THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

Quantum state characterization with deep neural networks

Shahnawaz Ahmed

Chalmers University of Technology
Microtechnology and Nanoscience - MC2
Applied Quantum Physics Laboratory
Göteborg, Sweden 2021

Quantum state characterization with deep neural networks
Shahnawaz Ahmed

Technical Report MC2-441
ISSN 1652-0769

Chalmers University of Technology
Microtechnology and Nanoscience - MC2
Applied Quantum Physics Laboratory
SE-412 96 Göteborg, Sweden
Telephone +46 (0)31 772 1000
www.chalmers.se

Author email: shahnawaz.ahmed@chalmers.se

Cover: Sketch of the idea explaining how a generative adversarial neural network can be used to learn quantum states.

Printed by Chalmers Digitaltryck
Göteborg, Sweden 2021

Quantum state characterization with deep neural networks
Shahnawaz Ahmed

Applied Quantum Physics Laboratory
Department of Microtechnology and Nanoscience (MC2)
Chalmers University of Technology

Abstract

In this licentiate thesis, I explain some of the interdisciplinary topics connecting machine learning to quantum physics. The thesis is based on the two appended papers, where deep neural networks were used for the characterization of quantum systems. I discuss the connections between parameter estimation, inverse problems and machine learning to put the results of the appended papers in perspective. In these papers, we have shown how to incorporate prior knowledge of quantum physics and noise models in generative adversarial neural networks. This thesis further discusses how automatic differentiation techniques allow training such custom neural-network-based methods to characterize quantum systems or learn their description. In the appended papers, we have demonstrated that the neural-network approach could learn a quantum state description from an order of magnitude fewer data points and faster than an iterative maximum-likelihood estimation technique. The goal of the thesis is to bring such tools and techniques from machine learning to the physicist's arsenal and to explore the intersection between quantum physics and machine learning.

Keywords: Machine learning, quantum computing, deep neural networks, classification, reconstruction, quantum state tomography, quantum state discrimination, bosonic states, optical quantum states.

Acknowledgements

I would like to thank everyone who has been a part of this journey starting with my supervisor Anton Frisk Kockum. Thank you for your constant encouragement, support and guidance. I appreciate your careful edits, comments and insights on my work and I am grateful for having a supervisor who cares so deeply. I could not have asked for a better guide and mentor.

I am very grateful to Göran Johansson for the opportunity to pursue this interesting research direction. A special thanks to my collaborator Carlos Sánchez Muñoz, who got me started with the project, and Franco Nori, who encouraged me to freely explore my ideas. The time at RIKEN under the guidance of Nathan Shammah, Neill Lambert, and Clemens Gneiting was pivotal in developing my skills and interests leading to this work.

I would like to thank my brilliant co-workers at AQP. My sincere thanks to Ingrid Strandberg and Fernando Quijandría for always being available to discuss ideas and clarify my doubts. A special thanks to Ingrid Strandberg, Yu Zheng, Pontus Vikstål, and Nathan Shammah for their valuable comments on my drafts. I am also thankful to Marina Kudra, Yong Lu, and Simone Gasparinetti for helping me understand experimental data and showing me the bigger picture.

My family and friends are a constant source of encouragement and support in my life and I would like express my whole-hearted gratitude to them. Lastly, this would not have been possible without the kind support from Okka Köster.

Göteborg, March 2021
Shahnawaz Ahmed

Publications

I. **Quantum state tomography with conditional generative adversarial networks**
Shahnawaz Ahmed, Carlos Sánchez Muñoz, Franco Nori, Anton Frisk Kockum
Submitted (2021). arXiv:2008.03240

II. **Classification and reconstruction of quantum states with deep neural networks**
Shahnawaz Ahmed, Carlos Sánchez Muñoz, Franco Nori, Anton Frisk Kockum
Submitted (2021). arXiv:2012.02185

Contents

Contents

Chapter 1

Introduction

Quantum mechanics is one of the most successful theories in physics. It describes the behaviour of quantum systems and can be applied to the study of elementary particles [1], atoms [2], electromagnetic fields [3] (light), and even black holes [4]. However, simulating quantum physics using classical computers could be challenging [5–9]. Therefore, the ability to simulate quantum physics or harness quantum information using quantum computers may have revolutionary consequences for science and technology [10–15].

In order to use the parallelism that quantum physics could provide by allowing to represent information in coherent superpositions of a quantum system, we must first tackle several practical difficulties [16–18]. A full classical description for such a superposition of states would require an exponentially growing number of parameters [19]. Also, we need to keep track of a number of classical parameters in quantum devices [13]. Therefore the characterization and control of quantum devices is difficult. The delicate nature of quantum information further complicates the problem since entanglement between different parts of a quantum system and its surroundings may amplify the effects of various types of noise [20]. Therefore, to be able to develop quantum information processing capabilities, new tools and techniques are required to solve issues related to the characterization and control of quantum devices.

Machine learning is emerging as one such tool that is being increasingly used to address some of the challenges in the field of quantum information. Machine learning aims to develop techniques that learn from data and emulate intelligence. The intersection of quantum physics and machine learning promises exciting new possibilities — quantum information processing could

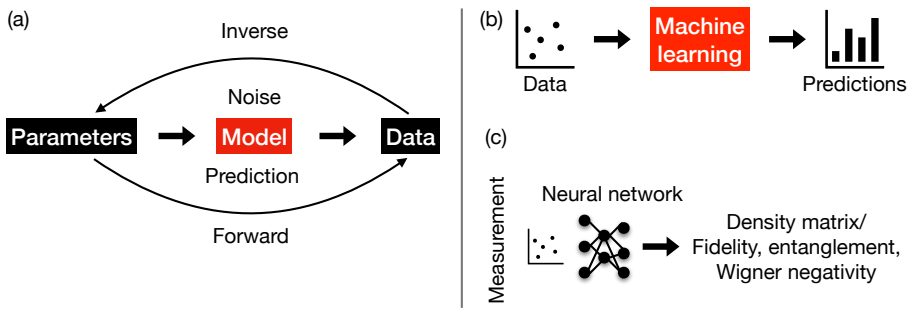

Figure 1.1: Many tasks in quantum information and computing can be framed as parameter estimation or inverse problems. (a) Inverse problems deal with estimating parameters from observed data that are related by a forward model. In most cases, due to noise, inverse parameter estimation problems are *ill-posed* and regularization methods are necessary to solve them. Prior information can be used to reduce the search space for parameters and constraint the problem to tackle ill-posedness. (b) Machine learning could offer an alternative automated procedure for parameter estimation or prediction of quantities of interest directly from data. (c) A neural network can take measurement data from a quantum system as input and then either generate a state description or directly predict properties of the state. Learning from data could help tackle some of the difficult inverse problems arising in the area of quantum information and computing.

speed up and enhance machine learning [21] while the latter can be applied to solve various problems in quantum physics [22]. In this thesis, we will discuss the latter and show how to apply neural-network-based machine learning to the problem of quantum state characterization.

Machine-learning-based techniques were recently shown to be promising for several problems in quantum information, e.g., faster tuning of quantum devices compared to human experts [23], designing of quantum experiments [24, 25], automated calibration, control & characterization [26, 27] and decreasing error rates for qubit readout [28]. Some of the most successful machine-learning techniques today use neural networks [29]. Therefore neural networks have also been applied to problems in quantum physics with some success [22, 30, 31]. Let us start by elucidating the relationship between quantum state characterization, machine learning, and the general idea of parameter estimation in inverse problems, see Fig. 1.1.

1.1 Quantum physics and machine learning

Quantum physics traces its roots to Max Planck's attempts to explain the spectrum of blackbody radiation [32]. In the late 1800's, the prevailing theories of (classical) physics made a non-sensical prediction — an ideal blackbody in thermal equilibrium should emit radiation at all frequencies, with more energy radiated as the frequency increases. This results in the conclusion that such an object will radiate all its energy instantaneously and was called the ultraviolet catastrophe. Classical physics was unable to model the experimentally observed spectrum of a blackbody relating the intensity of the emitted radiation to its frequency.

Max Planck considered oscillating charged particles emitting and absorbing radiation to model a blackbody. After making the assumption that energy can only be emitted or absorbed in discrete quanta, he derived an equation that perfectly described the data from a blackbody experiment [33]. The birth of quantum mechanics was therefore an attempt to fit observed data to a new model.

More than a century later, we are now at a stage where it is possible to manipulate information in quantum devices and develop the building blocks of quantum computers. A microwave cavity [34] is an example of such a quantum device. It is also a close approximation to a blackbody that works according to the principles of quantum physics. In the papers that this thesis is based on, we will take the example of quantum states in cavities for our demonstrations.

However, even though we have now developed the theory of quantum physics, we still need to fit observed experimental data to our models. There is a large computational effort involved in simulating, characterizing and verifying the working of quantum devices. The automation of some of the routines with machine-learning techniques could be beneficial to address some of the difficulties and reduce the computational effort required for characterization and control.

In order to simply model what exists inside a quantum device, a large number of measurements have to be performed to determine the exponentially growing number of parameters describing the state. The data analysis necessary to learn the full quantum state description presents a further challenge [35]. Then, we face the problem of decoherence that corrupts the encoded information in a quantum system due to entanglement with the environment [18, 36]. In addition to other types of experimental noise in the data, such issues motivate the use of machine-learning-based tools to

process and analyze quantum data and control a quantum device.

Machine-learning algorithms are designed such that they can automatically learn and improve their performance on a task with experience. This is usually achieved by training the algorithm with data, allowing it to recognize and exploit patterns in the data. With the availability of more data, new machine-learning algorithms running on better hardware are solving problems that posed significant issues for computers before. Tasks such as face recognition, automated driving, and natural language processing are very difficult to solve using hand-crafted algorithms. However, machine-learning algorithms can tackle such tasks [37–40] even achieving super-human performance [41, 42], and showing some level of creativity [43, 44].

The backbone of many such machine-learning algorithms today are neural networks that are loosely inspired by the structure of human brains and seemingly emulate intelligence. Beyond solving tasks that could be easy for humans such as driving a car, neural-network-based algorithms have also been applied to defeat or match humans at various types of games [41, 45]. New self-learning algorithms can even discover the rules of games by themselves [42]. More recently, machine-learning techniques have also been applied successfully to solve grand scientific challenges such as the protein-folding problem, outperforming other human-developed techniques and models [46]. In Fig. 1.2 we present a few examples of tasks that neural networks can be applied to solve which were difficult for computers to solve before.

However, with bigger machine learning models and an ever-increasing amount of data, we need new ways to manipulate information. Quantum information processing could therefore potentially provide a boost to machine learning, [21] but before that is feasible we need to improve upon the quantum technologies of today. Machine learning techniques appear to be worth exploring for that purpose.

1.2 Quantum systems today

In 2019, a small quantum computer using 53 superconducting qubits was demonstrated to create a quantum state that was the output of a pseudo-random quantum circuit [13]. It was seen as a landmark achievement in our ability to control a programmable quantum system. This experiment allowed a calculation that would require manipulating information in 2^{53} dimensions

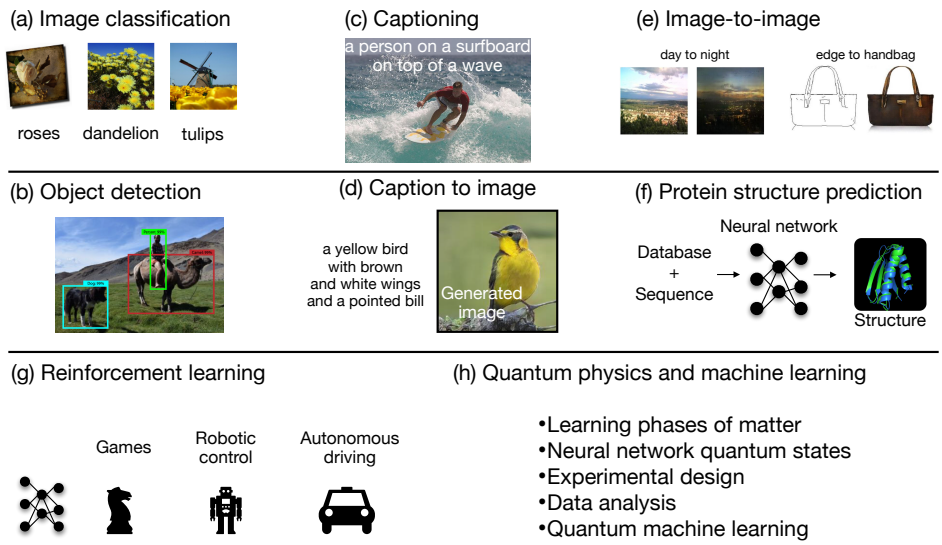

Figure 1.2: Neural-network-based machine learning is hugely successful in various types of tasks. (a) Image classification was one of the early successful applications [47]. We show a few examples from the popular dataset [48] that can be classified using a neural network. (b) Detection of objects in images was also a difficult task for computers that neural-network-based machine learning could solve [49]. The example shown here is from a recent version of the YOLO (You only look once) detection algorithm [50]. (c) Generating a textual description of images is a very interesting application of neural networks which shows the extent to which we could use them for descriptive classification beyond just labelling [51]. The image is taken from [52] and the caption was generated by a neural network trained using TensorFlow [53]. (d) Generative neural networks can also be taught to create data based on some input description such as text [54, 55]. (e) Image-to-image generation shows how generative models can learn maps between different data spaces [56]. (f) Neural networks also form the back-bone of many machine-learning methods and have recently showed success in a very difficult scientific problem — protein structure prediction [46]. (g) Reinforcement learning also uses neural networks to learn how to play games, or for control problems such as autonomous driving [39]. (h) Machine learning could have many applications in quantum physics and some of them have already demonstrated the benefit of using neural networks.

in the worst-case scenario. The calculation of the distribution of outputs would supposedly take around $10,000$ years on a classical supercomputer and requires $10,000$ terabytes of data to simply store the information describing the state. Even if efficient algorithms could possibly allow similar simulations on classical computers [57], this experiment showed the incredible potential and challenges of building a quantum device for information processing.

The calibration and control of this device required dealing with hundreds of parameters per qubit such as frequencies for each qubit, readout resonators and couplers between qubits [13]. The control of the qubits is through electromagnetic pulses and therefore analog in nature. A careful choice of continuous-valued parameters is then necessary to drive the system towards a desired target state. Automated calibration using a strategy named "Optimus" [58] was therefore essential to handle the large number of parameters. Typically the quantum computer required 36 hours for initial setup and 4 hours per day for calibration before experiments could be run on it. In other similar experiments with superconducting qubits, post-processing of data could take upto a week without specialized techniques such as GPU parallel programming [35]. These timescales and parameters demonstrate how difficult it would be to operate a quantum computer with the thousands of qubits necessary for error correction [59, 60].

Apart from superconducting qubits, bosonic systems [61] and trapped-ion based architectures [62] are some of the major platforms for realizing quantum computers. Bosonic qubits store and manipulate information in continuous degrees of freedom such as the electromagnetic field (light) trapped in a superconducting cavity. The typical hardware consists of a superconducting cavity where nonlinearities are introduced using a Josephson-junction-based qubit. Coupling cavities mediate interaction between different bosonic qubits using fixed or tunable resonance frequencies. Read-out cavities are required to obtain the information from the qubits. Again, there will be many parameters [63] that need to be estimated and optimized in such a system — resonator and qubit frequencies, non-linearities, interaction strengths, drive strengths and so on.

Ion traps use isolated ions for quantum computing by encoding information in the internal energy levels of atoms or using motional states. The information can be manipulated and extracted using lasers to excite the atomic transitions. One of the largest ion-trap quantum computers currently promises 32 qubits [64] but even with smaller systems, it has been

possible to simulate molecules such as water and compute its ground-state energy [65]. In such systems, variables such as the frequency, amplitude, phase or power of the control lasers, ion positions need to be tracked and automated methods become invaluable as systems scale up. Other emerging technologies such as neutral atoms [66, 67], silicon qubits [68, 69], quantum dots [70], diamond NV centers [71], and even molecules as qubits [72, 73] have also been proposed.

In other words, quantum computers today show a promising future where we can manipulate information at a scale unimaginable for classical computers. However, all such technologies face similar issues for calibration, control and characterization due to the large parameter space involved and intrinsic limitations due to quantum physics. The scaling up of such systems would require significant leaps to solve some of the difficult inverse problems related to parameter estimation and control. In the next section, we will present the general idea of inverse problems and parameter estimation that could be useful to understand how to address several of the challenges discussed above.

1.3 Inverse problems and parameter estimation

Inverse problems [74, 75] deal with determining parameters of interest, $p \in \mathcal{P}$, in a problem from data $d \in \mathcal{D}$. We consider a measurement operator, $\mathcal{O}$ that maps parameters to data in a forward problem. The measurement operator could be a causal model or a theory, see Fig. 1.1(a). The inverse problem tries to reconstruct the parameters from observed data as a solution to

$$d = \mathcal{O}(p). \tag{1.1}$$

The parameter and data spaces $(\mathcal{P}, \mathcal{D})$ are typically Banach or Hilbert spaces such that there exists an appropiate distance measure between the elements defined using the norm. It is also possible to formulate control as an inverse problem where we try to design a set of parameters to have a desired outcome.

Three central questions arise in the sense of Hadamard's beliefs about mathematical models for physical processes and their well-posed nature [76] : (i) existence of a solution, (ii) uniqueness of the solution, and (iii) continuous dependence of the solution on data. The first two beliefs assure that we have a consistent mathematical model for describing the system and the solution is a description of the reality of the physical process. The third

point analyses the stability of the model under noise or perturbations. Problems that are not well-posed according to the above beliefs are termed as *ill-posed*.

Inverse problems are often ill-posed. Noise is one of the main causes since noise could get amplified due to the inversion. The inversion could also be ill-posed due to an incorrect choice of the measurement operator, having a poor model or inadequate data. The simplest approach to deal with ill-posed problems could be to change the measurement operator or collect more data. However, if the amplification of noise is very high during inversion, collecting more data might not be the best solution. A useful strategy is putting constraints on the parameters in the form of prior knowledge. A noise model σ can be introduced as $d = \mathcal{O}(p) + \sigma$ to model detector noise or errors and imperfections in our model. These techniques that include prior knowledge and try to simplify the problem can be broadly termed as regularization.

