of optimizers for deep learning, arXiv:1910.05446.

[159] R. M. Schmidt, F. Schneider, and P. Hennig, Descending through a crowded valley—Benchmarking deep learning optimizers, arXiv:2007.01547.

[160] E. Hazan, A. Klivans, and Y. Yuan, Hyperparameter optimization: A spectral approach, arXiv:1706.00764.

[161] Code for Quantum state classification and reconstruction with deep neural networks is available at https://github.com/quantshah/qst-nn (2021).

[162] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, Handwritten digit recognition with a back-propagation network, in *Advances in neural information processing systems* (Morgan Kaufmann Publishers Inc., San Francisco, CA, 1990), p. 396.

[163] A. L. Maas, A. Y. Hannun, and A. Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, Vol. 30 (JMLR, Atlanta, GA, 2013), p. 3.

[164] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. **15**, 1929 (2014).

[165] D. Ulyanov, A. Vedaldi, and V. Lempitsky, Instance normalization: The missing ingredient for fast stylization, arXiv:1607.08022.

[166] Z. Liu, T. Bicer, R. Kettimuthu, D. Gursoy, F. De Carlo, and I. Foster, TomoGAN: Low-dose synchrotron x-ray tomography with generative adversarial networks: Discussion, J. Opt. Soc. Am. A **37**, 422 (2020).

[167] B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, and G. R. G. Lanckriet, On integral probability metrics, ϕ-divergences and binary classification, arXiv:0901.2698.

[168] R. Agrawal and T. Horel, Optimal bounds between f-divergences and integral probability metrics, J. Mach. Learn. Res. **22**, 1 (2021).

[169] M. Arjovsky, S. Chintala, and L. Bottou, Wasserstein GAN, arXiv:1701.07875.

[170] Y. Bai, T. Ma, and A. Risteski, Approximability of discriminators implies diversity in GANs, arXiv:1806.10586.

[171] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, Loss functions for image restoration with neural networks, IEEE Trans. Comput. Imaging **3**, 47 (2017).

[172] T. Sercu and Y. Mroueh, Semi-supervised learning with IPM-based GANs: An empirical study, arXiv:1712.02505.

[173] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, Improved Training of Wasserstein GANs, *Advances in Neural Information Processing Systems* (Curran Associates Inc., Red Hook, NY, 2017).

[174] A. I. Lvovsky, Iterative maximum-likelihood reconstruction in quantum homodyne tomography, J. Opt. B: Quantum Semiclassical Opt. **6**, S556 (2004).

[175] A. Karpathy, Software 2.0 (2017) https://medium.com/@karpathy/software-2-0-a64152b37c35.

[176] T. Fawcett, An introduction to ROC analysis, Pattern Recognit Lett. **27**, 861 (2006).

[177] C. D. Brown and H. T. Davis, Receiver operating characteristics curves and related decision measures: A tutorial, Chemom. Intell. Lab. Syst. **80**, 24 (2006).

[178] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, Scikit-learn: Machine

learning in Python, J. Mach. Learning Research **12**, 2825 (2011).

[179] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, Grad-CAM: Visual explanations from deep networks via gradient-based localization, Int. J. Comput. Vision **128**, 336 (2020).

[180] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila, Noise2Noise: Learning image restoration without clean data, in *35th International Conference on Machine Learning, ICML 2018* 7, 4620 (PMLR, Stockholm, Sweden, 2018).

[181] A. Montanaro, On the distinguishability of random quantum states, Commun. Math. Phys. **273**, 619 (2007).

[182] A. Mandarino, M. Bina, C. Porto, S. Cialdi, S. Olivares, and M. G. A. Paris, Assessing the significance of fidelity as a figure of merit in quantum state reconstruction of discrete and continuous-variable systems, Phys. Rev. A **93**, 062118 (2016).

[183] W. C. Karl, Regularization in image restoration and reconstruction, *Handbook of Image and Video Processing* (Elsevier, Amsterdam, 2005), p. 183.

[184] V. A. Andreev, D. M. Davidović, L. D. Davidović, M. D. Davidović, V. I. Man'ko, and M. A. Man'ko, A transformational property of the Husimi function and its relation to the Wigner function and symplectic tomograms, Theor. Math. Phys. **166**, 356 (2011).

[185] D. Sych, J. Reháček, Z. Hradil, G. Leuchs, and L. L. Sánchez-Soto, Informational completeness of continuous-variable measurements, Phys. Rev. A **86**, 052123 (2012).

[186] S. T. Flammia, D. Gross, Y.-K. Liu, and J. Eisert, Quantum tomography via compressed sensing: Error bounds, sample complexity and efficient estimators, New J. Phys. **14**, 095022 (2012).

[187] O. Landon-Cardinal, L. C. G. Govia, and A. A. Clerk, Quantitative Tomography for Continuous Variable Quantum Systems, Phys. Rev. Lett. **120**, 090501 (2018).

[188] R. Nehra, M. Eaton, C. González-Arciniegas, M. S. Kim, T. Gerrits, A. Lita, S. W. Nam, and O. Pfister, Generalized overlap quantum state tomography, Phys. Rev. Research **2**, 042002 (2020).

[189] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and T. scikit-image contributors, SciKit-image: Image processing in Python, PeerJ **2**, e453 (2014).