Modelling Non-Markovian Quantum Processes with Recurrent Neural Networks

Leonardo Banchi,[1] Edward Grant,[2] Andrea Rocchetto,[2,3] and Simone Severini[2]

[1]*QOLS, Blackett Laboratory, Imperial College London, London SW7 2AZ, United Kingdom*
[2]*Department of Computer Science, University College London, London WC1E 6EA, United Kingdom*
[3]*Department of Computer Science, University of Oxford, Oxford OX1 3QD, United Kingdom*

Quantum systems interacting with an unknown environment are notoriously difficult to model, especially in presence of non-Markovian and non-perturbative effects. Here we introduce a neural network based approach, which has the mathematical simplicity of the Gorini-Kossakowski-Sudarshan-Lindblad master equation, but is able to model non-Markovian effects in different regimes. This is achieved by using recurrent neural networks for defining Lindblad operators that can keep track of memory effects. Building upon this framework, we also introduce a neural network architecture that is able to reproduce the entire quantum evolution, given an initial state. As an application we study how to train these models for quantum process tomography, showing that recurrent neural networks are accurate over different times and regimes.

I. INTRODUCTION

Traditionally, in the physical sciences, the solution to mathematical problems for which no analytic solution is available involves modelling methods leveraging a combination of approximation techniques (such as perturbation theory or semiclassical approaches) and the use of symmetries to reduce the complexity of the problem. Recently, with the surge in popularity of machine learning techniques [1, 2] data–driven approaches, which instead rely on computational techniques that exploit statistical correlations, have started to find applications in different fields, ranging from chaos theory [3] to high–energy physics [4], eventually showing many applications and new perspectives even in the quantum domain [5, 6]. For instance, in quantum many–body physics, where numerical techniques are particularly challenged by the exponential scaling of the wavefunction, learning methods have been used in a number of different domains. In particular, artificial neural networks (a class of learning methods inspired by the functioning of biological neural networks) have been utilized for ground state estimation [7], quantum state tomography [8], classification of phases of matter [9], entanglement estimation [10], and to identify phase transitions [11]. Although the current theoretical understanding of the effectiveness of these models is limited, a growing body of literature has established connections between neural networks and standard frameworks commonly used to analyze quantum systems such as renormalization groups [12], tensor networks [13–15], and the complexity theoretic tools developed in quantum computing [16]. Moreover, classical optimization techniques borrowed from supervised machine learning have been employed to optimize the dynamics of many-body systems [17–21] and parametric quantum circuits [22–28].

Open quantum systems [29, 30] present further challenges. Here, any modelling effort must take into account that the system interacts with the surrounding environment. These effects, which are usually unknown, are treated in a phenomenological way and can significantly increase the complexity of the model. This approach contrasts with what is typically done in the neural network community, where overparametrized models *learn* from data, rather than describing an environment with complex functional dependencies that can be hard to derive from phenomenological observations. Data driven approaches are responsible for the success of deep networks into fields like natural language processing and image recognition [2], where huge amounts of data are readily available.

Here we investigate the ability of neural networks to model the non-Markovian evolution of open quantum systems. We focus on Recurrent Neural Networks (RNNs), a type of artificial neural network specifically designed to model dynamical systems with possibly long range temporal correlations. In the quantum setting, RNNs have been previously employed for quantum control [31, 32]. We consider two main applications. In the first one we define a master equation which has the mathematical simplicity of the Gorini-Kossakowski-Sudarshan-Lindblad master equation [33, 34], but which is able to model non-Markovian effects via the memory cells included in RNNs. The resulting RNN architecture shares some similarity with collisional models [35–40] or ϵ-machines [41], where explicit memory effects are introduced by using ancillary quantum systems. Nonetheless, our approach has the advantage that it can be easily modified by choosing different RNN architectures that better reproduce the expected properties in the temporal sequence. Moreover, it provides a convenient mathematical way to define the so called memory kernel [29], such that the resulting master equation has a completely positive solution. The reconstruction of the memory kernel from quantum state sequences, namely from quantum process tomography, has been the subject of different studies in recent years [42–47]. In most of the approaches considered in the literature to reconstruct the memory kernel, one normally makes some assumptions on the microscopic model of the environment, and then fits the free parameters given the experimental data. However, the details of the microscopic model are difficult to know, and many uncontrollable assumptions have to be made. Our main result is to show that RNNs can learn memory effects in data sequences, without making any assumption on the underlying physical model. Indeed, even the Hamiltonian of the system can be learnt during this process.