Regularization techniques could incorporate prior beliefs about the desired parameters sparsity [77]. Typically the problem is cast as optimizing a functional to find the parameters, for example,

$$p_\gamma = \arg\min_p \|\mathcal{O}(p) - d\|_{L_2} + \gamma \|p\|_{L_1}, \tag{1.2}$$

where $\gamma > 0$ is a regularization parameter. We used the L_2 and L_1 norm for the error and promoting sparsity of parameters p respectively. The k-norm for the finite dimensional vector space is defined as

$$\|\mathbf{v}\|_{L_k} = \sum_i (|v_i|^k)^{\frac{1}{k}}, \tag{1.3}$$

where $\mathbf{v}$ is a vector. The L_1 norm on the parameters penalizes non-zero values and therefore promotes sparsity.

However, sometimes the simple regularized least-squares optimization in Eq. (1.2) is not sufficient to obtain the parameters of interest. If the available data is not adequate, the Bayesian framework is suitable to include more prior information [75]. A prior distribution $\pi_0(p)$ is assumed that assigns a probability (density) for the parameters p. The noise model is incorporated with a likelihood function $\pi(d|p)$ that defines the conditional probability of the data d when we assume parameters p. The posterior distribution of the parameters is then simply given by Bayes theorem as

$$\pi(p|d) \propto \pi(d|p)\pi_0(p), \tag{1.4}$$

Now if we have a good prior for the parameters and sufficient data, we can obtain a posterior distribution for p that not only gives a single estimate but also tells us about the errors in the parameters. Therefore Bayesian techniques are very useful for parameter estimation, but they come with their own set of challenges [78]. It can be computationally challenging to compute the posterior as we need to sample $\pi(p|d)$ to estimate $\pi(d|p)$ by repeatedly simulating the forward operation $\mathcal{O}(p)$ [79].

It is clear that inverse problems are challenging in general and the simplest examples such as numerical differentiation of a noisy function can become very difficult [80]. In quantum information and computing, the problem of reconstructing a state's wavefunction or density matrix from measurements on the state can be seen as a simple linear inversion problem.

The transfer of one state to another or targeting a specific quantum state is an example of an optimal control problem that can also be related to inverse problems [81]. If y_t is some target state and we consider a partial-differential equation governing the evolution of states y as $\Delta y = u$ where u defines the desired control function, then we may seek a solution to

$$\arg\min_{y,u} \|y - y_t\|_{L_2}^2 + \gamma \|u\|_{L_1}, \text{s.t} \Delta y = u. \tag{1.5}$$

This optimization problem is very similar to Eq. (1.2) if we relate the measurement operator to the solution operator for the partial differential equation $\Delta y = u$ such that $y(\tau) = \mathcal{O}(u(\tau), y_0)$ with $y(\tau)$ denoting the state at time τ.

In the discussion so far, we have considered a model for the forward problem using the abstract idea of a measurement operator. Mathematical models that relates parameters to observed data are carefully designed and proposed by using observations and prior knowledge. The goal of inverse problems is to learn parameters of such models. Therefore we can formulate a wide variety of problems in science as inverse problems. But what about problems where it is not possible to clearly define a model, construct a well-defined measurement operator or write down mathematical formulations of our prior knowledge? Take the task of driving cars where the data could be video streams or lidar sensor inputs and we are interested in the control parameters — acceleration/braking and turning of the wheels. It is not straightforward to write a simple forward or backward model to capture driving and relate the data to the parameters of interest. Yet, human beings can easily determine the parameters in a dynamic way. This leads to the idea of learning automatically from data — *machine learning*

— that shares several ideas and concepts with inverse problems.

1.4 Machine learning for inverse problems

Machine learning can be related to regularization in inverse problems [82]. In a typical machine-learning problem, we aim to find a relationship between some input space X and output Y. The inputs $x \in X$ can be images and the outputs could be a decision or a label $y \in Y$. This is slightly different from the inverse problem formulation where we are interested in parameters mapped to some data instead of mapping two different data spaces. But the connection with parameter estimation will become clear soon.

The hope is that the learned relationship f can predict the outputs for new inputs $y = f(x)$ after training on a training dataset of samples $\{x_i, y_i\}$. We can assume a joint distribution $p(x, y) = \pi(x)\pi(y|x)$ that relates the inputs and outputs which is often intractable in real-world scenarios. But the idea is to capture this underlying distribution through the proposed f which is typically a learnable function. We can define an expected risk that measures how well f describes the data using a positive *loss function*:

$$R[f] = \int_{X \times Y} \text{loss}(f(x), y) dp(x, y). \tag{1.6}$$

Since we only have a finite amount of examples (x_i, y_i), the minimization of the expected risk is difficult. To tackle this, we consider a regularized least-squares minimization [82], fix some hypothesis space such as a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ [83] and look for a function f according to the following minimization:

$$\min_{f \in \mathcal{H}} \left[\frac{1}{N} \sum_{i=1}^{N} (f(x_i) - y_i)^2 + \gamma \Omega(f) \right]. \tag{1.7}$$

In a RKHS, two functions f and g that are close in their norm $||f - g||$ are also pointwise close $||f(x_i) - g(x_i)||$. The penalty term Ω could be $\Omega(f) = \|f\|_k^2$ with $\|*\|_k$ as the norm of the space $\mathcal{H}$.

The function f can now be approximated using a model $f(x; \theta)$ where θ represents the parameters of the model that can be learned by optimizing the loss function using the available data. If we consider Eq. (1.2) and Eq. (1.7), we can relate the measurement operator (or forward model) in inverse problems to the function f in machine learning. The parameters

that we are interested in are θ defining the function. Therefore learning from examples can be related to ill-posed inverse problems where we do not explicitly design a model but learn it from data. Recently, there are increasing applications of machine learning, specifically using neural networks to solve difficult inverse problems [84–86].

One choice for the family of parameterized functions to model f are neural networks. They are one of the most successful tools in machine learning today [29] that define a structure for $f(x;\theta)$ with θ being learnable parameters. Neural networks can act as universal function approximators from $\mathbb{R}^n \to \mathbb{R}^m$ [87, 88]. Therefore they can be used to replace the unknown forward and inverse models and estimated from data. More recently a combination of neural networks and model-based approaches have shown success in some inverse problems. The addition of priors to the neural network architecture could enable the use of the universal approximation capacity of neural networks as well as leverage human knowledge [86, 89, 90]. A striking example could be seen as the design of convolutional neural networks for image processing inspired by structure the human visual cortex [91].

One of the main contributions in this thesis is exploring of how quantum-mechanical rules can be used in conjunction with neural networks. This merger between neural networks and quantum physics opens up possibilities to use them in quantum physics with a better understanding and control of their black-box nature.

1.5 Outline of the thesis

The thesis is based on two appended papers that discuss applications of neural networks for characterizing quantum states. The introduction motivates how machine learning and quantum physics could benefit each other. Here we have discussed the connection between ill-posed inverse problems, machine learning and the necessity of new tools for characterizing and controlling quantum systems.

Chapter 2 discusses quantum state descriptions and their measurements. In the context of ill-posedness of inverse problems, we need to understand the noise that arises in such systems and the notion of informational completeness. I discuss these concepts, present the problem of quantum state reconstruction and issues faced by different existing methods in Chapter 2. The reconstruction of a quantum state from data is the central

problem addressed in this thesis. Additionally, I will also discuss weaker models for quantum state reconstruction that could simplify the problem and where neural networks might find potential use.

In Chapter 3, I present the theoretical tools and methods required for machine learning with neural networks. The core of many learning algorithms, differential programming, can be applied beyond just training neural networks. In this chapter, I will discuss the general working of such automatic differentiation with a brief presentation of the backpropagation algorithm. Then, I present various generative and discriminative models using neural networks that will be the main tools for the results in Paper I and Paper II.

Chapter 4 is devoted to the methods that apply neural networks to the problem of identifying quantum states and reconstructing their descriptions. This chapter is the backbone of the thesis that shows how to use neural networks for quantum physics problems. Further I discuss how quantum physics knowledge and noise models can be integrated into standard deep neural networks with custom architectures.

Chapter 5 gives a short overview of the results from Paper I and II. In Paper I, we propose a new technique to reconstruct quantum states using a very successful neural-network architecture called a generative adversarial network. Our approach can reconstruct quantum states from experimentally measured outcomes with orders of magnitude fewer datapoints or iterative steps than an iterative maximum-likelihood estimation. We also demonstrate single-shot reconstructions without any iterative steps using a network trained on simulated data.

Paper II discusses the general question of classification and reconstruction of quantum states using neural networks in the context of discriminative and generative modeling. We first show that distinguishing properties of quantum states from noisy data is possible using a simple convolutional neural network that identifies different bosonic states. Then, we show how a standard feedforward neural network can be adapted for state reconstruction by including quantum physics knowledge. The second part of Paper II is therefore a much more detailed investigation of the ideas presented in Paper I. We motivate how using a second neural network can act as a better loss function in combination to a standard loss metric. This is the crucial new idea where we allow neural networks to learn what it means to be a particular quantum state from observations and let it adapt to different situations and noise. Our approach proves useful and efficient in

terms of the time and data necessary for reconstruction and handles noise reasonably well.

Finally, in Chapter 6 we conclude with a summary and possible directions for the future.

Chapter 2

Learning quantum states

One of the basic postulates of quantum mechanics is that a quantum state is described by a complex vector $|\psi\rangle$ in a Hilbert space. Hermitian (self-adjoint) operators $\mathcal{E}$ can act on these complex vectors and their (real) eigenvalues e denote the observable quantities for the state:

$$\mathcal{E}\,|\psi\rangle = e\,|\psi\rangle\,. \tag{2.1}$$

The complete set of eigenvectors of such a Hermitian operator ($\mathcal{E}|e_i\rangle = e_i|e_i\rangle$) form an orthonormal basis for the Hilbert space. Therefore $\langle e_i|e_j\rangle = \delta_{ij}$ where δ_{ij} is the Kronecker-delta function and $\sum_i |e_i\rangle\langle e_i| = \mathbb{I}$. Any quantum state can be written in this basis as

$$|\psi\rangle = \sum_i c_i\,|e_i\rangle\,, \tag{2.2}$$

where c_i are complex-valued probability amplitudes such that the probability of observing an outcome with eigenvalue e_i is $|c_i|^2$. The state vector describes a pure quantum state. We can define a density matrix to represent a mixture of different pure states as

$$\rho = \sum_k p_k\,|\psi_k\rangle\langle\psi_k|\,, \tag{2.3}$$

where $\sum p_k = 1$ denotes the probability for each possible state $|\psi_k\rangle$. If we expand the state vectors $|\psi_k\rangle$ using the basis vectors $\{e_i\}$ from Eq. (2.2), the density matrix is characterized by the complex-valued coefficients ρ_{ij} given by

$$\rho = \sum \rho_{ij}\,|e_i\rangle\langle e_j|\,. \tag{2.4}$$

In case of a continuous-variable observable such as position, x, we consider the position operator X and its eigenstates $|x\rangle$,

$$X\,|x\rangle = x\,|x\rangle\,. \tag{2.5}$$

Assuming orthogonality and completeness, we postulate

$$\langle x'|x\rangle = \delta(x'-x) \tag{2.6}$$

and $\int_{-\infty}^{\infty} |x\rangle\langle x|\,dx = 1$. A general state in this continuous basis can be written as

$$|\psi\rangle = \int_{-\infty}^{\infty} \psi(x)\,|x\rangle\langle x|\,dx, \tag{2.7}$$

where $\psi(x) = \langle x|\psi\rangle$ is the wave function. [92, 93]

The learning of quantum states is the task of estimating the parameters describing a quantum state, such as the density matrix or wavefunction, from measurements on the state. This task is called quantum state tomography. The most common scheme for measurement is the von Neumann model, according to which the act of measurement irrevocably disturbs the state and changes it. Therefore measurements usually need to be repeated to collect the full statistics that can give us all the information necessary to construct the quantum state description. Since the measurement process involves interactions between different quantum and classical systems, noise can affects the data at various points and needs to be accounted for during tomography.

In this chapter, we will discuss the basic concepts that are required to understand quantum state tomography and different types of noise that make reconstruction difficult. We will also briefly outline some of the standard learning models and techniques used for tomography and the cost of data collection and computation associated with them.

2.1 Quantum state descriptions

The density matrix, ρ of a quantum state is given by Eq. (2.3) for a mixture of different pure quantum states. The rank r of this matrix determines the number of pure quantum states in the mixture. As an example, consider a pure two-level quantum system ($r = 1$) expressed in the eigenbasis of an observable that tells us if the state is in the ground state $|0\rangle$ or excited

state $|1\rangle$. This basis called the computational basis. It basis spans the two-dimensional Hilbert space such that we can write a state vector as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{2.8}$$

where α and β are complex amplitudes. The probabilities for observing the state in ground or the excited state is given by $|\alpha|^2$ or $|\beta|^2$, respectively. The density matrix for this state would be of the form

$$\rho = \rho_{00}\,|0\rangle\langle 0| + \rho_{01}\,|0\rangle\langle 1| + \rho_{10}\,|1\rangle\langle 0| + \rho_{11}\,|1\rangle\langle 1|\,. \tag{2.9}$$

However, since the state is pure, we know that

$$\rho_{00} = |\alpha|^2; \rho_{11} = |\beta|^2; \rho_{10} = \beta\alpha^*; \rho_{01} = \alpha\beta^*, \tag{2.10}$$

and only the complex parameters α need to be estimated as a global phase could be ignored. For states with higher rank (r $>$ 1), i.e., a mixture of pure states, we need to determine more terms in the density matrix.

If we now consider n such two-level systems in the computational basis, the possible number of basis vectors grows exponentially as 2^n:

$$\{|000...\rangle\,, |100...\rangle\,, |010...\rangle\,, ..., |111...\rangle\} \tag{2.11}$$

Therefore, the general density matrix for a composite quantum system with n units, with each unit having k dimensions, will be given by a $d \times d$ matrix of complex numbers, ρ_{ij}, where $d = k^n$. In general, without any assumption, we therefore need to estimate $d^2 - 1$ elements to fully characterize the density matrix.

The exponentially increasing size of the density matrix makes it difficult to estimate it. However, we can exploit prior knowledge in the form of quantum mechanical rules to decrease the number of parameters that need to be learned. The diagonal entries of the density matrix represent probabilities of occupying one of the basis states. Therefore, they should be real and positive, between 0 and 1 and sum to unity to be meaningfully interpreted as probabilities.

Assuming that the density matrix is positive semidefinite and has unit trace satisfies these conditions. Further, every positive-semidefinite operator is Hermitian. Therefore, while determining an unknown quantum state's density matrix we can restrict to some parameterization that ensures positive-semidefiniteness and unit trace. The Cholesky decomposition gives such a parameterization for a positive-semidefinite matrix as:

$$\rho = TT^\dagger \tag{2.12}$$

with T being a complex lower-triangular matrix with real entries on the diagonal [94].

The density matrix is the most general parameterization for a mixed quantum state with an exponentially growing number of parameters. However, there could be other restrictions on the quantum state that allows us to write efficient descriptions for the state and therefore reduce the number of elements that need to be estimated. Consider the example of a Greenberger-Horne-Zeilinger (GHZ) state of three two-level systems given by

$$|\mathrm{GHZ}\rangle = \frac{|000\rangle + |111\rangle}{\sqrt{2}}. \tag{2.13}$$

If we try to reconstruct the density matrix of an unknown GHZ state, we only need the four density matrix elements at the corners of the density matrix given by the coefficients of

$$|000\rangle\langle 000|\,, |111\rangle\langle 111|\,, |000\rangle\langle 111|\,, |111\rangle\langle 000|\,. \tag{2.14}$$

Similarly, consider a class of quantum states such as a binomial quantum state written in the Fock basis $\{|n\rangle\}$. These states are a superposition of Fock states with the weights given by the binomial coefficients [95]

$$|\psi_{\texttt{binomial}}\rangle = \frac{1}{\sqrt{2^{N+1}}} \sum_{m=0}^{N+1} \sqrt{\binom{N+1}{m}}\, |(S+1)m\rangle\,. \tag{2.15}$$

The states are simply parameterized by the integers N and S.

If we have such prior information about the state, we can use an efficient representation for the state and learn a reduced number of parameters that can help us reconstruct that state. In general, efficient representations such as a matrix product state [96] or a tensor network [97] could greatly reduce the complexity of the problem. In fact, a recent work on using a tensor network method to simulate a 53 qubit experiment using a small cluster of GPUs showed significant success [57]. In this regard, we are using the idea of regularization and priors in inverse problems to restrict the search space of parameters.

After we decide on a specific quantum state description such as the density matrix, let us consider how measurements on the state extract information about the parameters describing the state.

2.2 Measurements

Measurement in quantum mechanics is a topic of fascinating fundamental and philosophical oddities. They are a bridge between the quantum world and our macroscopic reality that allow us to determine quantum mechanical properties of a system. Measurements are carried out by an interaction of the quantum system with a measuring apparatus that gives a classical outcome.

Let us follow the discussion in [93] to demonstrate how such measurements can be realized in general. Let measurement operators $\{M_i\}$ on a quantum state in a Hilbert space $\mathcal{H}$ have different outcomes labeled by i. The probability that a measurement on the quantum state $|\psi\rangle$ results in the outcome i is

$$p(M_i) = \langle\psi| M_i^\dagger M_i |\psi\rangle . \tag{2.16}$$

We impose the completeness condition on the operators M_i to ensure that the probabilities sum to unity:

$$\sum_i M_i^\dagger M_i = \mathbb{I}. \tag{2.17}$$

If the initial state is a density matrix, the probability of obtaining the outcome i can be calculated as

$$p(M_i) = \mathrm{tr}\Big\{ M_i^\dagger M_i \rho \Big\}. \tag{2.18}$$

For projective measurements such as $P_i = |e_i\rangle\langle e_i|$ we can write $M_i = P_i$. Since for a projection operator $P_i^\dagger = P_i$ with $P_i^2 = P_i$, we recover the simple Born rule, $p(P_i) = \mathrm{tr}\{P_i\rho\}$.

The projections give an orthogonal decomposition of identity and have the effect of a filter on the state. Only a specific component of the state is allowed to pass after measurement, which corresponds to the projection $P_i |\psi\rangle$. Such measurements formally correspond to projection-valued-measures (PVMs).

