View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repository@Nottingham

Quantum Machine Intelligence (2019) 1:5–15
https://doi.org/10.1007/s42484-019-00003-8

RESEARCH ARTICLE

A hybrid machine learning algorithm for designing quantum experiments

L. O'Driscoll[1] · R. Nichols[1] · P. A. Knott[1]

Received: 11 December 2018 / Accepted: 28 February 2019 / Published online: 27 March 2019
© The Author(s) 2019

Abstract

We introduce a hybrid machine learning algorithm for designing quantum optics experiments to produce specific quantum states. Our algorithm successfully found experimental schemes to produce all 5 states we asked it to, including Schrödinger cat states and cubic phase states, all to a fidelity of over 96%. Here, we specifically focus on designing realistic experiments, and hence all of the algorithm's designs only contain experimental elements that are available with current technology. The core of our algorithm is a genetic algorithm that searches for optimal arrangements of the experimental elements, but to speed up the initial search, we incorporate a neural network that classifies quantum states. The latter is of independent interest, as it quickly learned to accurately classify quantum states given their photon number distributions.

Keywords Machine learning · Genetic algorithm · Artificial intelligence · Quantum state engineering · Quantum optics

1 Introduction

As artificial intelligence (AI) and machine learning develop, their range of applicability continues to grow. They are now being utilised in the fast-growing field of quantum machine learning (Dunjko and Briegel 2017; Biamonte et al. 2017; Schuld et al. 2015), with one particular application demonstrating that AI is an effective tool for designing quantum physics experiments (Knott 2016; Krenn et al. 2016; Melnikov et al. 2018; Arrazola et al. 2019; Sabapathy et al. 2018). In this vein, here we introduce a hybrid algorithm that designs and optimises quantum optics experiments for producing a range of useful quantum states, including Schrödinger cat states (Ourjoumtsev et al. 2007; Huang et al. 2015; Etesse et al. 2015) and cubic phase states (Gottesman et al. 2001).

The core of our algorithm, named AdaQuantum[1] and introduced in (Nichols et al. 2018) and (Knott 2016), uses a genetic algorithm to search for optimal arrangements of quantum optics experimental equipment. Any given arrangement will output a quantum state of light, and the algorithm's task is to optimise the arrangement to find states with specific properties. To assess the suitability of a given state, we require a *fitness function* that takes as input a quantum state and outputs a number—the *fitness value*—that quantifies whether the state has the properties we desire or not. Our previous works largely focused on quantum metrology, where our algorithm found quantum states with substantial improvements over the alternatives in the literature (Nichols et al. 2018; Knott 2016). While in Nichols et al. (2018) and Knott (2016) the fitness function assessed the phase-measuring capabilities of the states, in this paper, instead, we look at producing a range of useful and interesting states (introduced below) to a high fidelity, and hence we use as our fitness function the fidelity to our target states.

✉ P. A. Knott
Paul.Knott@nottingham.ac.uk

1 Centre for the Mathematics and Theoretical Physics of Quantum Non-Equilibrium Systems (CQNE), School of Mathematical Sciences, University of Nottingham, University Park, Nottingham, NG7 2RD, UK

[1]The algorithm, AdaQuantum, is named after Ada Lovelace, the worlds first computer programmer, and resident of Nottingham, where our own algorithm was born.

The search space for our genetic algorithm is huge, which means that typically the algorithm has to simulate and evaluate a vast number of quantum optics experiments in order to finally find strong solutions. This introduces a major challenge in our work: the speed, efficiency, and effectiveness of our algorithm depend in a large part on simulating and evaluating the experiments as quickly and accurately as possible. If short-cuts could be found that allow a given experimental set-up to be evaluated approximately without the full simulation being performed, then this would greatly improve our optimisation: the approximate evaluations would provide a quick guide for the search to progress in the right direction, and subsequently the exact evaluation, using the full quantum simulation, would provide the precise fitness value, thus validating and fine-tuning the search. Efficient computational models of this sort for approximating the fitness function are often known as surrogates or meta-models (Jin 2011).

In this paper, we use one such approximation during the evaluation stage: instead of explicitly calculating the fidelity of our output state to the desired states, we use a deep neural network (DNN) that learns to classify what type of state has been outputted, and approximately how close this state is to the target state. Explicitly calculating the fidelity the large number of times required to run our algorithm takes a surprisingly long time, whereas using our DNN, we get a useful, albeit modest, speed-up. While in the work presented here the speed-up is only small, our results can be seen as a proof of principle, paving the way for much more demanding fitness functions, such as the Bayesian mean squared error (Rubio et al. 2018; Nichols et al. 2018), to be approximately evaluated in this way. Once the DNN has guided the search in the right direction, the fidelity is then used to provide the exact fitness function (see Section 4.1 for the full description of our algorithm).

Our DNN for classifying quantum states is likely to be of independent interest, as it quickly learnt to recognise a quantum state just given its photon number distribution. This has potential uses in a wide range of applications, such as quantum computing or quantum cryptography, where it can be valuable to quickly recognise what type of quantum state has been produced.

Our hybrid algorithm, utilising a genetic algorithm to perform the automated search and a neural network to classify the outputted quantum states, quickly and efficiently found new quantum optics experiments to produce a range of useful quantum states to a high fidelity. This complements the recent work of Arrazola et al. (2019) who used machine learning to also find quantum optics experiments to produce a range of specific states, with one key difference being that our experiments are specifically designed to be practical, which comes at the cost of our states being of a lower fidelity than those in Arrazola et al. (2019).

This paper is structured as follows. First, we introduce a general experimental scheme for engineering optical quantum states, and introduce the various quantum optics experimental elements that are typically used. We then introduce the main goal of our work: finding experimental set-ups to create specific quantum states. We then describe the methods we use: firstly the genetic algorithm, which searches through different arrangements of the experimental equipment to find those that produce our desired states; and secondly our neural network, which classifies quantum states and enables a speed-up in our search algorithm. Finally, we present our results.

2 Quantum state engineering with light

The state-engineering scheme we use is shown in Fig. 1. Firstly, two input states, $|\psi_{0a}\rangle$ and $|\psi_{0b}\rangle$, are input into the two modes. The two modes then pass through a sequence of operators $\hat{O}_i$ where $i = 1, .., m$, and the final step is to perform a heralding measurement on one mode, producing the final state $|\psi_f\rangle$. With appropriate choices of input states, operators, and measurements this scheme is able to replicate a wide range of the quantum state engineering protocols in the literature, such as Ourjoumtsev et al. (2007), Huang et al. (2015), Etesse et al. (2015), Bartley et al. (2012), and Gerrits et al. (2010). The main difference between the present paper and the usual methods is that we do not *choose* the input states, operators, and measurements, but instead we use a genetic algorithm to *search* for states with the desired properties (see Section 4.1 for details of the algorithm).