Developing on the same formalism, we define and train a RNN architecture that reproduces the entire physical evolution given an initial quantum state, without introducing master equations. As a relevant application for this technique, we

consider quantum process tomography and show that RNNs are able to model the physical evolution of quantum systems over different regimes.

The paper is structured as follows. In Sec. II we present the relevant technical background. Specifically, in Sec. II A we introduce non-Markovian quantum processes and common master equation techniques to describe them, while in Sec. II B we introduce the RNN architecture employed in this paper. The main ideas are presented in Sec. III, where we introduce RNN-based master equations (Sec. III A), RNN-based quantum processes (Sec. III B), and applications for process tomography (Sec. III C). Numerical experiments are presented in Sec. IV and conclusions are drawn in Sec. V.

II. BACKGROUND

A. Non-Markovian processes

Quantum systems, even the purest ones, are inevitably in contact with an environment. Because of this normally unknown interaction, quantum evolution deviates from the predictions of the Schrödinger equation, and different extensions have been proposed [29, 48] to model such *open* quantum systems. When the interactions inside the environment happen on timescales much shorter than the internal timescales of the system, then a Markovian approximation is usually appropriate, and the evolution can be modeled via the Gorini-Kossakowski-Sudarshan-Lindblad (GKSL) master equation [33, 34]

$$\begin{aligned}\frac{\partial\rho(t)}{\partial t} =&\mathcal{L}[\rho(t)]\,, \\ \mathcal{L}[\rho] =&-i[H,\rho] + \sum_{\mu}\left[L_\mu\rho L_\mu^\dagger - \frac{1}{2}\left\{L_\mu^\dagger L_\mu,\rho\right\}\right]\,,\end{aligned} \tag{1}$$

where H is the Hamiltonian, which models the noise-free case, and L_μ are called Lindblad operators. The superoperator $\mathcal{L}$ is called Liouvillian. Mathematical properties of the above equation are well understood [30]. For any choice of H and L_μ the solution of the master equation $\mathcal{E}_t = e^{-\mathcal{L}t}$ defines a completely positive trace preserving linear map, and thus is a mathematically well-posed mapping between states to states. With properly chosen Lindblad operators, the above master equation models the most general Markovian interaction with an environment [33, 34]. However, the Markovian approximation is not accurate in many situations, for instance when the interactions inside the environment have comparable strengths to the interactions inside the system [49, 50]. In that case the master equation has to be modified to take into account non-Markovian effects. One of the first and most accurate descriptions of non-Markovian evolution is the Nakajima-Zwanzig (NZ) master equation [29]

$$\frac{\partial\rho(t)}{\partial t} = -i[H,\rho(t)] + \int_0^t ds\,\mathcal{K}_{t-s}^{\mathrm{NZ}}[\rho(s)] + \mathcal{I}(t)\,, \tag{2}$$

where $\mathcal{K}_t^{NZ}$ is a super-operator, the so-called *memory kernel*, which describes the interaction with the environment, while $\mathcal{I}(t)$ is due to the initial correlations between system and environment. If the system and environment are initially uncorrelated, then $\mathcal{I}(t) = 0$ for all t. It is clear that the above equation describes non-Markovian processes, because the state at time $t + dt$ depends not only on $\rho(t)$ but also on the states $\rho(s)$ for $s < t$. The Nakajima-Zwanzig equation is at the basis of powerful Green function methods to study the spectral properties of the system [50–53], since the convolution disappears in the frequency domain. On the other hand, in the time domain the above equation is not easy to solve numerically. To avoid this problem, a different but equally accurate master equation has been proposed, the so-called time-convolutionless (TCL) master equation [29], which reads