A more general type of measurement is given by positive-operator-valued-measures (POVMs) defined using a set of operators $\{\mathcal{O}_i = M_i^\dagger M_i\}$. The operators sum to the identity $\{\sum_i \mathcal{O}_i = \mathbb{I}\}$ and in general might not be projective or orthogonal. These operators generalize the idea of PVMs using a more general non-orthogonal decomposition of the identity operator.

2.2.1 Informational completeness

We now have an idea of how to measure quantum-mechanical properties of states. Specifically, if we wish to obtain the full density matrix for a d-dimensional quantum system, we seek a $d \times d$ density matrix defined by d^2-1 independent real numbers. Using von Neumann projective measurements we therefore need to perform repeated measurements on copies of ρ for a full quantum state tomography. The number of copies of ρ (or repeated measurements) required for tomography was known to be $O(d^4)$ [98, 99]. In 2014, [100] improved it to $O(d^3)$. Only in 2016, [101] showed that the optimal number of copies of ρ for full quantum state tomography was $O(d/\epsilon)$ to achieve an error ϵ in trace distance $|\rho' - \rho| \leq \epsilon$ with a high probability for the estimate ρ'.

To reveal the complete information about the state ρ, we can formulate a so-called informationally complete (IC) set of measurements. An informationally complete measurement on a quantum state is given by a POVM $\{\mathcal{O}_i\}$ that allows the computation of the expectation value of any arbitrary observable by completely specifying ρ. Such an IC-POVM consists of at least d^2 operators $\{\mathcal{O}_i\}$ that span the full Hilbert space of the quantum state.

As an example, consider an optical quantum state such as a quantized electromagnetic field in a cavity described using the Fock basis with a finite cutoff N_c. The heterodyne detection scheme can be used to measure the quadratures

$$\begin{aligned} \hat{x} &= \frac{1}{\sqrt{2}}(a^\dagger + a) \\ \hat{p} &= \frac{i}{\sqrt{2}}(a^\dagger - a). \end{aligned} \tag{2.19}$$

The field corresponding to the quantum state is first mixed with a strong, coherent local oscillator beam using beam splitters and then measured by two photodetectors. The photodetectors therefore play the role of the classical measurement apparatus and convert the quantized electromagnetic wave into an electric current. By subtracting the photocurrents and demodulating the result, the x and p quadrature values can be obtained.

Heterodyne detection can be seen as measuring the projection of the quantum state onto the coherent states, i.e., $\frac{1}{\pi}|\alpha\rangle\langle\alpha|$, and thereby producing the Husimi Q function. The real and imaginary components of α correspond

to the values for $\hat{x}$ and $\hat{p}$. A more general measurement for such optical states is the generalized Q function [102] given by measuring the Fock occupation probabilities of a displaced state,

$$Q_n^\beta = \mathrm{tr}\Big[|n\rangle\langle n|\, D(-\beta)\rho D^\dagger(-\beta)\Big], \tag{2.20}$$

where $|n\rangle$ is the Fock state with n photons, $D(\beta) = e^{\beta a^\dagger - \beta^* a}$ is the displacement operator, and $a(a^\dagger)$ is the bosonic annihilation (creation) operator of the electromagnetic mode.

The Husimi Q function is simply $(1/\pi)Q_0^\beta$. This generalized measurement could be implemented using a photodetector after applying a certain displacement to the state characterized by the term $\alpha = \frac{1}{\sqrt{2}}(x + ip)$.

In this setting, for IC, we require $N_c + 1$ different measurements with various α values [102, 103]. Each measurement reveals the occupation probability for N_c different Fock basis elements $|n\rangle$, thereby fixing the $O(N_c^2)$ real values needed for reconstructing ρ. Similarly, for a measurement of the projection on photon field quadratures $|x_\theta\rangle$ with θ determining the phase setting in homodyne detection, we need N_c different quadrature measurements with each quadrature discretizeable up to $2N_c - 1$ bins again leading to $O(N_c^2)$ real-valued data that is required for IC [103].

The design of informationally complete POVMs is not straightforward for arbitary quantum states and for higher dimensions. The class of symmetric, informationally complete POVMs (SIC-POVM) denotes the optimal IC-POVM with exactly d^2 elements [104]. There are exactly determined SIC-POVMs found by hand and computer algebra systems using supercomputers for dimensions as high as $d = 844$ [105]. For the case of optical quantum states, in [102] a numerical procedure is described to determine the $N_c + 1$ values of α that are optimal. A geometric interpretation is presented that can guide the choice of α values for a Schrödinger-cat state. Nevertheless, it is not straightforward to determine the best possible SIC-POVMs to completely reconstruct a quantum state and often we might use informationally over complete measurements [106]. Informationally overcomplete measurements are ICs which could have more than d^2 outcomes, e.g., measuring a set of qubits locally in all combinations of Pauli operators $(\mathbb{I} \pm \sigma_{x,y,z})/6$.

2.3 Noise: state preparation and measurement

Noise in an experiment refers to unwanted effects that can prevent obtaining the information of interest during measurement. In a classical setting, noise results from any physical process that influences the measurement apparatus, e.g., random fluctuations in the environment. A simple strategy to deal with such noise is to repeat the measurement and take averages. This strategy is based on assuming that the noise comes from independent random sources. With this assumption, we can use the central limit theorem to model the noise. The central limit theorem states that the addition of independent random variables leads to a (normalized) sum that forms a normal distribution (bell curve). Therefore we can model noise with an additive Gaussian term,

$$y = y_\mu + \mathcal{N}(0, \sigma), \tag{2.21}$$

where y_μ represents the average value and $\mathcal{N}(0, \sigma)$ denotes samples from a zero-mean Gaussian with standard deviation σ.

In a quantum measurement scheme, since the final outcome is read from a classical apparatus, noise can affect its outcome. Therefore we have to repeat the experiment to get a better estimate. However, due to the projective nature of the measurement, we will never have access to the same state after it is measured. Hence we need multiple copies of the state for repeating the measurement. Here is another potential source of noise that is more difficult to correct — state preparation and measurement noise (SPAM). We have to ensure that the process that creates the state to be measured is repeated exactly with no errors each time. Systematic errors in our instruments or decoherence of the quantum state might change the state itself $\rho \to \rho_{\text{noisy}}$. If we have an incorrect calibration of the measurement device we might have errors in the operators $\mathcal{O} \to \mathcal{O}_{\text{noisy}}$ and therefore measure incorrect information.

Apart from simple additive Gaussian errors and SPAM noise, there are multiplicative errors that can occur that become difficult to correct. In a linear time-invariant system, an output signal, y is given by the convolution of an input f with the impulse response of the system h,

$$y = f * h. \tag{2.22}$$

In quantum measurement schemes, linear detectors that amplify a signal therefore also add such a multiplicative noise to the signal given by a

convolution that depends on the state of the amplification channel [107]. Such type of noise is more challenging to correct and often requires very carefully crafted techniques to process the data.

In order to deal with different types of noise, we always require some regularization techniques or adding prior information to our learning algorithm. In many cases, knowing the background noise itself can help us subtract or filter it out. In the results presented in Paper II, we discuss the different types of noise and how they can be tackled. Now let us look at some of the learning models for quantum state learning and the costs associated with them in terms of the data required and computation.

2.4 Quantum state reconstruction

The reconstruction of quantum states from measurement data is a computationally demanding task. The implementation of a minimal set of measurements such as SIC-POVM or using mutually unbiased bases is difficult experimentally since they may involve non-local measurements and so informationally overcomplete measurements are used [106]. Informationally overcomplete measurements contain $O(d^2)$ measurement operators for a d-dimensional density matrix and therefore the cost associated with any numerical procedure becomes challenging. An 8-qubit quantum state would require measuring 10^6 operators, and each measurement must be further repeated for a number of times to reduce statistical errors.

Broadly speaking, reconstruction methods are based on linear inversion [108, 109], maximimum likelihood estimation [94, 110], or Bayesian techniques [111–113]. The algorithms used behind the scenes in these methods are usually optimization techniques such as least-squares regression, gradient descent, or semidefinite programming. In recent times, neural networks and ideas from machine learning have also been applied to state reconstruction with interesting results [30, 114–118]. We will focus on neural networks later in this thesis. First, let us discuss the different reconstruction methods in brief.

2.4.1 Reconstruction methods

In linear methods, the goal is to invert the linear equation relating measured data $\mathbf{d}$ to the density matrix ρ,

$$\mathbf{d} = A\rho_f, \tag{2.23}$$

where A denotes the sensing matrix that contains information about the measured operators $\{\mathcal{O}_i\}$ and ρ_f is the flattened density matrix. We can write this simple linear relation since the Born rule is linear and gives the elements for the data vector as $d_i = \mathrm{tr}\{\mathcal{O}_i\rho\}$. To solve Eq. (2.23), we can try inversion methods from linear algebra or apply least-squares minimization. The minimization has to be further constrained to recover a physical density matrix [94].

Maximum likelihood estimation (MLE) is an alternative to linear inversion which tries to estimate the best possible density matrix that is most likely to produce the observed data [94]. Maximum likelihood estimation also implements constraints on the density matrix to keep it physical. In MLE, we maximize the likelihood function given by,

$$\mathcal{L}(\rho) = \prod_i (\mathrm{tr}\{\mathcal{O}_i\rho\})^{m_i} \tag{2.24}$$

where m_i are the measured counts for the observable $\mathcal{O}_i$. In practice, it is more convenient to deal with the log-likelihood function that converts Eq. (2.25) to a sum over all observed data:

$$\mathcal{L}_{\log}(\rho) = \sum_i m_i (\mathrm{tr}\{\mathcal{O}_i\rho\}) \tag{2.25}$$

However, MLE is computationally expensive and sometimes could be slow to converge []. Therefore several modifications to standard MLE have been proposed to improve it.

In [110] a simple steepest-ascent method was used to maximize the log-likelihood function with an iterative algorithm. The density matrix was constrained to be positive and Hermitian within the iterative steps using a Cholesky decomposition. We will compare our neural-network based estimation methods to this iterative maximum likelihood estimation in Paper I and II.

The iterative maximum-likelihood method has a nice guarantee to converge due to the convex nature of the log-likelihood. However, there is no guarantee that every iterative step will lead to an increse in the likelihood as shown in [119] with a counter example. Therefore it might take a long time to converge. In [119], a diluted nonlinear iterative algorithm was proposed that allows a compromise between faster convergence and the guarantee on likelihood increase. Still, due to enforcement of quantum-mechanical constraints on the density matrix and implementing a minimization in the space of unconstrained operators, convergence is slow [120].

In [121], by assuming Gaussian statistics for the measurement noise, it was possible to develop a fast MLE method for state reconstruction. The likelihood function is now changed to

$$\mathcal{L}(\rho) \propto \prod_i \exp[-(d_i - \mathrm{tr}\{\mathcal{O}_i\rho\})]^2, \tag{2.26}$$

where we denote d_i to be the average value for the measurement outcome of operator $\mathcal{O}_i$ and we have ignored constant terms. The log-likelihood function now has a form similar to least-squares regression,

$$\mathcal{L}_{log}(\rho) \propto \sum_i -[d_i - \mathrm{tr}\{\mathcal{O}_i\rho\}]^2. \tag{2.27}$$

Therefore maximizing the log-likelihood for a Gaussian noise prior could be related to least-squares minimization.

Further improvements to MLE came with the application of projected gradient descent technique in [120]. Here, the authors use an accelerated gradient descent algorithm with a "momentum" that helped to achieve faster convergence. To implement the constraint of positive semidefiniteness and unit trace, at each step of the gradient-based optimization, the density matrix is projected to the space of valid quantum states. This is achieved by discarding negative eigenvalues of the estimated density matrix and reconstructing it back from the eigenvectors of the current estimate. With this adaptation, it was possible to perform the state reconstruction of an 8-qubit state within a minute and just hundreds of iterative steps. Although, maximum likelihood based reconstruction is widely used for state reconstruction, some authors have pointed out its flaws against linear regression techniques [108] and even questioned its admissibility [112, 122].

Bayesian quantum state estimation is an alternative to MLE that does not just give a single most likely density matrix but defines a probability distribution, $p(\rho)$ for the density matrix, and uses Bayes rule to update the posterior probability after observing some data,

$$p(\rho|\mathbf{d}) \propto \mathcal{L}(\rho)\pi(\rho). \tag{2.28}$$

where $\pi(\rho)$ is a prior for the distribution of density matrices. It is possible to assume that the density matrix is parameterized by some vector $\mathbf{x}$ and then define the prior and posterior over $\mathbf{x}$ that ensures physical density matrices. Then, the expectation value for any function on ρ can be computed as

$$\langle\phi(\rho)\rangle = \int d\mathbf{x}\pi(\mathbf{x})\phi(\rho(\mathbf{x})). \tag{2.29}$$

However, the challenge in Bayesian methods is computing the posterior or evaluating integrals of the form Eq. (2.29), which is usually approximated via Markov Chain Monte Carlo (MCMC) methods.

2.4.2 Weaker learning models

The computational cost associated with full quantum state tomography can be reduced by exploiting prior knowledge and simplifying the problem. In compressed sensing methods [98], it is assumed that the states that are reconstructed are of low rank, which greatly reduces the need for measurement data. If the rank of the state is r, compressed sensing methods allow the computation of a $d-$dimensional density matrix using $O(rd\log^2 d)$ measurements.

Permutationally invariant quantum tomography exploits permutational symmetry of the state to find a measurement strategy to reduce experimental effort [123]. Similarly, matrix-product-state [96, 97] or tensor-network tomography [115, 124] are based on a prior assumption on the entanglement properties of the state.

Apart from such prior assumptions, there have been different learning models that are weaker than full quantum state tomography but still are practically relevant [125]. In some of these learning models, we may only seek interesting properties of the quantum state without reconstruction the full state [126]. Some of the methods allow for a direct estimation of fidelity of the state to a target state [127] or computation of expectation values of observables directly without full state tomography [128, 129].

In probably approximately correct (PAC) learning [130], we assume a concept class of known quantum states $\mathcal{C}$ and an unknown distribution for measurement settings for two-outcome measurements $\mathcal{D} : \mathcal{O} \to [0, 1]$. The data is collected according to the choice of measurements given by sampling from $\mathcal{O}_i \sim \mathcal{D}$ as $\{\mathcal{O}_i, \mathrm{tr}\{\mathcal{O}_i\rho\}\}$. The PAC-learning framework then tries to estimate a density matrix σ that approximates ρ with its quality determined by two approximation parameters α and ζ such that the error between the observed data and calculated probabilties is low with a high probability,

$$p[|\,\mathrm{tr}\{\mathcal{O}'\sigma\} - \mathrm{tr}\{\mathcal{O}'\rho\}]| \leq \zeta] \leq (1-\alpha). \tag{2.30}$$

Intuitively, we are trying to learn an approximation to the state that can correctly tell us about statistics of measurements from the distribution $\mathcal{D}$. In this framework, it was shown that the number of examples (copies of

the density matrix and therefore measurements) that are needed can be linear in the number of qubits n [125, 130].

Some of the assumptions of PAC learning could be difficult to justify, e.g., that the measurements we are interested in are drawn from the same distribution $\mathcal{D}$ that we have measured experimentally [125]. Therefore PAC learning might not be valid with changing conditions in the environment. Further, it might only be possible to obtain sequential information from an experiment and not a tensor of identical states.

The online learning model [131] tackles these issues by maintaining at each instance an assumption of the density matrix σ and trying to maintain a low error rate

$$|\operatorname{tr}\{\sigma\mathcal{O}_i\} - y_i| \leq \epsilon, \tag{2.31}$$

where $y_i = \operatorname{tr}\{\mathcal{O}_i\rho\}$. The goal of online tomography is to minimine the number of mistakes, i.e., violations of Eq. (2.31) after which for all subsequent measurements we have a low error between predicted values and data. In this learning model, it could be shown that the number of errors, i.e., required measurements could be related to the sequential fat-shattering dimension [125, 131] of the class of quantum states $\mathcal{C}$ which could be much smaller that exponential in certain cases than exponential in the size of the quantum state [131]. The fat-shattering dimension is a combinatorial parameter that defines the richness of the class of functions $\mathcal{C}$ [132].

Finally, in shadow tomography [128, 129], the idea is to predict properties of the state or target functions on the state instead of estimating the full state. A classical description for the state is learned by repeatedly collecting results of local measurements (such as measuring the qubits) after applying random unitary transformations on the full state. This classical description is called the classical shadow. Then any target function of the state ρ is directly computed from a median of mean values prediction that uses a certain splitting of the classical shadows to give k different estimates for the density matrix [129]. In the original idea for shadow tomography, it was shown that given m measurements $\mathcal{O}_1, \mathcal{O}_2, ..., \mathcal{O}_m$, we can estimate $\operatorname{tr}\{\rho\mathcal{O}_i\}$ for all $i \in [0, m]$ with only $O(n, \log m)$ copies of the state ρ. [125, 128]

In the papers that this thesis is based on, we have proposed a new method for quantum state reconstruction following ideas from machine learning. Our method uses neural networks to predict a density matrix from measurement data and can be flexibly adapted to handle different scenarious for noise. Our techniques draw from some of the ideas discussed

in this section such as constraining the density matrix to be physical, using adaptive gradient-based optimization, and handling different cases of noise and likelihood functions. However, before moving to our proposed techniques, we first need to discuss the use of neural networks in machine learning in general.

Chapter 3

Machine learning with neural networks

Modern digital computers can be used to carry out a variety of tasks and therefore find applications in every aspect of human life. Digital computers are logically equivalent to Turing machines proposed by Alan Turing in 1936. A Turing machine defines an abstract computational model that manipulates symbols on a tape according to some instructions and therefore can execute an algorithm. An algorithm is a well-defined set of instructions to complete a task. These tasks could be anything — solving mathematical problems, controlling a robot, communicating information or simply listening to music. Therefore, the design of algorithms to carry out tasks using a computer is a major area of study in computer science.

However, for many types of tasks faced in the real world, straightforward algorithms cannot be applied. Complex tasks such as translating languages, playing games, recognizing an image, driving a car, or allowing a robot to navigate the world are not easy to solve using a simple set of instructions. Still, humans routinely solve such complex problems by learning from experience.