Our objective is to find *practical* experiments, and so we construct our state engineering protocols from elements of an experimentally-ready toolbox of quantum optics states, operators, and measurements, which is summarised in the table below.

Input states	Apparatus
Fock, $\|n\rangle$	Beam splitter, $\hat{U}_T$
Coherent, $\|\alpha\rangle$	Phase shift, $e^{i\hat{n}\theta}$
Squeezed, $\|z\rangle$	Displacement, $\hat{D}(\alpha)$
TMSV, $\|z\rangle_{12}$	Number measurement, $\|n\rangle\langle n\|$

Here, we only introduce the most important details of the toolbox; more details can be found in Appendix 1. Firstly, the input states we include are the single-mode squeezed vacuum $|z\rangle$, the two-mode squeezed vacuum (TMSV) $|z\rangle_{12}$, the coherent state $|\alpha\rangle$, and Fock states $|n\rangle$; the parameters z, α and n are constrained by what is possible experimentally (Mehmet et al. 2011; Müller et al. 2015; Claudon et al. 2010; Morin et al. 2012; Ourjoumtsev et al. 2006; Huang 2015).

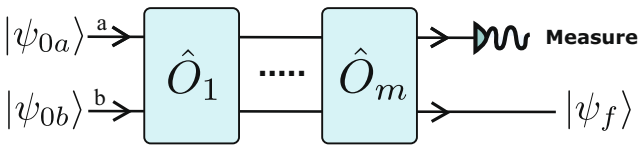

Fig. 1 The state engineering scheme we consider begins with two input states, $|\psi_{0a}\rangle$ and $|\psi_{0b}\rangle$, which are input into the two modes. The states then subsequently pass through a number of operators $\hat{O}_i$. To produce the final quantum state $|\psi_f\rangle$, a heralding measurement is performed on one mode

Next are the operators, of which the most important is the beam splitter $\hat{U}_T$, where T is the probability of transmission, which serves to mix and entangle the two modes, enabling more exotic and useful states to be produced when part of the entangled state is measured. The other operators we use are the displacement operator $\hat{D}(\alpha)$ and the phase shift $e^{i\hat{n}\theta}$. In addition, we include the identity operator $\hat{\mathbb{I}}$, because we are promoting the easiest-to-implement schemes, which would contain as many identities as possible.

The final step of the state engineering scheme is to perform a heralding measurement on one mode of the final state. If, for example, we wish to herald on the one photon state, we can perform a photon number resolving detection (PNRD), and only keep the output state if one photon is detected. A measurement outcome of one photon therefore *heralds* the desired final state. The heralding measurement corresponds to acting on the two-mode, pre-measurement state with $\langle 1| \otimes \hat{\mathbb{I}}$, followed by normalisation. We are then left with the single mode final state $|\psi_f\rangle$. Recent progress in PNRD (for example using transition edge sensors (Humphreys et al. 2015; Gerrits et al. 2010)) has enabled detections of larger numbers of photons possible, so here we allow for heralding number measurements of up to 8 photons.

Many more states, operators and measurements can be included in this toolbox. As discussed in our paper that fully introduces our algorithm (Nichols et al. 2018), the algorithm is design with flexibility in mind, so it is straightforward for more elements to be added (or removed) from the toolbox, depending on the available equipment or desired goal. But here we consider a simplified toolbox to discover whether such a limited toolbox of experimentally viable elements can still produce a range of quantum states to a high fidelity. In Nichols et al. (2018), we include experimental noise, but in this work, we stick to pure states and perfect operators/measurements.

3 Finding experiments to engineer specific quantum states

The task we set our algorithm, AdaQuantum, is to find experimental designs to produce a range of specific "target" states to a high fidelity (where the fidelity between two pure states $|\psi\rangle$ and $|\phi\rangle$ is defined as $F \equiv |\langle\psi|\phi\rangle|^2$). The target states are shown in the table below (normalisation is omitted): these states have a range of properties and are studied and used by both theorists and experimentalists. Figure 2 shows an artist's impression of our algorithm, AdaQuantum, producing a range of quantum states.

Name	State
Cat	$\|\text{cat}\rangle \backsim \|\alpha\rangle + e^{i\theta}\|-\alpha\rangle$
Squeezed cat	$\hat{S}(z)\|\text{cat}\rangle$
Zombie	$\|\alpha\rangle + \left\|e^{2\pi i/3}\alpha\right\rangle + \left\|e^{4\pi i/3}\alpha\right\rangle$
ON	$\|0\rangle + \delta\|n\rangle$
Cubic phase	$\exp\left(i\gamma\hat{q}^3\right)\hat{S}(z)\|0\rangle$

Here, $\delta, \gamma \in \mathbb{R}$; $\hat{S}(z)$ is the squeezing operator, given by $\hat{S}(z) = \exp\left[\frac{1}{2}(z^*\hat{a}^2 - z\hat{a}^{\dagger 2})\right]$, where $z \in \mathbb{C}$ and $\hat{a}^\dagger$ and $\hat{a}$ are the creation and annihilation operators, respectively (Barnett and Radmore 2002); and $\hat{q}$ is the position quadrature operator, given by $\hat{q} = (\hat{a} + \hat{a}^\dagger)/\sqrt{2}$.

The first state we search for is the optical Schrödinger cat state. This state is inspired by Schrödinger's famous thought experiment in which he proposed to put a macroscopic system—a cat—in a superposition of two distinct states (Schrödinger 1935). The implications of this thought experiment still spark heated debate and disagreement, but what escapes controversy is that it would be both important and interesting to create a macroscopic superposition of two distinct states. The optical Schrödinger cat state is moving

Fig. 2 An artist's impression of our algorithm, AdaQuantum, which designs experiments for engineering quantum states. The algorithm is designed for flexibility and can produce a wide range of quantum states: the illustration depicts, among others, the production of a Schrödinger cat state, a three-headed cat state (Lee et al. 2015), and a GKP state (Gottesman et al. 2001). Artwork by Joseph Namara Hollis

towards this goal, as it is a superposition of two distinct coherent states, $|\alpha\rangle$ and $|-\alpha\rangle$. While the magnitude of α so far produced in experiments is far from making the state macroscopic, the optical Schrödinger cat state might eventually be produced as a macroscopic superposition, given that $|\alpha\rangle$ is the state produced from a "perfect" laser. Optical Schrödinger cat states (often referred to as just "cat states") have been produced in a number of experiments, such as Ourjoumtsev et al. (2007), Huang et al. (2015), and Etesse et al. (2015).