$$\frac{\partial\rho(t)}{\partial t} = -i[H,\rho(t)] + \int_0^t ds\,\mathcal{K}_{t-s}^{\mathrm{TCL}}[\rho(t)] + \mathcal{I}(t)\,. \tag{3}$$

The main formal difference between Eq. (3) and Eq. (2) is that the whole history of states is fed into the NZ master equation, while in the TCL case the master equation explicitly depends on $\rho(t)$ only, and all non-Markovian effects are included into the memory kernel. As such the non-Markovian nature of the above equation is less obvious, but it is known that both NZ and TCL master equations can describe the same process. Indeed there are formal mappings between $\mathcal{K}^{\mathrm{NZ}}$ and $\mathcal{K}^{\mathrm{TCL}}$ such that both both Eq. (3) and Eq. (2) produce the same physical evolution [29, 54].

Although TCL master equations are relatively easy to solve numerically, the main problem is that the interaction with the environment is normally unknown. In other terms, while H is typically well characterized experimentally, the memory kernel depends on environmental quantities such as the temperature, but also on the spectral properties of the environment which are normally unknown. Non-linear spectroscopy can be used to find a model of the spectral density [51], but this still does not completely characterizes the memory kernel, without introducing further assumptions. The most commonly employed approximation is the assumption of a weak coupling between system and environment, such that the memory kernel can be formally obtained using perturbation theory. However, there are cases where these approximations are not justified, e.g. in quantum biology [49], where the strength of the interaction with the environment is comparable with the internal interactions inside the system. Motivated by these we study a new ansatz to model non-Markovian quantum processes based on the use of RNNs.

B. Recurrent Neural Networks

RNNs are a class of neural networks designed to model data sequences like time series. Unlike feedforward neural networks, that provide just a functional approximations of the input-output relationships in the data, RNNs can model temporal dependencies arising in the dataset. To understand their functioning, it is helpful to compare them with more standard feedforward networks. In feedfordward neural networks the input data s^0 propagates throughout many intermediate (hidden) layers before reaching the final output layer.

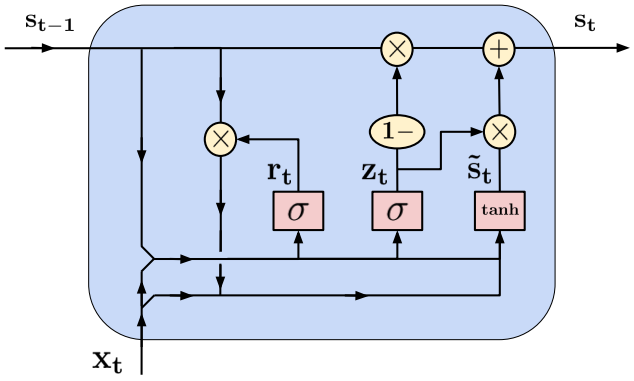

FIG. 1. **Network diagram of the GRU cell** . The output s_t and input s_{t-1} represent the state at times t and $t-1$, respectively, while x_t is an auxiliary input that depends on the previous times, before $t-1$. Rectangles represent neural network layers. Circles represent entry-wise operations. Bifurcations represent copy operations and joined lines represent concatenation. Details are presented in Appendix A.

Here "propagate" means that, step by step, the state $s^{\ell+1}$ of the $(\ell+1)$-th layer is updated, given the state of ℓ-th layer, as $s^{\ell+1} = f(W^\ell s^\ell + w^\ell)$ where W^ℓ is a weight matrix, w^ℓ a weight vector and f is some non-linear function. The state at the final layer of the network (output layer) depends on all the weight matrices and vectors. Training is performed by updating those weights such that the neural network learns some desired input-output relationship hidden in the data.