A baby can learn language, movement, and solve a plethora of tasks as it grows up. Even though as a human, we can learn to perform complicated tasks, are we following a simple set of instructions to accomplish them? The key is that we learn and somehow build the ability to solve a problem with experience in addition to following instructions. Is it possible then to also design a machine than can similarly learn from experience using algorithms that emulate intelligent behaviour?

One of Turing's lesser known contributions addresses these questions with his report on "Intelligent Machinery" in 1948 [133]. Turing defined "unorganized machines" with examples of an early form of neural networks. According to Turing, appropiate education could train such machines to perform complex tasks such as mathematics, learning or translating languages, and playing various card and board games. Turing even suggested the similarity between such machines and the infant human cortex. More remarkable is Turing's consideration of some of the principles involved in building such a brain without a body, e.g., proposing training techniques based on genetic evolution. Turing even had the idea of giving such a learning machine wheels, arms and camera eyes so that it was mobile enough to find out things for itself in the world. This proposal was met with outcries of "Turing is going to infest the countryside with a robot which will live on twigs and scrap iron!". [134]

In subsequent years, this idea of a network of units that can be trained to solve complex tasks transformed into modern neural networks. Further, taking inspiration from neural networks, differential programming [135, 136] emerged as a more general framework to write parameterized computer programs that can be optimized and changed dynamically. Such programs can include loops and logical conditionals but can also be differentiated using automatic differentiation techniques. Therefore these automatic-differentiable programs could be optimized by minimizing some loss function using data to complete a task using gradient descent. A quote by researcher Andrej Karpathy is a humorous reminder of the potential for writing better programs using differential programming — "Gradient descent can write code better than you [137]."

This chapter is focused on neural networks, differential programming and a brief description of probabilistic models for discriminative and generative tasks. We will use these techniques in Chapter 4 for recognizing classes of quantum states and performing quantum state tomography.

3.1 Neural networks

An artificial neuron is a biologically inspired model that defines the input-output relationship between variables x and y as the non-linear model

$$y = \sigma(\theta_1 x + \theta_0), \tag{3.1}$$

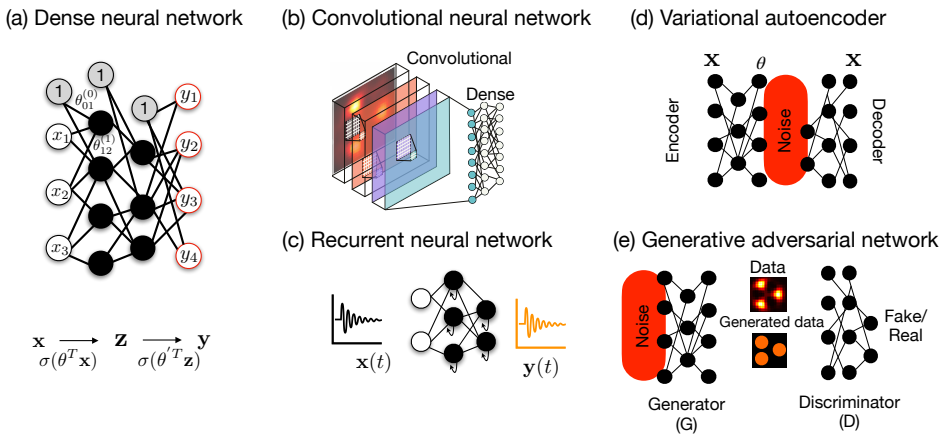

Figure 3.1: (a) A feed-forward, densely connected neural network with multiple layers applies sequences of linear and non-linear transformations to define a learnable function. (b) Convolutional neural networks are a modified architecture inspired by the human visual cortex. In each convolutional layer, spatial filters extract features such as edges to build higher abstractions and identify patterns in the input. A fully connected dense neural network uses these features to perform a labelling of the input data. (c) Recurrent neural networks are another architecture that can work with sequential data and have self-connections to emulate memory effects. (d) The variational autoencoder architecture can learn data distributions using an encoder-decoder formulation and sample new data similar to a given dataset. (e) Generative adversial neural networks use two competing neural networks to learn a data distribution. The generator tries to generate new data starting from randomly sampled noise vectors and the discriminator acts as learnable loss that evaluates the generator's performance and guides it to generate realistic data samples.

where θ_i are called weights with θ_0 sometimes called a bias term. The function σ denotes a nonlinear function called the activation function. The simple model in Eq. (3.1) follows from the idea of perceptrons [] developed by Frank Rosenblatt and is one of the most widely used today. Other formulations of artificial neurons such as spiking neurons [] are also actively researched and pursued.

In Eq. (3.1), if the input is a vector $\mathbf{x} = \{1, x_1, x_2, ..., x_n\}$, we can write the output of the neuron as

$$y = \sigma(\theta^T \mathbf{x}), \tag{3.2}$$

where $\theta = \{\theta_0, \theta_1, \theta_2, ..., \theta_n\}$ is a weight vector. We appended a unit term in the input vector for a concise notation that includes the bias terms using a dot product between the weight and the input vector.

Now, let us combine a set of such neurons in layers and construct a feed-forward neural network, see Fig. 3.1. Let the layers be indexed using a superscript notation $\mathbf{x}^{(l)}$, where l = 0,1,2,... and the neurons be indexed by subscripts $x_i^{(l)}$ with $x_0^{(l)} = 1$ for all layers to include bias terms. The intermediate outputs of neurons at each layer z_j^l of the neural network can now be written as a function of the inputs from the previous layer and a set of weights $\theta_{ij}^{(l)}$ that connect neurons $i \to j$:

$$z_j^{(l)} = \sigma\left(\sum_i \theta_{ij}^{(l)} x_i^{(l-1)}\right). \tag{3.3}$$

The simplest network is one with a single hidden layer ($l = 1$) that takes inputs from the input layer $l = 0$ and gives the single output $y_1^{(l=2)}$:

$$y_1^{(2)} = \sum_j \theta_{j1}^{(1)} z_j^{(1)} = \sum_j \theta_{j1}^{(1)} \sigma\left(\sum_i \theta_{ij}^{(0)} x_i^{(0)}\right), \tag{3.4}$$

where we have not applied the activation on the output layer.

3.1.1 Universal function approximation by neural networks

Neural networks with at least a single hidden layer can be shown to have the capacity to approximate arbitrary functions. This property of neural networks forms the theoretical basis for using them in approximating relationships between different type of data. In this thesis, we use neural

networks as a map between measurement data and a theoretical description of the state that generated the data. We consider neural networks as a classifier for the data and also as a generating function that outputs the density matrix with noisy data as input. The universal approximation theorems about neural networks therefore give a theoretical basis to using neural networks as maps between data and density matrices, or for classification.

In [88], it was shown that finite sums of the form in Eq. (3.4) can be a universal function approximator for real-valued continuous functions f in the hypercube

$$I_n \coloneqq [0,1]^n = \{\mathbf{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n | x_i \in [0,1]\}. \tag{3.5}$$

The finite sum can be written concisely as $y(\mathbf{x}) = \sum_j w_j \sigma(\mathbf{w}_j^T \mathbf{x})$ with weights of the output layer as w_j and weight vectors from the input to the hidden layer as $\mathbf{w}_j$. This finite sum approximates f arbitarily close for $\epsilon > 0$,

$$|y(\mathbf{x}) - f(\mathbf{x})| < \epsilon. \tag{3.6}$$

In the proof given in [88], the activation function σ needs to be a continuous sigmoidal function such that

$$\sigma(t) \to \begin{cases} \to 1, & \text{as } t \to +\inf \\ \to 0, & \text{as } t \to -\inf . \end{cases}$$

It is straightforward to extend this for other types of activation functions such as trigonometric or exponential using the Stone-Weierstrass theorem [138]. The Stone-Weierstrass theorem guarantees that for a continuous real-valued function f that is defined on a bounded real interval $[a, b]$, there exists a polynomial p such that for all x in $[a, b]$, $|f(x) - p(x)| < \epsilon$ for an error ϵ.

In [87], the Stone-Weirstrass theorem was used to show that arbitrary bounded non-constant activation functions (which might even be discontinuous) are sufficient for universal approximation by standard multi-layer feedforward neural networks. A later work [139] showed the more general result that "a standard multilayer feedforward network with a locally bounded piecewise continuous activation function can approximate any continuous function to any degree of accuracy if and only if the network's activation function is not a polynomial." Thereafter, it is straightforward to extend the universal approximation idea to functions for multidimensional inputs and output.

In recent times, there is an interest in understanding the expressive power of deep neural networks with multiple hidden layers. The universal approximation theorems for shallow, single-hidden layer neural networks only provide a mathematical guarantee on the expressivity to approximate arbitrary functions. But in practice, deep architectures perform much better [29, 140] and their expressivity has only been explored after their success.

There are demonstrations of functions that can be expressed more efficiently with deeper architectures than a single-hidden-layer network [141, 142]. In some conditions, deeper convolutional neural networks perform much better function approximation [143]. Some examples can also be constructed where a single-hidden-layer network cannot realize a three-hidden-layer network [144] or a deep convolutional network cannot be realized with a shallow network unless the number of nodes in the hidden layer grows exponentially [145]. In [140], a simple demonstration showed that multiplying n variables required 2^n neurons required using a single hidden layer, but $4n$ neurons using a deeper network with $\log_2 n$ layers.

Besides mathematical arguments, physics could also play a role in explaining why deep learning works well in practice. The data generation process, the forward model that links simple parameters to complex data, could be seen as a sequence of simple hierarchical steps. The structure of deep neural networks might be more suited to reverse these steps with each layer distilling information necessary only for the final output. [140]

The arguments presented so far demonstrate the universal approximation capabilities of neural networks and the possible advantages associated with deeper architectures rather than shallow networks. But training the network for a particular task is not easy. Modern deep neural networks might have up to a billion weights [40] that need to be learned. In 1986, a seminal paper [146] showed how to train these neural networks using the backpropagation algorithm. This simple algorithm, that is based on the chain-rule for differentiation, could be used to train complex neural-network architectures which is one of the main reasons for their recent success and applicability.

3.1.2 Training neural networks with backpropagation

Backpropagation uses the chain rule along to compute gradients of complicated functions. Along with gradient-descent, backpropagation is used to find the minima of a loss function such as the mean squared error between

a neural network's outputs $f(\mathbf{x};\theta)$ and a target variable:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{i=N}(f(\mathbf{x}_i;\theta) - y_i)^2.$$

We iteratively update the parameters θ in the opposite direction of the gradient of the loss function $\nabla_\theta\mathcal{L}(\theta)$ with respect to the model weights. A simple step of the gradient descent algorithm is

$$\theta(t+1) = \theta(t) - \eta\frac{\partial\mathcal{L}}{\partial\theta}. \tag{3.7}$$

Here, η represents the "learning rate" or step size. There can be thousands of weights which are updated at each iteration during training. At each step the gradient $\frac{\partial\mathcal{L}}{\partial\theta}$ has to be estimated for the all the weights in the network. For a complicated nonlinear function such as a deep neural network, this gradient calculation presents a challenge.

But it is possible to determine the gradients by "back propagating" the errors from the ouput layer to the input in a reverse direction. We can understand backpropagation with four simple fundamental equations which are easy to derive [147]. We show the crucial step now using the intermediate outputs $z_j^{(l)}$ and an error term defined as the gradient of the loss function with respect to the intermediate output,

$$\delta_j^{(l)} = \frac{\partial\mathcal{L}}{\partial z_j^l}. \tag{3.8}$$

If the final output layer is denoted by L, the error for the mean-squared loss is simply

$$\delta_j^L \propto (\sigma(z_L) - y) \odot \sigma^{'}(z_L) \tag{3.9}$$

where $\odot$ denotes element-wise multiplication or the Hadamard operator. Now we need to compute the errors δ_j^l as a function of the errors in the next layer δ_k^{l+1} which is given by simply applying the chain rule,

$$\delta_j^l = \frac{\partial\mathcal{L}}{\partial z_j^l} = \sum_k \frac{\partial\mathcal{L}}{\partial z_k^{l+1}}\frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l}\delta_k^{l+1}. \tag{3.10}$$

Since the intermediate output is given by

$$z_k^{l+1} = \sum_j \theta_{kj}^{l+1}\sigma(z_j^l), \tag{3.11}$$

the gradient term in the above equation is simply

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = \theta_{kj}^{l+1} \sigma'(z_j^l). \tag{3.12}$$

Therefore, the error terms in each layer are related to the errors in the next layer and the weights as

$$\delta_j^l = \sum_k \theta_{kj}^{l+1} \delta_k^{l+1} \sigma'(z_j^l). \tag{3.13}$$

Once we compute all the intermediate outputs in a forward run of the neural network and determined the errors δ_j^l, the gradients with respect to the weights can be simply computed by

$$\frac{\partial \mathcal{L}}{\partial \theta_{jk}^l} = \sigma(z_k^{l-1}) \delta_j^l. \tag{3.14}$$

Backpropagation only computes the gradients with respect to to a single training instance, but since we consider a loss function that can be averaged over different training examples, the total value of the gradients can similarly be computed by taking the average of the gradients for each training instance.

Once we have the gradients, other modifications to the actual update rule are possible, leading to variants such as batch gradient descent, stochastic gradient descent, momentum-based update rules or techniques such as Adam, that adaptively change the learning rate for individual weights. [148]

We have discussed how a neural network can be a powerful approximation for discovering the relationship between data spaces and how to train such large models using gradient descent. The crucial step in training is solved using backpropagation for computing gradients of complex functions. Backpropagation is a subset of the general method of automatic differentiation which allows writing computer code that can learn to change itself using differential programming.

3.2 Differential programming

The idea of differential programming is important for understanding how we can modify neural networks to incorporate quantum mechanics and still train them with deep-learning tools. Differential programming is

based on the ability to automatically differentiate a complex computer program. Automatic differentiation is different from symbolically taking the gradient of a function using the chain-rule. Even for a modestly complex function of many variables, such a symbolic differentiation formula would cover several pages. Automatic differentiation is also not the same as numerical computation of gradients using a finite difference rule. It allows the calculation of exact gradients in a constant time [136, 149].

Phil Wolfe made an observation regarding how difficult it is to compute the gradient of a scalar function with respect to n terms and the cost of computing the function itself [149],

> "If care is taken in handling quantities which are common to the function and derivatives, the ratio [between cost of gradient evaluations and evaluating the function] is usually 1.5, not $n+1$."

Andreas Griewank [149] showed that this observation is a theorem with the upper bound on the ratio as 5. Therefore automatic differentiation makes it possible to compute gradients of a complicated function with the same effort as evaluating the function.

Two modes exist for automatic differentiation — forward and reverse, where the reverse mode is similar to backpropagation. The key is to break down the computation of a complicated function into a sequence of simpler elementary operations such that the chain-rule can be exploited for calculating gradients.

Reverse mode differentiation has an advantage for functions with fewer outputs than inputs, i.e, $\mathbb{R}^n \to \mathbb{R}^m$ with $m \ll n$. However, it needs to store the intermediate outputs in memory and record how different intermediate terms are connected in a computational graph describing the function. In the case where the ouput is a single scalar value, such that we differentiate a function $f : \mathbb{R}^n \to \mathbb{R}$, reverse-mode autodifferentiation can evaluate all gradients in a single pass while the forward mode would require n evaluations. [136]

In the opposite situation, where $f : \mathbb{R} \to \mathbb{R}^n$, forward mode only requires one forward pass. In our works, we only use backpropagation or reverse-mode autodifferentiation but it is worthwhile to know that for other arbitrary learning problems, a mix of approaches can be considered. Extending the automatic differentiation technique to arbitrary computer programs leads to differential programming.

Differential programming is more than just algorithmically differentiating continuous-variable functions. We wish to construct learnable computer

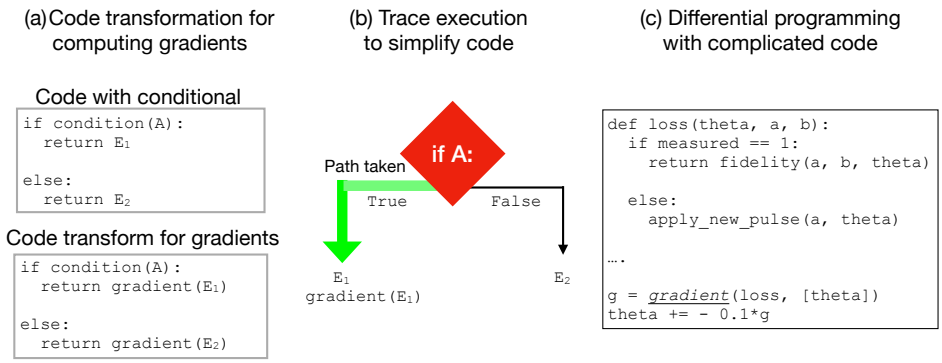

Figure 3.2: Differential programming requires automatic differentiation of computer programs. (a) Code transformations produce a code with similar conditionals as the input program for the gradients. (b) Execution-trace-based automatic differentiation uses a trace of the program that does not contain the conditionals since at execution, a computer code is a sequence of well-defined steps. (c) Differential programming could enable writing computer programs that can be treated in a similar way to neural networks and learned by training on data.

programs that can be trained using gradient descent. Such computer programs might consider logical statements such as if conditions and loops. Calculus does not provide a straightforward recipe to take gradients of conditional operations. However, new frameworks and languages have been proposed to resolve this issue and allow computer programs to be differentiable and thus learnable using training.

Two approaches to deal with more complicated computer programs are using code transformation and using evaluation traces [135]. Code transformations produce derivatives using similar conditionals as the source code, and therefore the code to take the derivative of an if statement would also be an if statement, see Fig. 3.2.

The other approach relies on the fact that during execution, a computer code will have a fixed path that would evaluate the conditions and therefore provide a sequence of elementary steps to compute an output without the conditional. This execution trace can be differentiated using automatic differentiation, see Fig. 3.2.

A number of softwares and differential programming languages are available today that can be used for designing more general computer programs that are trainable. TensorFlow [150], PyTorch [151], Julia-based [152] and Jax [153] are some examples. Differential programming can also be

applied to quantum programs such as variational circuits and tools such as PennyLane [154] make it easy to compose such programs.