The next two states are derived from the Schrödinger cat state. First, we can consider *squeezing* a cat state by applying the squeezing operator, as shown in the table. The resulting *squeezed cat state*, whilst in one sense being more exotic than the cat state, is not necessarily more difficult to produce experimentally (Huang et al. 2015; Etesse et al. 2015). The squeezed cat state can, for example, provide substantial enhancements in quantum metrology (Knott et al. 2016). Next, instead of making a superposition of *two* coherent states with different phases as in the cat state, we can make a superposition of *three*, with the relative phases now differing by $2\pi/3$. Such a state can be called a three-headed cat state (Lee et al. 2015; Jiang et al. 2016), but here we prefer the name *zombie cat state*, as it represents the superposition of three distinct macroscopic states: dead, alive, and undead.

We used the algorithm to search for cat states, squeezed cat states, and zombie cat states, with the following parameters: $\alpha \in [0, 2]$, $\theta \in [0, 2\pi]$, and $z \in \mathbb{C}$ with $|z| \in [0, 1.4]$.

Next, we consider the *ON state*, which is a superposition of the vacuum $|0\rangle$ with an n-photon state $|n\rangle$. By controlling the relative weighting δ, it has been shown that the quantum Fisher information of this state, which is often used to quantify the state's phase-measuring potential, can be made arbitrarily large whilst keeping the photon number arbitrarily small (Rivas and Luis 2012) (but this state cannot actually achieve infinitely precise measurements (Hall and Wiseman 2012; Hall et al. 2012)). These states are also important for continuous variable quantum computation because they can be used as a resource to implement cubic phase gates (Sabapathy and Weedbrook 2018). The latter enable universal quantum computation (Kok and Lovett 2010) (a non-Gaussian gate such as a cubic phase gate is essential, as Gaussian gates alone cannot be universal (Kok and Lovett 2010)). The ON states we searched for have $\delta \in [0, 1]$ and $n \in [1, 10]$.

Another state useful for continuous variable quantum computation is the *cubic phase state*, which too can be used as a resource to enable cubic phase gates (Gottesman et al. 2001; Sabapathy and Weedbrook 2018; Ghose and Sanders 2007; Takagi and Zhuang 2018; Gu et al. 2009). The cubic phase states we searched for have $\gamma \in [0, 0.25]$ and $z \in \mathbb{R}$ with $z \in [0, 1.4]$. Note that "true" cubic phase states are only obtained for infinite squeezing.

4 Methods

4.1 Using a genetic algorithm to design experiments

We will first introduce the general method of using a genetic algorithm (GA) to design experiments, before describing in more detail how our specific algorithm works. In order to use a GA, we must encode each possible arrangement of states, operators and measurements into a vector, which is known as a *genome*. The genome contains all the information necessary to re-construct a given experimental setup, including all the parameters of the experimental elements. The GA then starts by creating a collection of genomes, which together are known as the *population*. Next, the experimental setup corresponding to each genome is simulated, and the fitness function for each output state is evaluated. The fitness function must take a quantum state as an input, and output a number, the fitness value. The latter quantifies whether the states has the properties we desire or not. In this paper, our fitness function will be a measure of how close the output state is to one of those states we are searching for, which are introduced in Section 3. But as discussed in Nichols et al. (2018), the flexibility of our algorithm allows for a wide range of fitness functions to be used to find quantum states for any number of applications.

After the population of genomes is assessed for their fitness, the "fittest" genomes, i.e. the genomes with the largest fitness values, are then selected, and a new population of genomes (the *children*) is generated by mixing some of the genomes together (*crossover*), by randomly modifying (*mutating*) others, and keeping some genomes unchanged (the *elite children*). This next population should, in principle, be comprised of genomes that are "fitter" than before. This process repeats through a number of *generations*, until it is unlikely that any more generations will result in improvements. At this stage, if the algorithm has been designed appropriately, then the fittest genomes will encode optimised solutions. Through this process, our GA evolves quantum experiments that are highly suited to the task at hand. A flow chart of our algorithm is given in Fig. 3.

In order to assess each genome, we must simulate the quantum optics experiment that this genome encodes, then evaluate the fitness function on the output state. As introduced in Nichols et al. (2018), our simulation of the quantum optics experiments utilises a number of techniques to increase its efficiency. The result is a powerful simulation that allows us to simulate experiments with a truncation of up to 170 photons (in two modes) in the order of seconds,

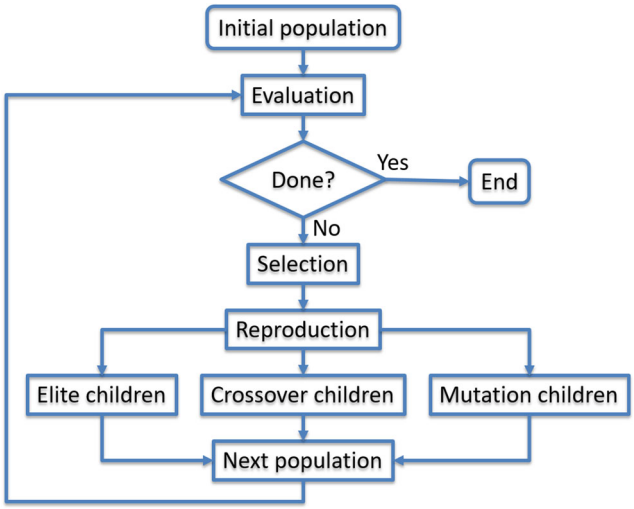

Fig. 3 Flowchart of a genetic algorithm. We first create an initial *population* of genomes, we evaluate these using the fitness function, and then select the "fittest" genomes. We then create a new population, named the *children*, using three methods, *elite*, *crossover*, and *mutation*, which are described in the main text. The process then repeats until we have found a genome that fulfils our requirements, or until some exit criteria are met (e.g. given by the maximum allowed running time)

thus allowing a broad range of exotic states—with important contributions at high photon numbers—to be assessed.