In the case of temporal data, each input has also an explicit time dependence s_t^0. Although, in principle, one could still use a giant feedforward network with these data, this is rarely the optimal choice, because the number of free parameters quickly increases with the number of time steps. RNNs solve this issue with a more advanced architecture which is tailored for temporal data. In RNNs, the update rule for hidden layers at time t does not only depend on the state s_t^ℓ, but also on the previous state of the unit. In other terms, the update rule is $s_t^{\ell+1} = f(W^\ell s_t^\ell + w^\ell, s_{t-1}^{\ell+1})$, where s_t^0 is the input temporal sequence. The free parameters W^ℓ and w^ℓ do not depend on t, and memory of the past is taken into account by the function f, which compresses and saves relevant informations of previous sequences into memory cells. This architecture allows RNNs to learn temporal sequences using a relatively small number of parameters, even when the temporal data has long range memory effects. The mapping between s_t^ℓ and $s_t^{\ell+1}$ defines a RNN cell.

In this work we use a variant of RNN cells called Gated Recurrent Unit (GRU) RNN, shown in Fig. 1, and discussed in Appendix A. GRUs use a gating mechanism that allows them to better model long term dependencies than more simple RNNs [55]. GRUs are based on a type of RNN cell called Long Short-Term Memory (LSTM) but can be more efficient for comparable performance [56, 57]. GRU and LSTM are commonly used and achieve state of the art performance for sequence modelling across multiple domains, including machine translation, image captioning and forecasting [58].

The GRU state s_t is a linear interpolation of the previous state s_{t-1} and a candidate state $\tilde{s}_t$, which depends on the auxiliary input x_t. The input x_t^j for a depth j cell at time t is the state from the cell in the previous layer s_t^{j-1}. GRU cells can be stacked to form a deep GRU network. More details can be found in Appendix A.

III. MAIN IDEA

In this section we present the three main contributions of this paper which all leverage on the modelling capabilities of RNNs to describe dynamics of open quantum systems. First, we describe a non-Markovian master equation. Second, we use RNNs to predict the time evolution of quantum state under a non-Markovian quantum process. Third, we show how these two techniques can be utilized to perform quantum process tomography.

A. RNN Quantum Master Equation

We postulate a quantum evolution similar to Eq. (1), namely

$$\frac{\partial \rho(t)}{\partial t} = \mathcal{L}_{\leq t}[\rho(t)] \,, \tag{4}$$

where the notation $\leq t$ refers to a superoperator that not only depends on t, but also on the entire history before time t, as in the TCL master equation (3). A convenient choice is then that of the GKSL form

$$\begin{aligned}\mathcal{L}_{\leq t}[\rho] = &-i[H + H^{LS}_{\leq t}, \rho] + \\ &+ \sum_\mu \left[L^\mu_{\leq t} \rho L^{\mu\dagger}_{\leq t} - \frac{1}{2} \left\{ L^{\mu\dagger}_{\leq t} L^\mu_{\leq t}, \rho \right\} \right] \,,\end{aligned} \tag{5}$$