For our work, we have used the ability of differential programming provided by TensorFlow to define custom operations in our models. Using differential programming therefore allows us to do more than simply writing a neural-network model, it enables optimizing a complex interactive model and we exploit it to add quantum-mechanical priors to our models. Now we are ready to discuss the different types of models that can be constructed using neural networks and put into context for different tasks. In the next section, we focus on building discriminative and generative models using neural networks.

3.3 Discriminative and generative models

We have seen how quantum state characterization entails learning the description of a quantum system from data. We could either learn the full state density matrix or ask for certain properties of the state. The universal function approximation capabilities of neural networks along with differential programming allows us to construct models that can map data to our target objective, and train them. We should now discuss the ideas behind discriminative and generative models that can be used for state classification or reconstructing the density matrix of the state.

The concepts of discriminative and generative models can be best understood by considering the probability distributions $p(y|x)$ and $p(x, y)$ for inputs x and outputs y. If x is an image, and y represents a label for the image, discriminative models attempt to build a model to compute the conditional probability $p(y|x)$ of assigning a label y to an instance of data x. A generative model, on the other hand, tries to learn the underlying data distribution, $p(x, y)$, so new data can be generated from it. Generative models can then generate random new instances of (x, y) or perhaps targetted instances of y given by the conditional $p(x|y)$.

In real-world examples, the data distributions can be complex and intractable. Consider the distribution of pixel-values in a 100×100 grid that generates grayscale images of human faces by varying each pixel value in the interval $[0, 256]$. There are more than $256^{10,000} \sim 10^{20000}$ possible images and it is a daunting task to write a multi-dimensional probability density function for the pixels that define human faces. The determination of such a probability density is even more difficult if you consider the

number of samples of human faces that we usually have in our training in comparison to the total possibilities. Even if we consider every single human being that walked on Earth and approximate it with 100 billion people, the ratio between all the possibilities and our data is extremely large (10^{20000} vs. 10^{11}). In real experiments to generate human faces, usually millions of images ($\sim 10^6$) are considered.

Yet there are neural-network-based generative modelling techniques that can learn to sample realistic human faces. We will discuss one such generative model called generative adversarial networks for learning the quantum state density matrix. This is the focus of this thesis, but first we discuss other concepts and ideas for generative models to get a perspective on why the generative adversarial framework could have advantages over the other methods.

3.3.1 Restricted Boltzmann machines

Restricted Boltzmann machines (RBMs) are an early type generative models that work slightly differently than the standard feed-forward neural-network architecture [155]. They are interesting because RBMs are currently used for quantum state tomography using the neural-network-quantum-states ansatz [30]. An RBM consists of visible and hidden units ($\mathbf{v} = \{v_1, v_2, ..., v_k\}, \mathbf{h} = \{h_1, h_2, ..., h_j\}$) which are restricted to give stochastic binary outputs $\{0, 1\}$. The stochasticity comes from defining the rule for updating the units. The value of a node h_j is set to 1 if the probability

$$p(h_j = 1|\mathbf{v}) = g\left(\theta_0 j + \sum_i \theta_{ij} v_i\right) \tag{3.15}$$

is greater than a random number sampled from a uniform distribution between 0 and 1. Here g is the activation, and the parameters θ_{ij} determine the interaction between v_i and h_j. The visible units $\mathbf{v}$ are updated similarly based on the state of the hidden units. The states of the visible units of the RBM converge to a Boltzmann distribution

$$p(\mathbf{v}; \theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h}; \theta)}, \tag{3.16}$$

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

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i \in \text{visible}} \theta_{0i} v_i - \sum_{j \in \text{hidden}} \theta_{0j} h_j - \sum_{i,j} v_i h_j \theta_{ij}. \tag{3.17}$$

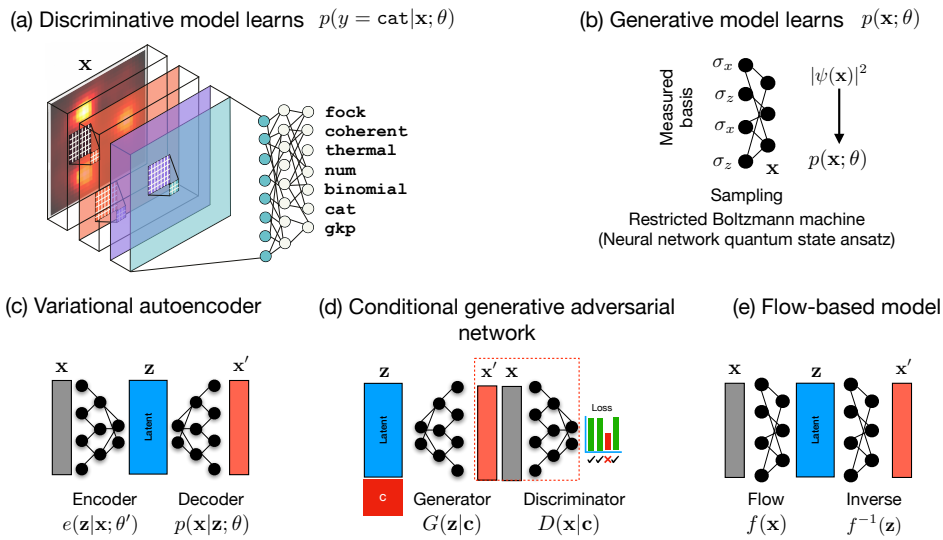

Figure 3.3: Discriminative and generative models using deep neural networks. (a) Convolutional neural networks can be used to estimate the probability of a data belonging to particular class, $p(y|\mathbf{x})$. (b) Generative models learn the data distribution and can sample new data. Restricted Boltzmann Machines use a stochastic neural network that models a parameterized Boltzmann distribution to approximate a data distribution. (c) Variational autoencoders learn a map from a latent space (noise) to the data using an encoder-decoder architecture. (d) Generative adversarial networks directly learn the data generation map from latent noise using a generator. A second neural network, the discriminator acts as a learnable loss function that trains the generative process. (e) Flow-based models estimate the probability density of the data using a sequence of invertible maps that can be easily differentiated and directly provide the probability density function for the data distribution.

Therefore RBMs use the parameterized Boltzmann distribution to approximate the underlying probability distribution of a set of data vectors $\mathbf{v}_k$. The training step optimizes a measure of statistical divergence, e.g., the Kullback–Leibler (KL) divergence between the sampled vectors from the RBM and the data. After training, we seek parameters that give a high probability for a data point $\mathbf{d}_k$ set as the visible units $p(\mathbf{d}_k;\theta)$. New data points can then be generated by sampling the RBM using some sampling method. For continuous-valued data we further need to convert them to a binary encoding or use variants like Gaussian-Bernoulli RBMs [156].

Training of RBMs is not straightforward as simple backpropagation would not work due to the stochasticity of the outputs in neurons. Techniques such as contrastive divergence [155, 157] that are based on a sampling procedure and a gradient-based update of the weights, enabled training of RBMs. However, the difficulties remain due to the stochastic nature of these models that requires sampling. More straightforward generative modelling techniques such as variational autoencoders, generative adversarial networks or flow-based models are easier to work with than RBMs. Such techniques allow back-propagation based learning, and tackle the sampling aspect using a latent (noise) space, and mapping randomly sampled elements $\mathbf{z}$ from this latent space to a data point using a neural network as the deterministic mapping function:

$$f : \mathbf{z} \rightarrow \mathbf{x}. \tag{3.18}$$

Although both variational autoencoders and generative adversarial networks use a parameterized neural network to approximate the data-generating function f, they are very different in how this function is learned.

3.3.2 Variational autoencoders

Variational autoencoders (VAEs) work by making a set of approximations to the likelihood of a data point and the posterior probability for a random latent vector given a data point, $p_\theta(\mathbf{z}|\mathbf{x})$. The integral of the marginal likelihood is given by [158, 159]

$$p_\theta(\mathbf{x}) = \int p(\mathbf{x},\mathbf{z})d\mathbf{z} = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} \tag{3.19}$$

and is intractable since it is not possible to sum all possible values of random noise vectors $\mathbf{z}$. In VAEs, a decoder neural network is used to approximate

the map from the latent space to the data parameterized by θ as $p_\theta(\mathbf{x}|\mathbf{z})$ which approximates the likelihood. The posterior $p_\theta(\mathbf{z}|\mathbf{x})$ is given by

$$p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{x})} = \frac{p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p_\theta(\mathbf{x})}, \tag{3.20}$$

which is also difficult to compute and is parameterized by an encoder $q_\phi(\mathbf{z}|\mathbf{x})$.

In order to train the encoder and decoder networks, the KL divergence between the generated data and true data is minimized. However, applying backpropagation is not possible directly due to sampling of the latent noise vectors $\mathbf{z}$. This was solved by the reparameterization trick making backpropagation-based training work on VAEs [158]. The reparameterization trick expresses the random variable $\mathbf{z}$ as the output of $\mathbf{z} = e_\phi(\mathbf{z}, \epsilon)$ with an independent random variable ϵ thereby allowing autodifferentiation to work.

In image-generation tasks, VAEs are susceptible to generate samples that might be blurry and not very sharp with varying opinions as to why this happens. The use of the KL-divergence loss, simple Gaussian priors or small latent space dimension are some possible reasons that may lead to VAEs not being to able to capture the data distribution well [160]. In this regard, generative adversarial networks perform better and produce sharper outputs in image generation tasks [161, 162].

3.3.3 Generative adversarial networks

Generative adversarial networks (GANs) [163] take a different approach to learning the map between a latent space and data. Here, a generator network, G (similar to the decoder in VAEs) is trained by letting a second neural network, the discriminator D, evaluate its outputs. The discriminator considers the input $\mathbf{x}'$ and outputs a probability $D(\mathbf{x}';\theta_D)$ for the data point $\mathbf{x}'$ to belong to the data distribution. The generator and discriminator network can now be trained in an alternating sequence until the generator learns to produce data which is similar to the true data distribution and the discriminator can no longer distinguish generated data from the true data.

The conditional generative adversarial network architecture further improves upon the standard idea of adversarial learning by allowing to shape the data generation process with the mapping

$$f : \mathbf{z}|\mathbf{y} \to \mathbf{x}, \tag{3.21}$$

where $\mathbf{y}$ is a conditioning variable such as a label. Therefore, conditional generative adversarial networks can be used to model complex conditional maps between different data. The use of the discriminator network to evaluate the performance of the generator performance allows possible learning of a non-trivial loss metric. However, since generative adversarial networks do not explicitly model the data distribution and rather learn it implicitly, there are doubts whether they are truly representative of the data distribution [164].

We have seen that both VAEs and GANs do not explicitly learn the data distrubution $p(\mathbf{x})$ and use a latent noise space to generate samples of data. Variational autoencoders make an approximation of a data distribution using a mixture model and it is not clear if generative adversarial networks actually learn the data distribution perfectly [164]. Another generative modeling approach called flow-based models solves this issue by directly estimate the probability density for data.

3.3.4 Flow-based generative models

Flow-based models use the idea of the change-of-variable theorem from probability theory [165]. A new random variable which we consider as our data $\mathbf{x}$ can be constructed from another easy-to-sample distribution $p(\mathbf{z})$ using a one-to-one mapping $\mathbf{x} = f(\mathbf{z})$. The inverse of this operation is given by $\mathbf{z} = f^{-1}(\mathbf{x})$ and the probability density for the generated data can be written as

$$p(\mathbf{x}) = p(\mathbf{z}) \left| \det \frac{d\mathbf{z}}{d\mathbf{x}} \right| = p(f^{-1}(\mathbf{x})) \left| \det \frac{df^{-1}}{d\mathbf{x}} \right|, \tag{3.22}$$

where det denotes the determinant. Now, if we consider a series of such invertible functions, we can convert the simple latent noise $\mathbf{z}$ to a much more complex one sequentially by applying nonlinear bijective functions

$$f_N \circ f_{N-1} \circ ... \circ f_1 : \mathbf{z} \to \mathbf{x}. \tag{3.23}$$

It can be shown that the composition of such N bijective function is invertible such that inverse is given by

$$f_1^{-1} \circ f_2^{-1} \circ ... \circ f_N^{-1} : \mathbf{x} \to \mathbf{z}. \tag{3.24}$$

Now, having a set of invertible and differentiable functions that map an easy distribution to a complex one makes the likelihood function for the data tractable. The random variable traverses a path $\mathbf{z}_i = f_i(\mathbf{z}_{i-1})$

which is called the flow. The chain of the intermediate distributions p_i is called a normalizing flow due to movement from a complicated data distribution to a more regular or "normal" form. Different normalizing-flow models can be constructed by the choice of the functions f which need to satisfy two conditions — easy invertibility and computation of the Jacobian determinant [165].

Normalizing flows have the benefit that they provide us with the probability density for the data. Therefore they make it possible to sample new data points, predict the probability for a possible data, fill in missing data and more. They also connect to the narrative of [140] where deep neural networks provide a good tool to model the hierarchical transformation of data using simpler steps to generate complex data. We will not use normalizing flows for this thesis but see it as an interesting alternative to our approach for learning the density matrix of quantum states.

Now that we have reviewed different techniques for discriminative and generative modeling we can apply them to quantum state characterization. In the most basic setting, we wish to identify properties of the quantum state which resembles a discriminative task. For complete characterization, we would like to obtain the full wave function or density matrix of the state. In the next chapter, we will introduce the necessary concepts from quantum mechanics to set the stage for the results and show how deep neural networks can be applied for characterizing quantum systems.

Chapter 4

Neural networks for characterizing quantum systems

Neural networks are one of the most successful and widely studied machine-learning techniques today. Therefore, it is no surprise that physicists have been exploring the use of neural networks to tackle problems in quantum physics. Early works in 2001 on using unsupervised neural networks to solve the Schrödinger equation showed how a trained neural network was better than other numerical approaches to find the wave function and capture its features [166]. More recently, the idea of using Restricted Boltzmann machines as a variational ansatz for a many-body quantum state was proposed in [30], which led to many new results that relied on the efficient antsatz [115]. There were also many other proposals to use neural networks for quantum state reconstruction using architectures such as variational autoencoders, recurrent neural networks or simple feed-forward neural networks which showed promising results [97, 116, 117].

Besides reconstructing state descriptions, neural networks were also used for correcting state-preparation and measurement errors in an experiment [117], calibrating a photonic sensor [167], detecting non-classicality [168], Wigner negativity [169], distinguishing different mode superpositions in a twisted light transmission experiment [170], and discriminating a coherent light source from thermal light at the single-photon level [171]. Another interesting application includes reconstructing the dynamics of a quantum system by observing its evolution [172].

These successful applications show the potential applications of neural networks in solving practical problems in quantum physics. The problems

can be broadly formulated as classification and generative modelling tasks in machine learning. In this chapter, we will string together all the concepts discussed so far and show how the general problem of classification and reconstruction of quantum states can be formulated as learning tasks for neural networks. We will then discuss how different types of noise can be tackled in this framework.

4.1 Classification of states with discriminative networks

In the standard quantum state discrimination task, we wish to identify an unknown quantum state ρ_i selected from a set of quantum states $\{\rho_i\}$. The probability of selecting the i^{th} state is p_i such that $\sum_i p_i = 1$. The goal is to devise a measurement scheme that allows us to discriminate an unknown state ρ in a single shot using a POVM ($\{\mathcal{O}_i\}$) where the probability of observing the i^{th} outcome corresponds to the probability that the unknown state is ρ_i. Therefore classification is a simple way to characterize the quantum state by identifying it by measurement.

However, perfect single-shot discrimination is not always possible or reliable, e.g., when the states are not orthogonal. A more realistic situation is when we have access to repetitions of the experiment and measurement outcomes, or possibly a continuous stream of outcomes from a weak measurement. Therefore, in this thesis we do not treat the standard task of finding best measurements for discriminating between different quantum states. We are rather concerned with the situation where we will have access to a dataset of measurement outcomes and wish to distinguish the underlying state according to its properties, e.g., non-classicality, entanglement, or the type of state.

We can connect the classification task to the standard machine-learning objective of determining the conditional probability of an instance of data $\mathbf{x}$ belonging to a class y given by $p(y|\mathbf{x})$ as discussed in Sec. 1.4. The input data is then a set of measurement outcomes and the description of the measurement $\mathbf{x} = [\{d_1, \mathcal{O}_1\}, \{d_2, \mathcal{O}_2\}, ..., \{d_k, \mathcal{O}_k\}]$ and the target, y are encoded labels such as entangled/non-entangled (0/1), Wigner-positive/negative (0/1).

A neural network can be used to parameterize the conditional probability $p_\theta(y|\mathbf{x})$ with weights θ. We represent this neural network as $f(\mathbf{x};\theta)$ that gives an output probability for an input data to belong to a class y. If there are multiple classes, the outputs can also be encoded as a vector

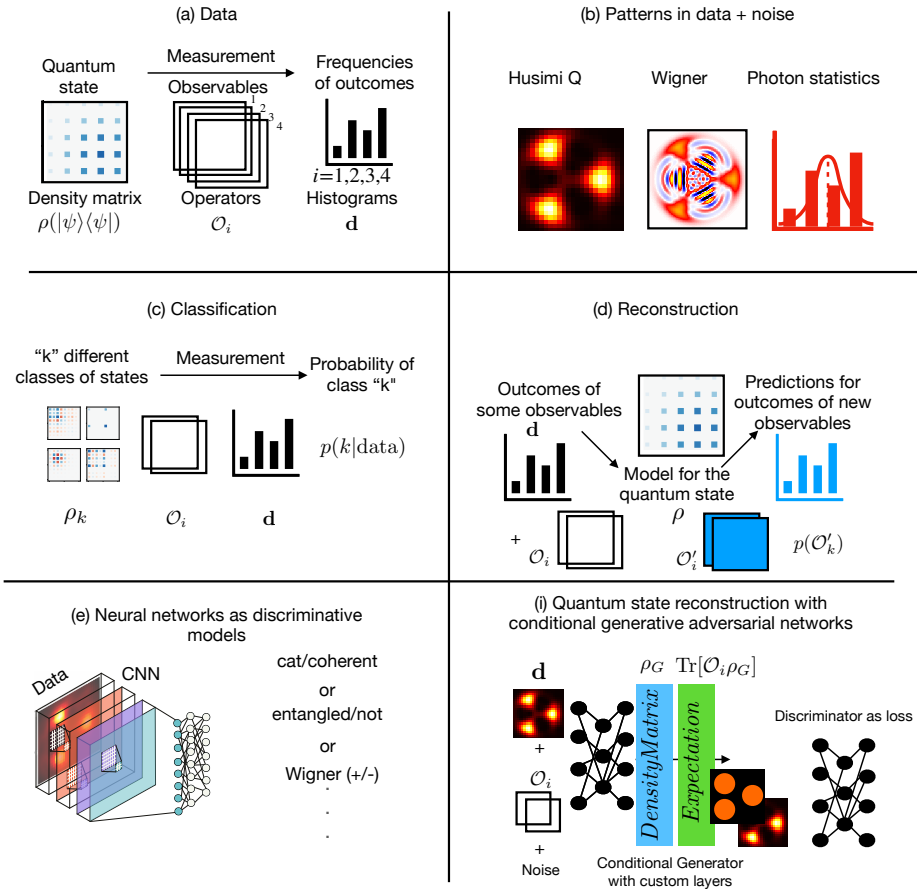

Figure 4.1: (a) The data in quantum state characterization problems is obtained by measuring a quantum state and collecting statistics of outcomes. (b) The data could have patterns depending on the underlying quantum state but noise could corrupt the data. (c) A classification task such as identifying which type of quantum state was produced in an experiment can be tackled using a neural network that is trained on simulated data. (d) In reconstruction tasks, we seek the underlying density matrix of the state that allows us to estimate the probability for arbitrary measurements on the state. (e) A convolutional neural network is well-suited to solve classification tasks where the data can be arranged as an image. The spatial patterns in the data are extracted by the convolutional neural network to make a decision. (e) Quantum state reconstruction using conditional generative adversarial neural networks uses the idea of two competing neural networks. One of them acts as a generator for probability of outcomes and the second neural network evaluates its performance as the discriminator. We further augment the generator neural network with custom layers that enforce rules of quantum mechanics.