4.1.1 Our three-stage algorithm

One significant challenge in our approach is that to simulate each experiment accurately we need to truncate the Hilbert space at a high number, but the higher the truncation is, the slower the algorithm will run. Our search space of quantum experiments is huge, and to do an effective search, we need to evaluate a large number of different experiments, but this becomes increasingly more computationally expensive for larger truncations. To overcome this, our GA, introduced in Nichols et al. (2018), has three stages:

1. A large number of random genomes are created and evaluated. In this stage, the truncation of the Hilbert space is small (we vary this, but it is often around 30), and therefore the quantum simulation is only approximate. But it is fast. A collection of genomes with the best fitness values are selected for the next stage. While the simulation here is only approximate, it still provides a valuable guide as to where the GA is likely to find experiments with higher fitness values—this stage seeds the GA in the subsequent stage with a substantially better starting population than picking purely at random.
2. We run MATLAB's inbuilt GA (https://uk.mathworks.com/help/gads/index.html) with a medium-sized population. The simulation is less approximate than stage 1 (because the truncation is larger, around 80), and hence slower. This stage only runs for a set number of generations, usually 10. This stage performs a medium-speed global search and provides the final stage with a strong population.
3. In the final stage, the simulation is accurate but slow. In this stage, the fitness function will first simulate the circuit specified by the input genome at a very low truncation, then repeat this, increasing the truncation on each iteration, either until the average number of photons in the final state converges or until the maximum truncation is reached (where the maximum truncation is specified by the user). This ensures the results are reliable and accurate, while still running in a reasonable time. Here, we again use MATLAB's inbuilt GA (https://uk.mathworks.com/help/gads/index.html), but the population is smaller, and the search is more local.

4.1.2 Overcoming challenges in evaluating our fitness function

As our task is to find specific states, the obvious fitness function here is to evaluate the fidelity between the target state and the state outputted in each simulation. For example, if we wish to find a cat state, then we can calculate the fidelity to a cat state. But *which* cat state should we be evaluating against? An (unnormalised) cat state given by $|\alpha\rangle + e^{i\theta}|-\alpha\rangle$ has three real-valued parameters (the value of θ and the magnitude and phase of α), so we should compare against every combination of these parameters. Even restricting to small cat states, e.g., $|\alpha| \leq 2$, and discretising the parameters, we still have a large number of cat states to compare against (in the order of 10^5 for this paper). This is not a problem in stages 2 and 3 of our algorithm, because the run-time is dominated by simulating each experiment. But in stage 1, where the truncation is small, the overhead from evaluating the fidelity becomes significant.

This problem is exacerbated when we consider how our algorithm will commonly be used to design new quantum experiments. Generally, we expect a user to specify which states, operators, and measurements they have available, and then to run the algorithm to find which states, and to what fidelity, can be produced with their given equipment. In this case, in stage 1 of the algorithm, ideally, we would search for *all* of the quantum states of interest (e.g. the 5 classes used in this paper) simultaneously, but this would require a vast number of fidelities to be computed for each simulation (around 10^6 for this paper).

We overcome this problem by using a deep neural network (DNN) to classify each quantum state, thus bypassing the need to calculate each fidelity. The DNN is

introduced in detail in the next section, but it suffices for now to consider the DNN as a black box for which we input a quantum state, and the DNN outputs a classification. More specifically, we input a quantum state into the DNN, and the DNN will output a probability distribution as to whether the state is a cat state, squeezed cat state, zombie cat state, ON state, cubic phase state, or none of the above. For example, if we input the state $|\alpha = 1\rangle + |\alpha = -1\rangle$ into the DNN, it should give the output "cat state". We add a class of state that we call "other", so that we know whenever a simulation produces a state that is not close to any of our desired states. See the next section for more details of the DNN, including how we train it to perform the classification.

When used in stage 1 of our algorithm, this method of using a DNN has two significant advantages over the alternative method of calculating the fidelities. Firstly, once the DNN is trained to classify states it runs substantially faster than calculating the fidelity. Secondly, the DNN classifies for all 5 states (or more) simultaneously. We therefore use the DNN as follows: In stage 1, we evaluate a large number of genomes, using the DNN to classify each. We then take a number of the genomes that come closest to producing cat states, according to the DNN, and then send these genomes through stages 2 and 3, using the fidelity as the fitness function (population sizes for the runs of AdaQuantum in this paper are given in Appendix 2). We then repeat stages 2 and 3 for the other 4 states. When used in this way in stage 1, the DNN is around two orders of magnitude faster than calculating all the fidelities. The overall structure of our hybrid algorithm that incorporates both the DNN and the GA is shown in Fig. 4.

Our algorithm AdaQuantum is available free-to-use on GitHub (https://github.com/paulk444/AdaQuantum).

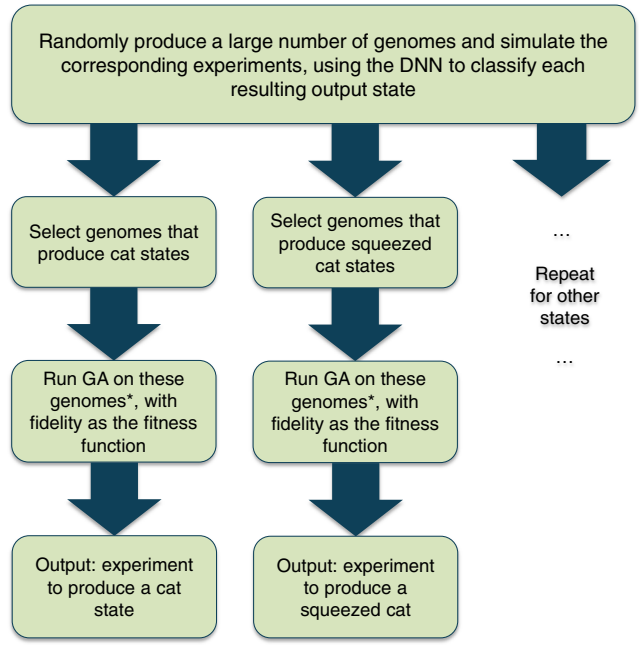

Fig. 4 A flowchart of the overall structure of our hybrid algorithm that incorporates both a deep neural network (DNN) and a genetic algorithm (GA) to design experiments to produce a range of quantum states. *The GA runs in stages 2 and 3 of the 3-stage algorithm discussed in Section 4.1.1 of the main text

4.2 A neural network for classifying quantum states

Having explained *how* and *why* we choose to utilise a DNN for classifying quantum states, we will now introduce neural networks, and explain how we construct and train ours.

A classifier deep neural network (DNN) is a machine learning technique used to classify data into a set of classes. In our case, we input a quantum state, and the DNN will output a probability distribution as to whether the state is a cat state, squeezed cat state, zombie cat state, ON state, cubic phase state, or none of the above. Data is input to the DNN as a vector $\vec{x} \in \mathbb{R}^n$. The network is built up of layers of "neurons". In a given layer, each neuron is assigned a value that is calculated by first taking a linear combination of the values in the previous layer, then doing a non-linear activation function. Mathematically, this is expressed as:

$$\vec{x}_{i+1} = \sigma\left(M_i \vec{x}_i + \vec{b}_i\right)$$

where $\vec{x}_{i+1}$ are the neuron values for the next layer, M_i is matrix of *weights*, $\vec{b}_i \in \mathbb{R}^n$ is a *bias* vector, and σ is the activation function, which is applied element-wise.

Our final layer has 6 neurons, each corresponding to a class of state; this layer uses a different activation function than the rest of the network and produces a probability distribution, corresponding to the probability that the network thinks the input state was each of the classes (e.g. if the first neuron in the output layer is 0.9, the network has determined that the input state has a 90% probability of being a cat state). The neuron with the maximum probability therefore indicates which class the input state is predicted to belong to. The values of M_i and $\vec{b}_i$ must be *learnt* by the network so that it produces the desired output—this is achieved by training, which is described below. A visualisation of the neural network is shown in Fig. 5.

To train the neural network, we first need some labelled training data, which should take the form of a set of quantum states, and a label for each state (the label gives the "class" of each state, for example a "cat state"). This was generated by sampling parameter values for the states from a random distribution. In other words, we first create a cat state with ramdomly generated parameters (where the parameters for the cat state are α and θ). By associating with this state the label "cat state", we now have our first piece of training data. This is repeated for the cat state a number of times

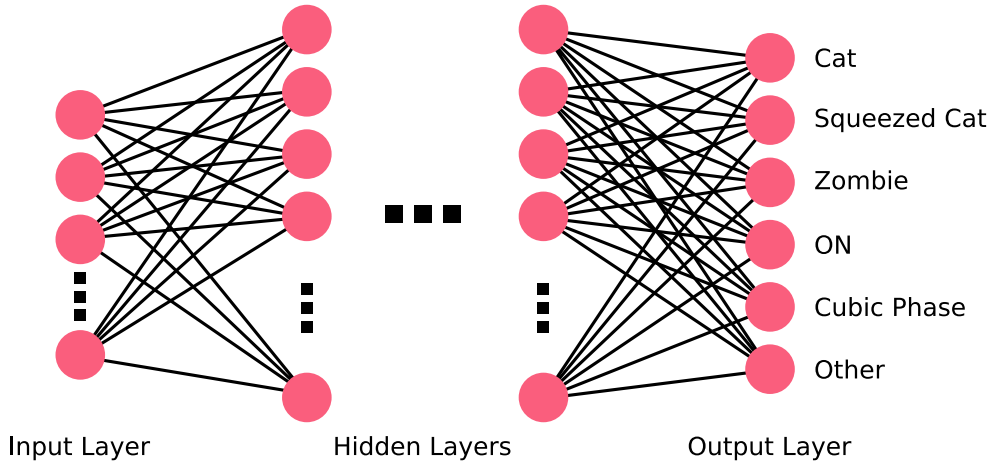

Fig. 5 Our deep neural network (DNN) for classifying quantum states. We input the number distribution of a quantum state, and the DNN outputs a probability distribution corresponding to the probabilities that the inputted state was one of a fixed set of classes (introduced in Section 3): cat state, squeezed cat state, zombie cat state, ON state, cubic phase state, or none of the above

(approx. 1700 in this paper), before moving on to the other classes of quantum states. We repeat this process to create the testing data. Once the training (and testing) data has been generated, this can be used to train (and test) the DNN. The goal of the DNN is therefore to take in a given state, and correctly tell us which of the classifications this state belongs to.

When producing quatum states for the training and testing data, the coefficients of the quantum states in the Fock basis were then calculated using either the analytic expressions (where possible), or by matrix representations of operators acting on the vacuum state (in both cases using a truncated Hilbert space) (Barnett and Radmore 2002). Some states with random coefficients were also generated (labelled *other*). The generated training states are inputted to the neural network and a loss function (the softmax cross entropy) is computed using the values of the output layer, and the actual label of the state. The aim of training the network is to modify the values of the biases and weights so that this loss function is minimised; to achieve this, we used an optimisation algorithm called Adam (Kingma and Ba 2014), a variation on the stochastic gradient descent method commonly used in machine learning.

Data must be inputted to the neural network as a real, finite-dimensional vector, but our quantum states belong to a complex, infinite-dimensional Hilbert space. We choose to convert the complex coefficients of each state (in the truncated Fock basis) to real numbers by taking the modulus of the Fock coefficients (throughout the paper we refer to the set of moduli of the Fock coefficients as the "number distribution"). We tried inputting the phase information as extra inputs to the DNN, but this did not improve the accuracy.

A common problem encountered in machine learning is overfitting, which occurs when the model learnt by the neural network fits to outliers in your training data set, but does not generalise well (as an extreme example, if the DNN has enough free parameters it can "memorise" the whole training set given enough training iterations). One method to detect this is to split your data set in two: training and testing. The network is trained using only the data in the training set. Its accuracy can then be evaluated on the testing data set (which the network has never seen before). If the accuracy of the network on the testing set is significantly less than its accuracy on the training set, then it is likely that overfitting has occurred. Using larger training/testing data sets can help to avoid overfitting, as well as techniques such as dropout and regularisation, all of which we experimented with here.

We implemented our DNN in TensorFlow (Abadi et al. 2016). We used a training data set containing 10,000 states, and a testing data set of 3000 states, where approximately one-sixth of the states in each data set belong to each class. After some experimentation, we settled on a DNN consisting of 3 fully connected hidden layers, comprising 25, 25, and 10 neurons, respectively. After 5000 steps (*epochs*) of training, the network classified the test data with an accuracy of 99.3%. Our DNN is available on GitHub (https://github.com/lewis-od/Quantum-Optics) (which includes the code for generating states for the training data).