$\mathbf{y} = [y_1, y_2, ..., y_n]$ where the y_n are probabilities for the data to belong to the n^{th} class.

If we now have a set of training data $\{\mathbf{d}_i, \mathbf{y}_i\}$, the weights θ can be learned by minimizing some loss function such as the cross-entropy loss between the predicted labels $\mathbf{y}'$ and the target,

$$\text{cross-entropy}(\mathbf{y}, \mathbf{y}') = -\sum_n y_n \log(y'_n). \tag{4.1}$$

We have assumed that since a neural network is a universal function approximator, it finds an approximation to the conditional probability for a class label given a data point. Once trained using a training dataset, we can predict the label for a new data point $\mathbf{d}'$ and use the neural network for classification without requiring the full quantum state description.

In the results presented in Paper II, we consider a convolutional neural network to classify seven different classes of optical quantum states - `fock`, `coherent`, `thermal`, `num`, `binomial`, `cat` and `gkp`. A convolutional neural network is well-suited to detect patterns in images using filters that learn specific features such as edges. The layer-wise architecture then progressively extracts more complex features, see Fig. 4.1(a). A fully connected set of layers in the end use the detected features to make a prediction that the data comes from a specific quantum state.

We considered as inputs the distribution of outcomes of measuring projections on coherent states $\mathcal{O}_i = |\alpha_i\rangle\langle\alpha_i|$ characterized by complex numbers α_i. These measurement outcomes can be arranged into a 2D matrix — the Husimi Q function, and contain specific patterns for each type of state. Note that the information about the measurement operators are only included implicitly in the data as the fixed grid for the Husimi Q function. It would also be possible to simply measure at random α_i and provide $\{\alpha_i, d_i\}$ as inputs to the neural network, where d_i is the estimated value for $\text{tr}\{\mathcal{O}_i\rho\}$. a

This task of classifying a state from its Husimi Q function could be seen as rather simple for a physicist, see Fig. 4.2. However, it is only possible with the naked eye once all the data is collected and arranged in a meaningful way. The advantage of using a neural network is the automation of this task and similar problems in the presence of noise in a real experiment. Moreover, if the measurement data is sparse and not arranged in a meaningful way, it would not be easy to just look at the data and make a prediction for the state. As we show in Paper II, a neural-network approach for reconstructing the full density matrix works well

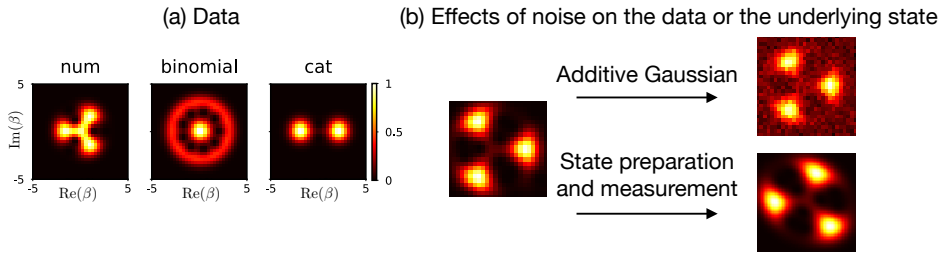

Figure 4.2: (a) Husimi-Q functions for different types of bosonic code states — `num`, `binomial` and `cat`. There are patterns in the data when they are arranged in a 2D matrix which makes it easy to recognize the type of state. Even if for a physicist, it is easy to recognize the states due to the symmetries, e.g., rotational, it is not straightforward to write a computer program to do the same. The difficulty comes from noise effects shown in (b). The noise might corrupt the state or the data but even then a physicist could identify the state. We train a neural network for a similar classification task which could also work with noisy or a subset of the full data allowing for automation in experiments.

even with a few measurements of randomly selected α_i. Our classification example simply shows how to formulate a general discriminative task using measurement data.

There are other possible advantages to using a neural network for classification. The information regarding the class of the state could be a powerful prior for other data-processing techniques. Since we are not estimating the full density matrix of the state, the method does not depend directly on the size of the state as long as the data contains a characteristic pattern that distinugishes the state. If we are only interested in certain properties of the state, we can use discriminative neural networks to reduce the cost of data analysis or requirement of a large number of data points. In Paper II, we also show how probing what the neural network considers important for its decision could help identify the regions in the data that gives the most information and consequently could guide the data collection process.

4.2 Reconstruction with custom generative neural networks

Quantum state reconstruction is a more difficult task than classification since we are interested in the full density matrix of the state. The interest

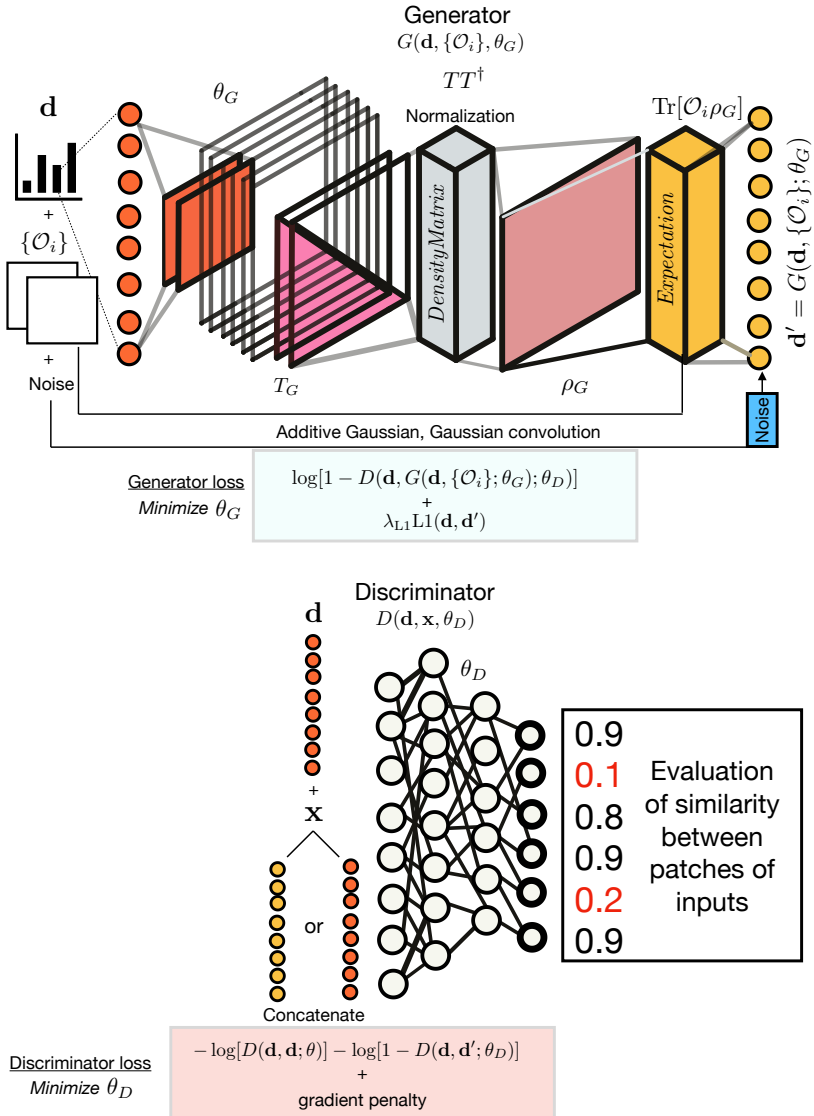

Figure 4.3: Reconstruction of a density matrix using a generative model that outputs a generated density matrix ρ_G using a custom *DensityMatrix* layer. The outputs of the model are the expectation values for a particular measurement operator $\mathcal{O}$ computed by the *Expectation* layer. A second neural network — the discriminator, evaluates the quality of reconstruction by ouputing a probability that the data comes from the target distribution. The generator and discriminator networks compete to fulfil opposing objectives until the generator learns to produce the correct output and the discriminator can no longer distinguish the generated data from the true data.

in the density matrix is really to generate statistics for any measurement outcome with the probability given by the Born rule,

$$p(\mathcal{O}) = \mathrm{tr}\{\mathcal{O}\rho\}. \tag{4.2}$$

The simplest approach to use a neural network for reconstruction is to consider an approximation of $p(\mathcal{O})$ using a neural network which we call a generator $g(\mathcal{O};\theta)$. The information about the state ρ is then captured by the weights of the generator and its outputs are related to the expectation value $\mathrm{tr}\{\mathcal{O}\rho\}$.

With this basic idea of using a neural network to capture the distribution of measurement outcomes on a state we can design different neural-network approches to learn the state [97]. Restricted Boltzmann machines were one of the earliest models for this task, but later other architectures such as variational auto-encoders, recurrent neural networks and Transformers were proposed for quantum state reconstruction [173]. In this work, we introduce a new framework using conditional generative adversarial networks (CGANs).

Our formulation is a generative network which is conditioned on the measurement outcomes $\mathbf{d} = \{d_1, d_2, ..., d_k\}$, which are the outcomes (or averages) for measurements $\{\mathcal{O}_k\}$,

$$g(\mathcal{O};\theta|\mathbf{d}). \tag{4.3}$$

In this formulation, we directly assume that the neural network outputs approximately predict the value of $\mathrm{tr}\{\mathcal{O}\rho\}$ for some state ρ and operator $\mathcal{O}$. However, we do not explicitly determine ρ in this model. We can then customize the neural network to first give an estimate of the density matrix — the generated density matrix, ρ_G, and then compute the probabilities for measurements as $\mathrm{tr}\{O\rho_G\}$.

We have two approaches to generative modelling for quantum state reconstruction now. We can either directly compute the probability for a measurement outcome as the output of a neural network. Or, we can output a density matrix and compute the probabilities by applying Born rule. The first implicit approach has the benefit that it will not scale exponentially with the dimension of the quantum system since we never explicitly compute the full density matrix. However, it is a black-box approach that does not give us access to the density matrix or allows us to put constraints on it, e.g., positive semidefiniteness or unit trace that make the state physical. Therefore, we may obtain results that are unphysical. An example is the

use of Transformer neural networks for state reconstruction where fidelities greater than 1 were reported in some reconstructions [173].

In our approach, where we explicitly ask for the density matrix, we both have a meaningful representation and can enforce constraint by construction. However, now we may have an exponentially growing number of parameters in the model since we are demanding the full density matrix. A workaround could be to assume a different, more efficient representation for the state such as a matrix product state and then define the equivalent of the Born rule to compute expectation values. In [97], the efficient parameterization of quantum states using tensor networks was combined with similar generative modelling concepts for reconstructing quantum states.

In Papers I and II, we show how to construct a generator using standard feed-forward deep neural networks. Let us now look at the inputs and outputs of the generator network more closely. At the input, we have a list of measurement operators $\{O_i\}$ which are the most general representation of a measurement operator. The dimension for these operators will grow exponentially with the system size. However, in most cases the measurement setting is defined by a smaller set of parameters, e.g., measuring in the coherent basis, we only need to specify the coherent displacement α. Therefore, we can replace the inputs $\mathcal{O}_i$ with the measurement parameters that define them. But then we would also need to define the custom rules that compute expectation values.

We can construct these custom layers following the rules of quantum mechanics and the properties that the density matrix (or any other state representation) should follow, see Fig. 4.3. These custom layers act as a regularization and ease the extra task of learning the Born rule implicitly via training. Further, with differential programming under our belt, we can write any customization adding more prior knowledge and train the generative model with backpropagation.

Other benefits of the neural-network approach using standard feedforward models are the flexibility to input data as either single shots, averages or even reconstruct multiple states with the same model by feeding data for different states to the same network. For each data vector $\mathbf{x}_i = \mathcal{O}_i, d_i$, there could be a neural-network estimate for the state ρ_i that best matches this data. Then, the generator g is acting as a map from the space of data points and measurements to the set of density matrices

$$g : \{\mathcal{O}, \mathbf{d}\}_i \to \{\rho_i\}. \tag{4.4}$$

We use this idea in Paper I for single-shot prediction of the density matrix of

a class of states after training on a set of data. An intuitive understanding for this formulation is using a neural network to generate realistic-looking human faces. There are some basic features common to all human faces, but depending on various inputs (woman, dark, smiling) the same network can output different realizations.

We have discussed how to formulate quantum state reconstruction as a generative modelling task. Now we will introduce the custom layers *DensityMatrix* and *Expectation* that enforce the rules of quantum mechanics in our model, see Fig. 4.3. This approach could be seen as adding prior information or regularization in the language of inverse problems. The *DensityMatrix* layer simply rearranges the outputs of the neural network into a complex matrix T, discards the upper triangular part of T and makes its diagonal real. This allows T to be interpreted as the Cholesky decomposition of some density matrix $\rho_G = TT^\dagger$, ensuring positive semidefiniteness of the matrix and Hermiticity. Then, we can implement a unit trace by normalizing (dividing) ρ_G with the trace value.

The *Expectation* layer simply applies the Born rule to the output of the *DensityMatrix* layer as $\mathrm{tr}\{\mathcal{O}\rho_G\}$. For other representations of the density matrix, e.g., matrix product states, the *Expectation* layer has to be redefined to correctly compute the expectation values from the given representation of the state and the measurement settings. With the general setup of the neural network model that can predict probabilities for measurements and also provides an intermediate density matrix representation, we are ready to train and learn the weights of the model.

4.3 Likelihood of the data and adversarial loss

Our formulation of the generative model captures the properties of the state in its weights θ. By construction, we have not obscured how this happens and have explicity added the *DensityMatrix* layer to give the full density matrix as an intermediate output. To learn the state, i.e., determine the θ values, we now minimize a loss function between the observed data and outputs from our neural network. If we consider single-shot outcomes $0, 1, 0, 0, ...$ for measurements, our neural network outputs a probability that is proportional to the frequencies for each outcome and we can define a cross-entropy loss $\mathcal{L}(\theta)_{\text{cross-entropy}}$ between the prediction and data.

Now, using any gradient-based optimization technique, we can minimize the loss function to determine the values θ which gives us an estimate

for the density matrix ρ_G. Since all the operations in the loss function are well-defined (including the custom operations in *DensityMatrix* and *Expectation* layers), auto-differentiation with a differential programming tool suffices to minimize $\mathcal{L}(\theta)_{\text{cross-entropy}}$. The motivation for the cross-entropy loss function is that minimizing the cross-entropy loss maximizes the log-likelihood function for observing the given data with our estimate of the density matrix,

$$\text{log-likelihood}(\theta) = \sum_i d_i \log(\text{tr}\{\mathcal{O}_i \rho_G\}). \tag{4.5}$$

However, the choice of the likelihood function is dependent on our assumption of the noise model as discussed in Sec. 2.4.1. Since the outcomes are interpreted as the probability of some event, the likelihood function was simply the product of the predicted probabilities repeated for how many times they occurred. But there is the possibility of another likelihood function with the assumption that the observed data has a Gaussian noise associated with each measurement outcome. Maximizing the log-likelihood with the assumption of Gaussian noise results in minimizing the mean squared error between predictions and data, see Sec. 2.4.1.

Therefore, there are different possibilities for the likelihood function and choosing the loss function. Consider the case that a particular measurement is much more informative than others and we would like it to have more weight in the loss function. Or, we do not fully know the different possible sources of noise. In such situations, it might be useful to also make an abstract formulation for the loss function as another neural network and learn it along with the parameters of the generator.

Here comes the idea of adversarial learning with generative adversarial networks (GANs). In the GAN framework, two neural networks — a generator and a discriminator, compete to optimize opposing objectives. The generator tries to produce outputs that resemble the data and the discriminator outputs the probability that an input data belongs to a particular distribution, see Fig. 4.3. Therefore the generator tries to produce outputs that have a similarity to the training data and the discriminator evaluates the quality of generation and learns a loss function implicitly.

In Paper II, we present a detailed comparison of training a generator with standard loss functions against training using the GAN framework. The discriminator network $D(\mathbf{d}')$ outputs a value that signifies the probability of the sample $\mathbf{d}'$ to belong to the true data distribution. This output could be a single value $\in [0, 1]$ or a set of values $[0, 0, 1, ..., 1]$ which further tells

us the confidence for different parts of the input to match the training data instances. The idea follows from the PatchGAN architecture of [56]. An intuitive way to understand it is to think about a generated image from a neural network. The discriminator output could also be an image that gives a value ($\in [0, 1]$) measuring the correctness of a patch of pixels such as eyes in the generated image.

In Paper II, we compared the results of using such as discriminator network along with a standard L_1 loss for Gaussian noise to motivate the choice of such a loss function over standard losses. One of the benefits of such as loss function is that is is not dependent on our assumption of the likelihood and can flexibly learn higher abstract concepts. We now turn to dealing with various types of noise in data and how we can include them in our models. The intuition is that we could potentially train neural-network methods to learn to be agnostic to noise and learn an underlying state.