A standard metric to monitor how well a neural network is classifying data, aside from the accuracy, is the confusion matrix. This is a 6×6 matrix (in our case), where the rows represent the actual classes of the states, and the columns represent the classes predicted by the network. The entry in the ith row and jth column shows how many states of class

Table 1 The results found by running AdaQuantum to find experimental schemes to produce 5 different quantum states

Target state	Target parameters	Experiment	Experiment parameters	Fidelity
Cat	$\alpha = -2 - i, \theta = 0$	$\langle 6 \vert \hat{U}_{T_1} \vert z_1, z_2 \rangle$	$z_1 = 0.701e^{4.10i}, z_2 = 0.156e^{0.847i}, T_1 = 0.407$	99.85%
Squeezed cat	$\alpha = -0.2\dot{2} + 0.2\dot{2}i, z = 1.09 + 0.47i$	$\langle 4 \vert \hat{U}_{T_2} \vert z_3 \rangle_{12}$	$z_3 = 1.28e^{0.422i}, T_2 = 0.499$	99.78%
Zombie	$\alpha = -0.28 + 0.53i$	$\langle 4 \vert \hat{D}_1(\alpha_1)\hat{U}_{T_3} \vert z_4, 0 \rangle$	$z_4 = 1.26e^{2.64i}, T_3 = 0.724, \alpha_1 = 2.16e^{0.265i}$	96.84%
ON	$n = 2, \delta = 0.32$	$\langle 8 \vert \hat{U}_{T_4} \vert z_5 \rangle_{12}$	$z_5 = 0.985e^{6.28i}, T_4 = 0.606$	97.77%
Cubic phase	$\gamma = 0.05, z = 0.29$	$\langle 5 \vert \hat{U}_{T_5} \vert z_6, 1 \rangle$	$z_6 = 0.586e^{3.14i}, T_5 = 0.612$	96.11%

Normalisation constants are omitted. Also, in the heralding measurements, we omit the identity operator that acts on the remaining mode, i.e. $\langle n \vert \equiv \langle n \vert \otimes \mathbb{I}$, where $\langle n \vert$ acts on the first mode and $\mathbb{I}$ acts on the second mode. The final column shows the fidelity of the output state with the target state. All of these states can be made using present day experimental equipment, and all experimental designs produced states with a fidelity of at least 96% with their target state. See Fig. 6 for a schematic of the experiment to produce an ON state

i were classified as class j. The confusion matrix calculated using our test data was:

$$\begin{array}{c} \\ \text{Cat} \\ \text{Sq.-Cat} \\ \text{Zombie} \\ \text{ON} \\ \text{CP} \\ \text{Other} \end{array} \begin{array}{c} \begin{array}{cccccc} \text{Cat} & \text{Sq.-Cat} & \text{Zombie} & \text{ON} & \text{CP} & \text{Other} \end{array} \\ \begin{pmatrix} 497 & 3 & 0 & 0 & 0 & 0 \\ 0 & 500 & 0 & 0 & 0 & 0 \\ 1 & 0 & 499 & 0 & 0 & 0 \\ 0 & 0 & 1 & 499 & 0 & 0 \\ 5 & 7 & 3 & 0 & 485 & 0 \\ 0 & 0 & 0 & 1 & 0 & 499 \end{pmatrix} \end{array}$$

From this, we can see that the network is best at classifying squeezed cat states; however, it has also classified 10 states incorrectly as squeezed cat states. It is also clear that the network is worst at classifying cubic phase states, which is to be expected as they have the most complex structure of all the states we are interested in. Also note that no states were incorrectly classified as "other", meaning we were unlikely to falsely classify states that we are looking for as being useless.

5 Results

We ran AdaQuantum to find the 5 states introduced in Section 3, and obtained the results in Table 1. The hyperparameters and settings of the genetic algorithm that were used to obtain these results are given in Appendix 2. All of the states were found to a fidelity above 96%, and two of them were over 99.7%.

As an example, to produce an ON state to a fidelity of 97.77%, we first send a two-mode squeezed vacuum state through a beam splitter. This creates the two-mode state $\hat{U}_{T_4} \vert z_5 \rangle_{12}$. Next, we do an 8-photon heralding measurement on the first mode of the output. As explained in Appendix 1, we can calculate the output state from such a heralding measurement by acting with $\langle 8 \vert \otimes \mathbb{I}$, where $\mathbb{I}$ is the identity (acting on the second mode). Finally, we normalise this state. The full ON state in this example is therefore given by

$$\mathcal{N} \left(\langle 8 \vert \otimes \mathbb{I} \right) \hat{U}_{T_4} \vert z_5 \rangle_{12} , \tag{1}$$

where $\mathcal{N}$ is the normalisation constant. Note that in Table 1, the normalisation constant and the identity are omitted for all states. The single-mode state given in Eq. 1 has a 97.77% fidelity to the target ON state. A schematic of this experiment is shown in Fig. 6.

Similar schemes—with similar simplicity—were found for all 5 states. As discussed above, all of the experimental elements used here are accessible with current technology, with perhaps the biggest challenge in producing the states in Table 1 being the larger-number heralding measurements (Humphreys et al. 2015; Gerrits et al. 2010). The purpose of this paper is to introduce the method of using such an algorithm, and demonstrate its effectiveness; future work will undertake a deeper analysis of the capabilities of AdaQuantum and the experiments and states that it has produced.

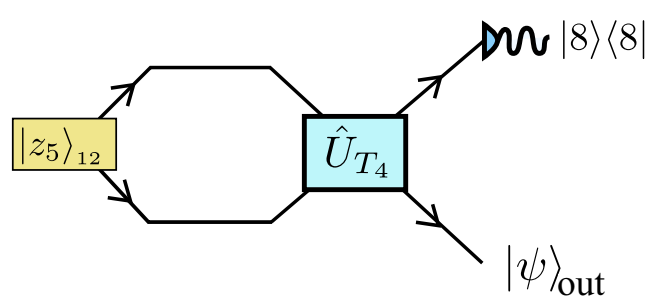

Fig. 6 A sample experiment designed by AdaQuantum: this arrangement of quantum optics elements creates an ON state to a fidelity of 97.77%. See Table 1 for details and parameter values. As desribed in the main text, this state is created by sending a two-mode squeezed vacuum state ($\vert z_5 \rangle_{12}$) through a beam splitter (U_{T_4}), and then acting with the POVM $\vert 8 \rangle \langle 8 \vert$ on the first mode of this two-mode state

6 Conclusion

We have introduced a hybrid machine learning algorithm for designing quantum optics experiments. A genetic algorithm was used to search for optimal arrangements of experimental elements that produce a range of useful and interesting optical quantum states, and a deep neural network was used to speed up the evaluations of each experimental arrangement by quickly and accurately classifying quantum states. Combining these techniques, our algorithm found experimental arrangements to produce all 5 states we asked it to, all to a high fidelity. This demonstrates the power and flexibility of the technique of using methods from artificial intelligence and machine learning to design and optimise quantum physics experiments.