4.4 Tackling noise

Noise makes many types of inverse problems ill-posed and leads to a failure of different quantum-state reconstruction techniques. Using neural networks, we have multiple options to handle noise. For classification tasks, the noise could be added in data while training such that the predictions of the neural network are agnostic to the noise. This method is a standard practice in most pipelines using deep neural networks and allows robust predictions. Consider the experiment where a neural network was used to classify mode patterns of spatial modes of light that travelled through 3 km of turbulence through a city [170]. Since the neural network was trained on data that contained atmosphere-induced disturbances, it robustly learned the patterns necessary for classification.

It is also possible to add noise layers in the neural-network model itself. Then, the network learns the correct underlying state and becomes agnostic to the noise since it is now a part of the model. In the generative model, noise layers could be added at the output of the *Expectation* layer, e.g., additive Gaussian noise.

Finally, the choice of the loss function is crucially dependent on the noise model as we saw in Sec. 2.4.1. However, statistical arguments can be used to show that it is possible to reconstruct a clean output from noisy observations directly by only considering the noisy data [174]. An interesting property of the mean squared loss was the crucial insight that

the expected value of the loss function remains unchanged if the targets are replaced with random numbers whose expectations match the target. Therefore the network weights are unchanged if the training targets are corrupted by zero-mean noise and hence the network learns a model that is noise-agnostic. A further benefit of using the discriminator as a loss function is that the network can lean to focus on overall properties of the state and ignore the noise.

In our works, we have considered different types of noise, see Chapter 2, for both the classification and reconstruction tasks. We have shown that it is possible to include known additive Gaussian and convolution noise in the generator and still learn the underlying quantum state. However, note that noise makes the inverse problem ill-defined. So, it is possible that we require more information for truly verifying if the reconstruction is correct since there could be many different possible states that can be corrupted with noise to give the same data. Regularization and prior information become essential in such cases as there is no silver bullet to tackle all types of noise.

In the next chapter, we summarize the results of classification and reconstruction presented in the papers I and II following the concepts discussed so far in this thesis.

Chapter 5

Summary of papers

Here we give an overview of Papers I and II that this thesis is based on and connect the theoretical ideas discussed in the previous chapters to the results of the paper.

In Paper I, we introduce the idea of using generative adversarial networks to learn the density matrix of a quantum state. We outlined the basic problem of quantum state tomography in Chapter 2 and the difficulties associated with learning quantum states. The adversarial learning framework, as discussed in Sec. 3.3.3, uses two neural networks, a generator and a discriminator, that compete with each other to learn a data distribution. We presented a new approach that uses conditional generative adversarial networks (CGANs) for quantum state tomography (QST) — the QST-CGAN. In our approach, a generator computes the statistics for measurement outcomes on a quantum state. The output of the generator is conditioned on a given experimental dataset of measurement outcomes. The generator can then compute the probability for any measurement and therefore characterizes the quantum state that generates the data.

The predicted output from the generator is a density matrix that is used to compute probabilities for measurement outcomes. In order to have physically justifiable probabilities we require some constraints on the density matrix, that was discussed in Sec. 2.1. In our formulation, using the CGAN framework, we therefore implement two customizations in the generator. First, an intermediate *DensityMatrix* layer moulds the output from any standard neural network into a valid density matrix. This custom layer ensures that the estimated density matrix is physically valid by using the idea of Cholesky decomposition to enforce a positive semidefinite

matrix. We also normalize the trace of the intermediate density matrix to be unity. Then, an *Expectation* layer computes the actual probability for a measurement using the Born rule.

In order to train the generator, we use the ideas of automatic differentiation and differential programming discussed in Sec. 3.1.2 and Sec. 3.2. The combination of a simple loss function and a second neural network, the discriminator, allows a flexible approach that does not assume an explicit likelihood for the data. As we discuss in Sec. 2.4, the choice of the likelihood function is dependent on the assumed noise. The discriminator compares and evaluates the predicted outcomes to the actual data in patches. Therefore we expect the QST-CGAN method to learn abstract patterns and features in the data with the discriminator guiding the generator to learn the density matrix.

We then show in Paper I that the QST-CGAN method can learn the density matrix of an optical quantum state such as a Schrödinger cat state with a high fidelity (~ 0.99) from samples of a simulated Husimi Q function as data. The advantage of using QST-CGAN is demonstrated by examples that show orders of magnitude faster reconstruction than the iterative maximimum likelihood estimation (iMLE) technique discussed in Sec. 2.4.1. We demonstrate reconstruction using $\sim 10\times$ fewer data points. As we discuss in Sec. 2.2.1, quantum state reconstruction requires an exponential number of measurements in the size of the quantum system. Therefore one advantage of our method is potentially lowering the number of measurements necessary for reconstruction.

We also demonstrate the effectiveness and the flexibility of our method by reconstructing an optical quantum state from noisy experimental data of an experimentally measured Wigner function. Finally, we show how such a method can be flexibly adapted to reconstruct quantum states with high fidelities in a single evaluation of the generator after training on a class of quantum states.

In Paper II, we connect the tasks of quantum state classification and reconstruction to generative and discriminative modelling problems in machine learning which we explain in detail in Sec. 3.3. In this much longer and more detailed work, we first show how neural networks can be used for classification of quantum states and then discuss more details about the QST-CGAN method. We demonstrate that a convolutional neural network can classify different optical quantum states with almost perfect accuracy even in the presence of significant additive Gaussian noise

or photon loss. The classification accuracies remained almost perfect for additive Gaussian noise up to 20% of the maximum value of the input data. In case of photon loss, we demonstrate that the neural network could recognize a Schrödinger cat state when it lost almost 70% of the initial photons. These demonstrations show the ability of the neural-network approach to make decisions by learning general patterns and features in the data and demonstrate robustness against noise when trained properly. Since noise makes such inverse problems ill-posed as we discuss in Chapter 1, neural-network-based techniques could be effective in dealing with such tasks.

To conclude the discussion on classification by neural networks, we further explore how the neural network predictions depend on the input data. We apply the Grad-CAM method to highlight which parts of a noisy input data the network focuses on to make a prediction, thereby indicating that the network can identify parts of the image that are relevant and be agnostic to Gaussian additive noise.

In the second part of Paper II, we focus on quantum state reconstruction using our QST-CGAN approach under different types of noise, for mixed states and with a very small subset of the full data from phase space measurements. In order to motivate the benefit of using a learnable loss function, we consider the fidelity of reconstruction using simple loss functions such as L_1, L_2, cross-entropy and KL divergence on noisy data. In Sec. 3.3.2, we discussed briefly how choosing a good loss function might be pivotal with a discussion on how blurry images generated by autoencoders might be a consequence of a simple loss metric. We also presented the relation between a least-squares minimization and Gaussian noise in Sec. 2.4.1. Therefore our empirical analysis of how various loss functions influence the reconstruction of quantum states provides some motivation for using a learnable loss function.

We continue our discussion about noise in Paper II, by adding Gaussian noise to data from a zero-mean normal distribution with standard deviation $\sigma = 0.05$ after normalizing the data between 0 and 1. The QST-CGAN approach, that uses a combination of discriminator loss and L_1, consistently leads to mean fidelities for reconstruction > 0.9 for 30 different sets of noise for each loss function. On the other hand, the other loss functions fail to reach a mean fidelity of 0.9 with only L_2 loss giving a mean fidelity of 0.87. Note that the L_2 loss is the natural choice for the loss function when considering additive Gaussian noise. In comparison, iMLE is highly

unstable and does not converge in the same number of iterations as the QST-CGAN.

Then, we show how to add Gaussian convolution noise to the QST-CGAN so that we can reconstruct quantum states from data that has been corrupted by multiplicative noise. Such a situation occurs in linear detection schemes where amplification of the signal corrupts it by a convolution operation with the background signal from the detection channel. We can recover a single-photon Fock state with perfect fidelity from data that has been convoluted with Gaussian noise corresponding to a thermal state in the channel with mean photon number 5. However, we also show that such reconstruction is not always possible as sometimes even if the reconstruction has a perfect match with the data, the underlying quantum state might be incorrectly identified.

Finally, we show how the QST-CGAN can also reconstruct mixed states with a random set of data points. We consider examples with mixed states of rank $r = 2, 3, 4$, where the QST-CGAN approach only requires a few hundred data points while the iMLE method fails to reconstruct the state at all. In such situations, the QST-CGAN approach learns the best description given a smaller subset of the data. Therefore our neural-network based approach could be applicable to a wide variety of cases both with simplifications in the problem and considering weaker learning models discussed in Sec. 2.4.2. In an example with a rank-4 state, we show that 256 random measurements on the phase space were enough for QST-CGAN to reconstruct the underlying quantum state. Our demonstrations therefore show the benefits of the QST-CGAN approach over iMLE as well as using standard loss functions.

Chapter 6

Conclusion and outlook

In this thesis, we have linked various concepts related to the application of machine learning in quantum physics. Specifically, we focused on the characterization of quantum states using a successful machine-learning method — deep neural networks. We have introduced the basic ideas behind inverse problems, machine learning and the problem of estimating quantum states from data in Chapter 1 and Chapter 2. In Chapter 2, we presented various existing techniques for quantum state reconstruction and their complexities. Then, in Chapter 3 we discussed neural networks and the idea of differential programming that allows writing computer programs that can be trained similar to neural networks. Finally, in Chapter 4 we presented the ideas of the classification and reconstruction of quantum states using discriminative and generative models. By showing that standard feedforward deep neural networks can be flexibly adopted for state classification and reconstruction tasks, we have presented a unified approach to apply deep learning for quantum state characterization. The main contribution of the thesis is therefore to reduce the gap between tasks in quantum physics and machine learning with deep neural networks.

Deep neural networks are ubiquitous in machine learning, and slowly becoming an interesting tool for physicists. The neural-network approach could be very effective in analyzing noisy experimental data, or setting up online methods that automatically improve on various tasks. We have shown how classification of quantum data can be easily formulated as a problem to be solved using deep neural networks. In the appended papers, we presented the results of such a formulation and showed its success. We have also hinted at possibilities to set up adaptive methods for reducing

experimental effort by exploiting patterns that a neural network learns during training.

In terms of fully characterizing a quantum state by learning the density matrix, we have introduced a new method using generative adversarial networks — QST-CGAN. We outlined a general connection between inverse problems, tasks in quantum physics and discussed how deep neural networks can be customized to incorporate quantum-mechanical rules. Our proposed QST-CGAN approach was able to show multiple advantages in state reconstruction tasks compared to an iterative maximum likelihood estimation. Moreover, we showed how to handle noisy data and customize neural networks in general for state reconstruction. To demonstrate the further flexibility of our proposed neural-network approach, we reconstructed a quantum state from noisy experimental data and showed how to perform single shot reconstruction with pre-training. We demonstrated the ease with which noise can be handled using the neural-network approach and showed the effectiveness of using a discriminative training technique over standard loss functions. Further, we also showed evidence that the neural-network approach could work with much fewer data points and iterative steps than the iterative maximum likelihood estimation.

Our results and ideas for state classification can be extended to other problems such as verifying genuine multipartite entanglement or improving readout of qubits. The prediction of properties of complicated quantum systems with reduced data, and in the presence of noise, is therefore a straightforward extension of our work on classification. For reconstruction, we would like to explore how well the QST-CGAN method works for other types of quantum systems such as qubits. Since we demand the full density matrix of the state, the method will not scale straightforwardly with the size of the quantum system. However, we might be able to formulate an approach that gives us a reduced representation for the state. This is an interesting future direction to explore..

Another direction for future work would be to explore the representation capacity of standard neural networks in the adversarial learning framework to learn a complicated quantum state. We have used a fairly complex neural network for the state-reconstruction task inspired by standard deep generative models. As future work, we would like to explore if smaller networks with much fewer parameters suffice for reconstruction. Lastly, the ability to flexibly incorporate quantum mechanical rules and priors in neural network models opens up several possibilities to tackle other tasks such

as quantum process tomography using deep neural networks. This could allow for efficient methods to learn quantum states and processes using deep neural networks in an online manner using the adversarial training approach we discussed. The final goal would be to explore the reduction in data collection and post-processing costs for experiments and characterizing noisy quantum devices in an efficient manner.

References

[1] A. Sudbery, *Quantum Mechanics and the Particles of Nature: An Outline for Mathematicians* (Cambridge University Press, 1986).

[2] R. F. Bader, "A Quantum Theory of Molecular Structure and Its Applications", Chemical Reviews **91**, 893 (1991).

[3] C. C. Gerry and P. L. Night, *Introductory Quantum Optics* (Cambridge University Press, 2005).

[4] D. Harlow, "Jerusalem lectures on black holes and quantum information", Reviews of Modern Physics **88**, 15002 (2016).

[5] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines", Journal of Statistical Physics **22**, 563 (1980).

[6] Y. Manin, *Computable and Uncomputable (in Russian)*, Sovetskoye Radio, Moscow, 1980.

[7] R. P. Feynman, "Simulating physics with computers", International Journal of Theoretical Physics **21**, 467 (1982).

[8] Y. I. Manin, "Classical computing, quantum computing and Shor's factoring algorithm", arXiv:quant-ph/9903008 (2000).

[9] L. J. Samuel, "Quantum Computation: A Grand Mathematical Challenge for the Twenty-first Century and the Millennium", in Proceedings of Symposia in Applied Mathematics, Vol. 58 (2002).

[10] D. R. Simon, "On the power of quantum computation", SIAM journal on computing **26**, 1474 (1997).

[11] U. Vazirani, "On the power of quantum computation", Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **356**, 1759 (1998).

[12] C. H. Bennett and D. P. DiVincenzo, "Quantum information and computation", Nature **404**, 247 (2000).

[13] F. Arute et al., "Quantum supremacy using a programmable superconducting processor", Nature **574**, 505 (2019).

[14] I. H. Deutsch, "Harnessing the Power of the Second Quantum Revolution", PRX Quantum **1**, 20101 (2020).

[15] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu, et al., "Quantum computational advantage using photons", Science **370**, 1460 (2020).

[16] R. Landauer, "The physical nature of information", Physics Letters, Section A: General, Atomic and Solid State Physics **217**, 188 (1996).

[17] R. Jozsa, "Entanglement and Quantum Computation", arXiv:quant-ph/9707034 (1997).

[18] C. G. Almudever et al., "The engineering challenges in quantum computing", in Proceedings of the 2017 Design, Automation and Test in Europe, DATE (2017), pp. 836–845.

[19] J. Haah, A. W. Harrow, Z. Ji, X. Wu, and N. Yu, "Sample-optimal tomography of quantum states", IEEE Transactions on Information Theory **63**, 5628 (2017).

[20] A. A. Clerk, M. H. Devoret, S. M. Girvin, F. Marquardt, and R. J. Schoelkopf, "Introduction to quantum noise, measurement, and amplification", Reviews of Modern Physics **82**, 1155 (2010).

[21] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning", Nature **549**, 195 (2017).

[22] S. Das Sarma, D. L. Deng, and L. M. Duan, "Machine learning meets quantum physics", Physics Today **72**, 48 (2019).

[23] H. Moon et al., "Machine learning enables completely automatic tuning of a quantum device faster than human experts", Nature Communications **11**, 4161 (2020).

[24] 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", Proceedings of the National Academy of Sciences **115**, 1221 (2018).

[25] M. Krenn, M. Erhard, and A. Zeilinger, "Computer-inspired quantum experiments", Nature Reviews Physics **2**, 649 (2020).

[26] A. Youssry, G. A. Paz-Silva, and C. Ferrie, "Characterization and control of open quantum systems beyond quantum noise spectroscopy", npj Quantum Information **6**, 95 (2020).

[27] N. Wittler et al., "An integrated tool-set for control, calibration and characterization of quantum devices applied to superconducting qubits", arXiv:2009.09866 (2020).

[28] B. Lienhard et al., "Deep Neural Network Discrimination of Multiplexed Superconducting Qubit States", arXiv:2102.12481 (2021).

[29] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning* (MIT press, 2016).

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

[31] Schütt, Kristof T. et al., *Machine learning meets quantum physics. Lecture notes in physics* (Springer International Publishing, 2020).

[32] M. J. Klein, "Max Planck and the beginnings of the quantum theory", Archive for History of Exact Sciences **1**, 459 (1975).

[33] M. Planck, "On the Law of the Energy Distribution in the Normal Spectrum", Annalen der Physik **4**, 553 (1901).

[34] H. Walther, B. T. H. Varcoe, B.-G. Englert, and T. Becker, "Cavity quantum electrodynamics", Reports on Progress in Physics **69**, 1325 (2006).

[35] 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 Journal of Physics **18**, 083036 (2016).

[36] M. Namiki, "Decoherence and wavefunction collapse in quantum measurements", Foundations of Physics **29**, 457 (1999).

[37] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in Proceedings of the 2001 ieee computer society conference on computer vision and pattern recognition. cvpr 2001, Vol. 1 (IEEE, 2001), pp. 511–518.

[38] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: real and stealthy attacks on state-of-the-art face recognition", in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16 (2016), 1528–1540.

[39] A. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep Reinforcement Learning framework for Autonomous Driving", Electronic Imaging **2017**, 70 (2017).

[40] T. B. Brown et al., "Language Models are Few-Shot Learners", arXiv:2005.14165 (2020).

[41] D. Silver et al., "Mastering the game of go without human knowledge", Nature **550**, 354 (2017).

[42] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, "Mastering Atari, Go, chess and shogi by planning with a learned model", Nature **588**, 604 (2020).

[43] L. Gatys, A. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style", Journal of Vision **16**, 326 (2016).

[44] G. Branwen, *GPT-3 Creative Fiction*, Accessed on 10.03.2021, (2020) `https://www.gwern.net/GPT-3`.

[45] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, and C. Blundell, "Agent57: Outperforming the atari human benchmark", arXiv (2020).

[46] J. Jumper et al., "High Accuracy Protein Structure Prediction Using Deep Learning", in Fourteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstract Book) (2020), pp. 22–24.

[47] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", Commun. ACM **60**, 84–90 (2017).

[48] The TensorFlow Team, *Flowers*, Accessed on 10.03.2021, (2019) `http://download.tensorflow.org/example_images/flower_photos.tgz`.

[49] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection", in Proceedings of the 26th International Conference on Neural Information Processing Systems, NIPS'13 (2013), 2553–2561.

[50] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement", arXiv:1804.02767 (2018).

[51] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention", 32nd International Conference on Machine Learning, ICML 2015 **3**, 2048 (2015).

[52] Wikimedia Commons, *Surfing in Hawaii*, Accessed on 10.03.2021, `https://commons.wikimedia.org/wiki/Surfing#/media/File:Surfing_in_Hawaii.jpg`.

[53] C. He and H. Hu, *Image Captioning with Text-Based Visual Attention*, Accessed on 10.03.2021, (2019) `https://www.tensorflow.org/tutorials/text/image_captioning`.

[54] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative Adversarial Text to Image Synthesis", in 33rd International Conference on Machine Learning, Vol. 3 (PMLR, 2016), pp. 1681–1690.

[55] B. Li, X. Qi, T. Lukasiewicz, and P. Torr, "Controllable text-to-image generation", in Advances in Neural Information Processing Systems, Vol. 32 (2019).

[56] P. Isola, J. 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) (2017), pp. 5967–5976.

[57] F. Pan and P. Zhang, "Simulating the Sycamore quantum supremacy circuits", arXiv:2103.03074 (2021).

[58] J. Kelly, P. O'Malley, M. Neeley, H. Neven, and J. M. Martinis, "Physical qubit calibration on a directed acyclic graph", arXiv:1803.03226 (2018).

[59] S. J. Devitt, W. J. Munro, and K. Nemoto, "Quantum error correction for beginners", Reports on Progress in Physics **76**, 076001 (2013).

[60] D. A. Lidar and T. A. Brun, *Quantum error correction* (Cambridge University Press, 2013).

[61] J. E. Bourassa, R. N. Alexander, M. Vasmer, A. Patil, I. Tzitrin, T. Matsuura, D. Su, B. Q. Baragiola, S. Guha, G. Dauphinais, K. K. Sabapathy, N. C. Menicucci, and I. Dhand, "Blueprint for a Scalable Photonic Fault-Tolerant Quantum Computer", Quantum **5**, 392 (2021).

[62] D. Kielpinski, C. Monroe, and D. J. Wineland, "Architecture for a large-scale ion-trap quantum computer", Nature **417**, 709 (2002).

[63] 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 **584**, 205 (2020).

[64] P. Chapman, *Introducing the Worlds Most Powerful Quantum Computer*, Accessed on 10.03.2021, (2020) `https://ionq.com/posts/october-01-2020-introducing-most-powerful-quantum-computer`.

[65] Y. Nam et al., "Ground-state energy estimation of the water molecule on a trapped-ion quantum computer", npj Quantum Information **6**, 33 (2020).

[66] H.-J. Briegel, T. Calarco, D. Jaksch, J. I. Cirac, and P. Zoller, "Quantum computing with neutral atoms", Journal of Modern Optics **47**, 415 (2000).

[67] L. Henriet, L. Beguin, A. Signoles, T. Lahaye, A. Browaeys, G.-O. Reymond, and C. Jurczak, "Quantum computing with neutral atoms", Quantum **4**, 327 (2020).

[68] M. Friesen, P. Rugheimer, D. E. Savage, M. G. Lagally, D. W. van der Weide, R. Joynt, and M. A. Eriksson, "Practical design and simulation of silicon-based quantum-dot qubits", Physical Review B **67**, 121301 (2003).

[69] R. Zhao et al., "Single-spin qubits in isotopically enriched silicon at low magnetic field", Nature Communications **10**, 5500 (2019).

[70] R. Li, L. Petit, D. P. Franke, J. P. Dehollain, J. Helsen, M. Steudtner, N. K. Thomas, Z. R. Yoscovits, K. J. Singh, S. Wehner, L. M. K. Vandersypen, J. S. Clarke, and M. Veldhorst, "A crossbar network for silicon quantum dot qubits", Science Advances **4**, eaar3960 (2018).

[71] S. Pezzagna and J. Meijer, "Quantum computer based on color centers in diamond", Applied Physics Reviews **8**, 011308 (2021).

[72] N. Gershenfeld and I. L. Chuang, "Quantum Computing with Molecules", Scientific American **278**, 66 (1998).

[73] V. V. Albert, J. P. Covey, and J. Preskill, "Robust Encoding of a Qubit in a Molecule", Physical Review X **10**, 031050 (2020).

[74] G. Nakamura, *Inverse Modeling An introduction to the theory and methods of inverse problems and data assimilation* (IOP Publishing, 2015).

[75] G. Bal, "Introduction to inverse problems", Lecture Notes - Department of Applied Physics and Applied Mathematics, Columbia University, New York (2012).

[76] J Hadamard, "Sur les problems aux derivees patielles et leur signification physique", Princeton University Bulletin **13**, 49 (1902).

[77] T. A. N., "On the solution of improperly posed problems and the method of regularization", in Sov. Math. Vol. 5, 3 (Russian Academy of Sciences, 1963), p. 1035.

[78] T. Helin and M. Burger, "Maximum a posteriori probability estimates in infinite-dimensional Bayesian inverse problems", Inverse Problems **31**, 085009 (2015).

[79] A. F. M. Smith and A. E. Gelfand, "Bayesian Statistics without Tears: A Sampling-Resampling Perspective", The American Statistician **46**, 84 (1992).

[80] R. Chartrand, "Numerical differentiation of noisy, nonsmooth, multidimensional data", in 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP) (2017), pp. 244–248.

[81] D. A. Lorenz and A. Rösch, "Error estimates for joint Tikhonov and Lavrentiev regularization of constrained control problems", Applicable Analysis **89**, 1679 (2010).

[82] E. De Vito, L. Rosasco, A. Caponnetto, U. De Giovannini, F. Odone, and P. Bartlett, "Learning from examples as an inverse problem.", Journal of Machine Learning Research **6** (2005).

[83] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning", Annals of Statistics **36**, 1171 (2008).

[84] J. Adler and O. Öktem, "Solving ill-posed inverse problems using iterative deep neural networks", Inverse Problems **33**, 124007 (2017).

[85] A. Lucas, M. Iliadis, R. Molina, and A. K. Katsaggelos, "Using Deep Neural Networks for Inverse Problems in Imaging: Beyond Analytical Methods", IEEE Signal Processing Magazine **35**, 20 (2018).

[86] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations", Journal of Computational Physics **378**, 686 (2019).

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

[88] G. Cybenko, "Approximation by superpositions of a sigmoidal function", Mathematics of control, signals and systems **2**, 303 (1989).

[89] 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", Journal of the Optical Society of America A **37**, 422 (2020).

[90] T. Beucler, M. Pritchard, S. Rasp, J. Ott, P. Baldi, and P. Gentine, "Enforcing Analytic Constraints in Neural Networks Emulating Physical Systems", Physical Review Letters **126**, 098302 (2021).

[91] G. W. Lindsay, "Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future", arXiv:2001.07092 (2020).

[92] J. J. Sakurai and J. Napolitano, *Modern quantum mechanics*, 2nd ed. (Cambridge University Press, 2017).

[93] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2010).

[94] K. Banaszek, G. M. D'Ariano, M. G. A. Paris, and M. F. Sacchi, "Maximum-likelihood estimation of the density matrix", Physical Review A **61**, 010304 (1999).

[95] 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", Physical Review A **97**, 032346 (2018).

[96] 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", Nature Communications **1**, 149 (2010).

[97] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, "Reconstructing quantum states with generative models", Nature Machine Intelligence **1**, 155 (2019).

[98] S. T. Flammia, D. Gross, Y.-K. Liu, and J. Eisert, "Quantum tomography via compressed sensing: error bounds, sample complexity and efficient estimators", New Journal of Physics **14**, 095022 (2012).

[99] R. O'Donnell and J. Wright, "Quantum spectrum testing", in Proceedings of the forty-seventh annual acm symposium on theory of computing, STOC '15 (2015), 529–538.

[100] R. Kueng, H. Rauhut, and U. Terstiege, "Low rank matrix recovery from rank one measurements", Applied and Computational Harmonic Analysis **42**, 88 (2017).

[101] R. O'Donnell and J. Wright, "Efficient quantum tomography", in Proceedings of the forty-eighth annual ACM symposium on Theory of Computing (2016), pp. 899–912.

[102] 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", Physical Review A **94**, 052327 (2016).

[103] D. Sych, J. Řeháček, Z. Hradil, G. Leuchs, and L. L. Sánchez-Soto, "Informational completeness of continuous-variable measurements", Physical Review A **86**, 052123 (2012).

[104] J. M. Renes, R. Blume-Kohout, A. J. Scott, and C. M. Caves, "Symmetric informationally complete quantum measurements", Journal of Mathematical Physics **45**, 2171 (2004).

[105] C. Fuchs, M. Hoang, and B. Stacey, "The SIC Question: History and State of Play", Axioms **6**, 21 (2017).

[106] H. Zhu, "Quantum state estimation with informationally overcomplete measurements", Physical Review A **90**, 012115 (2014).

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

[108] B. Qi, Z. Hou, L. Li, D. Dong, G. Xiang, and G. Guo, "Quantum state tomography via linear regression estimation", Scientific reports **3**, 1 (2013).

[109] A. I. Lvovsky and M. G. Raymer, "Continuous-variable optical quantum-state tomography", Reviews of Modern Physics **81**, 299 (2009).

[110] A. I. Lvovsky, "Iterative maximum-likelihood reconstruction in quantum homodyne tomography", Journal of Optics B: Quantum and Semiclassical Optics **6**, S556 (2004).

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

[112] R. Blume-Kohout, "Optimal, reliable estimation of quantum states", New Journal of Physics **12**, 043034 (2010).

[113] C. Granade, J. Combes, and D. G. Cory, "Practical Bayesian tomography", New Journal of Physics **18**, 033024 (2016).

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

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

[116] S. Lohani, B. T. Kirby, M. Brodsky, O. Danaci, and R. T. Glasser, "Machine learning assisted quantum state estimation", Machine Learning: Science and Technology **1**, 035007 (2020).

[117] 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 Information **6**, 20 (2020).

[118] 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", Physical Review A **102**, 042604 (2020).

[119] J. Řeháček, Z. Hradil, E. Knill, and A. I. Lvovsky, "Diluted maximum-likelihood algorithm for quantum tomography", Physical Review A **75**, 042108 (2007).

[120] J. Shang, Z. Zhang, and H. K. Ng, "Superfast maximum-likelihood reconstruction for quantum tomography", Physical Review A **95**, 062336 (2017).

[121] J. A. Smolin, J. M. Gambetta, and G. Smith, "Efficient Method for Computing the Maximum-Likelihood Quantum State from Measurements with Additive Gaussian Noise", Physical Review Letters **108**, 070502 (2012).

[122] C. Ferrie and R. Blume-Kohout, "Maximum likelihood quantum state tomography is inadmissible", arXiv:1808.01072 (2018).

[123] G. Tóth, W. Wieczorek, D. Gross, R. Krischek, C. Schwemmer, and H. Weinfurter, "Permutationally Invariant Quantum Tomography", Physical Review Letters **105**, 250403 (2010).

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

[125] S. Arunachalam, Y. Quek, and J. Smolin, "Private learning implies quantum stability", arXiv:2102.07171 (2021).

[126] M. P. da Silva, O. Landon-Cardinal, and D. Poulin, "Practical Characterization of Quantum Devices without Tomography", Physical Review Letters **107**, 210404 (2011).

[127] S. T. Flammia and Y. K. Liu, "Direct fidelity estimation from few Pauli measurements", Physical Review Letters **106**, 230501 (2011).

[128] S. Aaronson, "Shadow Tomography of Quantum States", SIAM Journal on Computing **49**, STOC18 (2020).

[129] H.-Y. Huang, R. Kueng, and J. Preskill, "Predicting many properties of a quantum system from very few measurements", Nature Physics **16**, 1050 (2020).

[130] A. Rocchetto, S. Aaronson, S. Severini, G. Carvacho, D. Poderini, I. Agresti, M. Bentivegna, and F. Sciarrino, "Experimental learning of quantum states", Science Advances **5**, eaau1946 (2019).

[131] S. Aaronson, X. Chen, E. Hazan, S. Kale, and A. Nayak, "Online learning of quantum states", Journal of Statistical Mechanics: Theory and Experiment.

[132] A. Rakhlin, K. Sridharan, and A. Tewari, "Online learning: Random averages, combinatorial parameters, and learnability", Advances in Neural Information Processing Systems **23** (2010).

[133] Alan M Turing, *Intelligent machinery*, Accessed on 10.03.2021, (1948) `http://www.alanturing.net/intelligent_machinery/`.

[134] C. S. Webster, "Alan Turing's unorganized machines and artificial neural networks: his remarkable early work and future possibilities", Evolutionary Intelligence **5**, 35 (2012).

[135] M. Abadi and G. D. Plotkin, "A simple differentiable programming language", Proceedings of the ACM in Programing Languages **4** (2019).

[136] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey", Journal of Machine Learning Research **18**, 1 (2018).

[137] *"Gradient descent can write code better than you"*, `https://twitter.com/karpathy/status/893576281375219712`.

[138] N. Cotter, "The Stone-Weierstrass theorem and its application to neural networks", IEEE Transactions on Neural Networks **1**, 290 (1990).

[139] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function", Neural Networks **6**, 861 (1993).

[140] H. W. Lin, M. Tegmark, and D. Rolnick, "Why Does Deep and Cheap Learning Work So Well?", Journal of Statistical Physics **168**, 1223 (2017).

[141] S. Moon, "ReLU Network with Bounded Width Is a Universal Approximator in View of an Approximate Identity", Applied Sciences **11**, 427 (2021).

[142] O. Delalleau and Y. Bengio, "Shallow vs. deep sum-product networks", in Advances in Neural Information Processing Systems, Vol. 24, edited by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger (2011).

[143] H. N. Mhaskar and T. Poggio, "Deep vs. shallow networks: An approximation theory perspective", Analysis and Applications **14**, 829 (2016).

[144] R. Eldan and O. Shamir, "The power of depth for feedforward neural networks", in Conference On Learning Theory (2016).

[145] N. Cohen, O. Sharir, and A. Shashua, "On the expressive power of deep learning: a tensor analysis", in 29th Annual Conference on Learning Theory, Vol. 49, Proceedings of Machine Learning Research (2016), pp. 698–728.

[146] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", Nature **323**, 533 (1986).

[147] M. A. Nielsen, *Neural Networks and Deep Learning* (Determination Press, 2015).

[148] S. Ruder, "An overview of gradient descent optimization algorithms", arXiv:1609.04747 (2016).

[149] A Griewank, "On automatic differentiation. Preprint ANL/MCS-P10-1088", Mathematical Programming: Recent Developments and Applications **6**, 83 (1989).

[150] 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", in 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16) (2016), pp. 265–283.

[151] A. Paszke et al., "Pytorch: an imperative style, high-performance deep learning library", in *Advances in neural information processing systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., 2019), pp. 8024–8035.

[152] *JuliaDiff: Differentiation Tools in Julia*, `https://github.com/JuliaDiff/`.

[153] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, and S. Wanderman-Milne, *JAX: composable transformations of Python+NumPy programs*, 2018.

[154] "PennyLane: Automatic differentiation of hybrid quantum-classical computations", arXiv:1811.04968 (2018).

[155] G. E. Hinton, "A practical guide to training restricted boltzmann machines", in *Neural networks: tricks of the trade: second edition*, edited by G. Montavon, G. B. Orr, and K.-R. Müller (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012), pp. 599–619.

[156] A. Krizhevsky, *Learning multiple layers of features from tiny images*, tech. rep. (University of Toronto, 2009).

[157] G. E. Hinton, "Training Products of Experts by Minimizing Contrastive Divergence", Neural Computation **14**, 1771 (2002).

[158] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes", 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings (2013).

[159] D. P. Kingma and M. Welling, "An Introduction to Variational Autoencoders", Foundations and Trends® in Machine Learning **12**, 307 (2019).

[160] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially Learned Inference", in 5th International Conference on Learning Representations (2017).

[161] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", in 4th International Conference on Learning Representations (2015).

[162] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric", in Proceedings of The 33rd International Conference on Machine Learning, Vol. 48, edited by M. F. Balcan and K. Q. Weinberger, Proceedings of Machine Learning Research (2016), pp. 1558–1566.

[163] I. 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. 27 (2014).

[164] S. Arora and Y. Zhang, "Do GANs actually learn the distribution? An empirical study", arXiv:1706.08224 (2017).

[165] D. Rezende and S. Mohamed, "Variational Inference with Normalizing Flows", in Proceedings of the 32nd International Conference on Machine Learning (2015), pp. 1530–1538.

[166] C. Monterola and C. Saloma, "Solving the nonlinear Schrödinger equation with an unsupervised neural network: estimation of error in solution", Optics Communications **222**, 331 (2003).

[167] V. Cimini, I. Gianani, N. Spagnolo, F. Leccese, F. Sciarrino, and M. Barbieri, "Calibration of quantum sensors by neural networks", Phys. Rev. Lett. **123**, 230502 (2019).

[168] V. Gebhart and M. Bohmann, "Neural-network approach for identifying nonclassicality from click-counting data", Physical Review Research **2**, 023150 (2020).

[169] V. Cimini, M. Barbieri, N. Treps, M. Walschaers, and V. Parigi, "Neural networks for detecting multimode Wigner-negativity", arXiv **125**, 160504 (2020).

[170] M. Krenn, R. Fickler, M. Fink, J. Handsteiner, M. Malik, T. Scheidl, R. Ursin, and A. Zeilinger, "Communication with spatially modulated light through turbulent air across Vienna", New Journal of Physics **16**, 113028 (2014).

[171] 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", Applied Physics Reviews **7**, 021404 (2020).

[172] 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).

[173] P. Cha, P. Ginsparg, F. Wu, J. Carrasquilla, P. L. McMahon, and E. A. Kim, "Attention-based Quantum Tomography", arXiv:2006.12469 (2020).

[174] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila, "Noise2Noise: learning image restoration without clean data", in Proceedings of the 35th International Conference on Machine Learning, Vol. 80, edited by J. Dy and A. Krause (2018), pp. 2965–2974.