Acknowledgements We thank Joseph Namara Hollis for the artwork in Fig 2. We acknowledge discussions with Gerardo Adesso, Ryuji Takagi, Tom Bromley and Ender Özcan. P.K. acknowledges support from the Royal Commission for the Exhibition of 1851. L.O'D. was supported by a vacation bursary from EPSRC.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

Appendix 1. Quantum optics toolbox details

Input states The squeezed vacuum is given by $|z\rangle = \hat{S}(z)|0\rangle$ where the squeezing operator is $\hat{S}(z) = \exp\left[\frac{1}{2}(z^*\hat{a}^2 - z\hat{a}^{\dagger 2})\right]$ and $z = re^{i\theta_s}$ where r is the (positive and real) amplitude, $\theta_s \in [0, 2\pi]$ is the squeezing angle and $\hat{a}$ ($\hat{a}^\dagger$) is the annihilation (creation) operator. Squeezed states can be made up to $r \approx 1.4$, but this is extremely challenging experimentally so we set the limit to $r = 1.3$ (Mehmet et al. 2011). Similarly, the two-mode squeezed vacuum is given by $|z\rangle_{12} = \hat{S}_{12}(z)|0, 0\rangle$, where the two-mode squeezing operator is $\hat{S}_{12}(z) = \exp(z^*\hat{a}\hat{b} - z\hat{a}^\dagger\hat{b}^\dagger)$, where $\hat{a}$ and $\hat{b}$ act on modes 1 and 2, respectively, and again z is complex. The coherent state is given by $|\alpha\rangle = \hat{D}(\alpha)|0\rangle$ where the displacement operator is $\hat{D}(\alpha) = \exp(\alpha\hat{a}^\dagger - \alpha^*\hat{a})$, $\alpha = |\alpha|e^{i\theta_c}$ where $|\alpha|$ is the amplitude, and $\theta_c \in [0, 2\pi]$ is the coherent state phase. The amplitude of the coherent state can be large in experiments, so instead it is limited by the numerical methods we use: we set the limit to $\alpha = 4$. The final input state is the Fock state of which the simplest is the vacuum $|0\rangle$. Single photons, $|1\rangle$, can be emitted from a quantum dot (Müller et al. 2015; Claudon et al. 2010) or heralded (Morin et al. 2012). We also consider the two-photon state, $|2\rangle$, which has been made in Ourjoumtsev et al. (2006) and Huang (2015). Higher number Fock states can be made, e.g. by heralding, but are challenging to produce to a high fidelity and are not included here.

Operators The beam splitter is described by the unitary operator $\hat{U}_T = e^{-i\theta_b(e^{i\phi_b}\hat{a}^\dagger\hat{b} + e^{-i\phi_b}\hat{a}\hat{b}^\dagger)}$, where $\hat{a}$ and $\hat{b}$ are annihilation operators for the two modes, and we choose the arbitrary phase to be $\phi_b = -\pi/2$. Here, $T = \cos^2\theta_b$ is the transmissivity of the beam splitter and therefore for a 50:50 beam splitter $\theta_b = \pi/4$ giving $\hat{U}_{T=50}$. Next, the displacement operator, $\hat{D}(\beta)$ (defined above), is implemented by mixing the state with a large local oscillator at a highly transmissive beam splitter (Paris 1996) (β has the same restrictions as α). The phase operator is given by $e^{i\hat{n}\theta}$ where $\hat{n} = \hat{a}^\dagger\hat{a}$ and $\theta \in [0, 2\pi]$.

Measurements After we have applied a number of operators, we perform a heralding measurement on one mode of the final state. For example, if we wish to herald on the one photon state, we can perform a number resolving detection (Humphreys et al. 2015; Gerrits et al. 2010), and only keep runs that measure one photon. The measurement is given by a projection (Nielsen and Chuang 2010): to follow the single photon example, we project with $|1\rangle\langle 1| \otimes \hat{\mathbb{I}}$. We are then left with a separable state $|1\rangle \otimes |\psi_f\rangle$, but we can ignore the measurement mode, and after normalisation, we are left with the final one mode state: $|\psi_f\rangle$. This whole process can be more easily modeled by acting on the two-mode, pre-measurement state with $\langle 1| \otimes \hat{\mathbb{I}}$. In the main text, we drop the identity and just write $\langle 1|$, and this measurement is always performed on the first mode of Fig. 1.

Appendix 2. Running AdaQuantum

To obtain the results in this paper, we ran AdaQuantum with the following settings (see Nichols et al. (2018) for a detailed description of how AdaQuantum works): The population sizes for stages 1, 2, and 3 are 8×10^6, 5×10^6, and 10^4, respectively. Stages 2 and 3 use MATLAB's build-in genetic algorithm (https://uk.mathworks.com/help/gads/index.html), and both these stages have the same hyperparameters (except for the population sizes). We use the `scattered` crossover function, with crossover fraction 0.3. We use `tournament` selection, with a tournament size of 8. MATLAB did not have a mutation function with enough flexibility for our purposes, so we introduced a mutation functions that we call `power mutation`, which is based on Deep and Thakur (2007). In short, `power mutation` mutates every gene in the genome by a random distance, whose maximum magnitude

is determined by the value of a hyperparameter named `power`, for which `power`= 1 will mutate each gene to a completely random new value, whereas `power`= ∞ does not mutate at all. Here, we used `power`= 10. The number of elite children was 10.

References

Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M et al (2016) Tensorflow: a system for large-scale machine learning. OSDI 16:265–283

AdaQuantum is available and free to use on GitHub. https://github.com/paulk444/AdaQuantum, (2019)

Arrazola JM, Bromley TR, Izaac J, Myers CR, Brádler K, Killoran N (2019) Machine learning method for state preparation and gate synthesis on photonic quantum computers. Quantum Science and Technology 4:024004

Barnett SM, Radmore PM (2002) Methods in Theoretical Quantum Optics, vol 15. Oxford University Press, Oxford

Bartley TJ, Donati G, Spring JB, Jin X-M, Barbieri M, Datta A, Smith BJ, Walmsley IA (2012) Multiphoton state engineering by heralded interference between single photons and coherent states. Phys Rev A 86(4):043820

Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S (2017) Quantum machine learning. Nature 549(7671):195

Claudon J, Bleuse J, Malik NS, Bazin M, Jaffrennou P, Gregersen N, Sauvan C, Lalanne P, Gérard J (2010) A highly efficient single-photon source based on a quantum dot in a photonic nanowire. Nature Photon 4(3):174–177

Deep K, Thakur M (2007) A new mutation operator for real coded genetic algorithms. Appl Math Comput 193(1):211–230

Dunjko V, Briegel HJ (2017) Machine learning and artificial intelligence in the quantum domain. arXiv:1709.02779

Etesse J, Bouillard M, Kanseri B, Tualle-Brouri R (2015) Experimental generation of squeezed cat states with an operation allowing iterative growth. Phys Rev Lett 114(19):193602

Gerrits T, Glancy S, Clement TS, Calkins B, Lita AE, Miller AJ, Migdall AL, Nam SW, Mirin RP, Knill E (2010) Generation of optical coherent-state superpositions by number-resolved photon subtraction from the squeezed vacuum. Phys Rev A 82(3):031802

Ghose S, Sanders BC (2007) Non-gaussian ancilla states for continuous variable quantum computation via gaussian maps. J Mod Opt 54(6):855–869

Gottesman D, Kitaev A, Preskill J (2001) Encoding a qubit in an oscillator. Phys Rev A 64(1):012310

Gu M, Weedbrook C, Menicucci NC, Ralph TC, van Loock P (2009) Quantum computing with continuous-variable clusters. Phys Rev A 79(6):062318

Hall MJW, Wiseman HM (2012) Heisenberg-style bounds for arbitrary estimates of shift parameters including prior information. New J Phys 14(3):033040

Hall MJW, Berry DW, Zwierz M, Wiseman HM (2012) Universality of the Heisenberg limit for estimates of random phase shifts. Phys Rev A 85(4):041802

Huang K (2015) Optical Hybrid Architectures for Quantum Information Processing. PhD thesis l'École Normale supérieure de Paris and East China Normal University.

Huang K, Le Jeannic H, Ruaudel J, Verma VB, Shaw MD, Marsili F, Nam SW, Wu E, Zeng H, Jeong Y-C et al (2015) Optical synthesis of large-amplitude squeezed coherent-state superpositions with minimal resources. Phys Rev Lett 115(2):023602

Humphreys PC, Metcalf BJ, Gerrits T, Hiemstra T, Lita AE, Nunn J, Nam SW, Datta A, Kolthammer WS, Walmsley IA (2015) Tomography of photon-number resolving continuous-output detectors. arXiv:1502.07649

Jiang L-y, Guo Q, Xu X-x, Cai M, Yuan W, Duan Z-l (2016) Dynamics and nonclassical properties of an opto-mechanical system prepared in four-headed cat state and number state. Opt Commun 369:179–188

Jin Y (2011) Surrogate-assisted evolutionary computation: Recent advances and future challenges. Swarm Evol Comput 1(2):61–70

Kingma DP, Ba J (2014) Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG], 12

Knott P, Proctor T, Hayes A, Cooling J, Dunningham J (2016) Practical quantum metrology with large precision gains in the low-photon-number regime. Phys. Rev. A 93(3):033859

Knott PA (2016) A search algorithm for quantum state engineering and metrology. New J Phys 18(7):073033

Kok P, Lovett BW (2010) Introduction to Optical Quantum Information Processing. Cambridge University Press, Cambridge

Krenn M, Malik M, Fickler R, Lapkiewicz R, Zeilinger A (2016) Automated search for new quantum experiments. Phys Rev Lett 116(9):090405

Lee S-Y, Lee C-W, Lee J, Nha H (2015) Quantum phase estimation using a class of entangled states: NOON-type states. arXiv:1505.06000

Lee S-Y, Lee C-W, Nha H, Kaszlikowski D (2015) Quantum phase estimation using a multi-headed cat state. JOSA B 32(6):1186–1192

Mehmet M, Ast S, Eberle T, Steinlechner S, Vahlbruch H, Schnabel R (2011) Squeezed light at 1550 nm with a quantum noise reduction of 12.3 db. Opt Express 19(25):25763–25772

Melnikov AA, Nautrup HP, Krenn M, Dunjko V, Tiersch M, Zeilinger A, Briegel HJ (2018) Active learning machine learns to create new quantum experiments. In: Proceedings of the National Academy of Sciences, p 201714936

Morin O, D'Auria V, Fabre C, Laurat J (2012) High-fidelity single-photon source based on a type II optical parametric oscillator. Opt Lett 37(17):3738–3740

Müller K, Rundquist A, Fischer KA, Sarmiento T, Lagoudakis KG, Kelaita YA, Sanchez Muñoz C, del Valle E, Laussy FP, Vučković J (2015) Coherent generation of nonclassical light on chip via detuned photon blockade. Phys Rev Lett 114(23):233601

Nichols R, Mineh L, Rubio J, Matthews JCF, Knott PA (2018) Designing quantum experiments with a genetic algorithm. arXiv:1812.01032

Nielsen MA, Chuang IL (2010) Quantum Computation and Quantum Information. Cambridge University Press, Cambridge

Our DNN and quantum-state generator on GitHub. https://github.com/lewis-od/Quantum-Optics, (2018)

Ourjoumtsev A, Tualle-brouri R, Grangier P (2006) Quantum homodyne tomography of a two-photon Fock state. Phys Rev Lett 96(21):213601

Ourjoumtsev A, Jeong H, Tualle-Brouri R, Grangier P (2007) Generation of optical 'schrödinger cats' from photon number states. Nature 448(7155):784

Paris MGA (1996) Displacement operator by beam splitter. Phys Lett A 217(2):78–80

Rivas A, Luis A (2012) Sub-heisenberg estimation of non-random phase shifts. New J Phys 14(9):093052

Rubio J, Knott P, Dunningham J (2018) Non-asymptotic analysis of quantum metrology protocols beyond the Cramér–rao bound. Journal of Physics Communications 2(1):015027

Sabapathy KK, Qi H, Izaac J, Weedbrook C (2018) Near-deterministic production of universal quantum photonic gates enhanced by machine learning. arXiv:1809.04680

Sabapathy KK, Weedbrook C (2018) On states as resource units for universal quantum computation with photonic architectures. Phys Rev A 97(6):062315

Schrödinger E (1935) Die gegenwärtige situation in der quantenmechanik. Naturwissenschaften 23(49):823–828

Schuld M, Sinayskiy I, Petruccione F (2015) An introduction to quantum machine learning. Contemp Phys 56(2):172–185

Takagi R, Zhuang Q (2018) Convex resource theory of non-gaussianity. Phys Rev A 97:062337

T M Inc., Global optimization toolbox: user's guide (r2018a). https://uk.mathworks.com/help/gads/index.html. Last accessed 2018-07-17

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.