where H^{LS} is a "Lamb-shift" term, namely a correction to the Hamiltonian induced by the environment, while L^μ are Lindblad operators. The reason for this choice is that for small enough Δ_t the time evolution is simply given by $\rho(t+\Delta_t) \approx e^{\Delta_t \mathcal{L}_{\leq t}}[\rho(t)]$, and since $\mathcal{L}_{\leq t}$ is in the GKSL form, $e^{\Delta_t \mathcal{L}_{\leq t}}$ is a completely positive trace preserving quantum channel, mapping states to states. If H^{LS} and L^μ are simply time-dependent functions, namely they depend only on t and not on previous times, then the above master equation is always Markovian [54]. The main idea of this work is to use a RNN to define each Lindblad operator $L^\mu_{\leq t}$ and the correction Hamiltonian $H^{LS}_{\leq t_j}$, see fig. 2(a). In order to ensure the Hermiticity of the $H^{LS}_{\leq t}$ operator, we construct it as $H^{LS}_{\leq t} = A(t_j) + A(t_j)^\dagger$, where $A(t_j)$ is the output of the network and $A(t_j)^\dagger$ its conjugate transpose. Since in RNNs the predicted output at time t depends on the entire history at previous times, this parametrization is expected to accurately reproduce genuinely non-Markovian effects, even with possible long-range dynamical correlations. The master equation then resembles a TCL non-Markovian master equation (3), but where the complicated memory superoperator is expressed via a simpler GKSL form with RNNs. We call our Eq. (5) the Quantum Recurrent

Neural (QRN) Master Equation. Similarly, we call the operators $L^{\mu}_{\leq t}$ recurrent Lindblad operators. A schematics of the resulting neural network is shown in Fig. 2(a).

B. RNN Quantum Processes

For any initial time t_0 the mapping between the initial state $\rho(t_0)$ and the state at time t, namely

$$\rho(t) = \mathcal{E}(t, t_0)[\rho(t_0)] \,, \tag{6}$$

defines a completely positive map, assuming no initial correlation with the environment. For any intermediate times $t_0 < \tau < t$ the mapping $\mathcal{E}(\tau, t_0)$ is always completely positive. When also $\mathcal{E}'(t, \tau) := \mathcal{E}(t, t_0)\mathcal{E}(\tau, t_0)^{-1}$ is completely positive, then the mapping is called *divisible* [59]. Divisibility is another way of characterising Markovianity, as quantum processes obtained from the GKSL master equation are always divisible with $\mathcal{E}(t', t) = \mathcal{T} \exp\left(\int_t^{t'} \mathcal{L}_s \, ds\right)$.

In the previous section we have introduced the QRN master equation and shown that it can model non-Markovian effects, even when the evolution between intermediate steps is completely positive. This is possible because the RNN keeps a compressed record of the previous evolution. Using the formalism of the previous section we can indeed write

$$\mathcal{E}(t, t_0) = \overleftarrow{\prod_j} \mathcal{E}^{QRN}(t_j + \Delta_t, t_j) + O(\Delta_t^{-2}), \tag{7}$$

where $\overleftarrow{\prod}_j X_j$ is the ordered product $\cdots X_3 X_2 X_1$. Moreover,

$$\mathcal{E}^{QRN}(t_j + \Delta_t, t_j) = e^{\Delta_t \mathcal{L}_{\leq t_j}} \tag{8}$$

is completely positive, being the operator exponential of a Liouvillian, but depends on the previous evolution, via the recurrent Lindblad operators. As such, the total map (7) is not divisible.

Based on this analogy, we can now drop the master equation and define a non-Markovian process as

$$\mathcal{E}(t, t_0) = \overleftarrow{\prod_j} \mathcal{E}^{RNN}_{\leq t_j}(t_{j+1}, t_j) \,, \tag{9}$$

where $\mathcal{E}^{RNN}_{\leq t_j}(t_{j+1}, t_j)$ is completely positive, but depends on the entire history before t_j. Each $\mathcal{E}^{RNN}_{\leq t_j}(t_{j+1}, t_j)$, being completely positive, outputs a valid quantum state at intermediate times $\rho(t_j)$ and, being a RNN, then updates its internal memory. Complete positivity can be ensured by using the Kraus decomposition or the environmental representation.

For better comparison with the QRN master equation, in this work we use the simpler strategy shown in Fig. 2(b), where we use the output $A(t_j)$ of the network to define a density operator via $\rho(t_{j+1}) = A(t_j)A(t_j)^\dagger / \operatorname{Tr}[A(t_j)A(t_j)^\dagger]$. This ensures that the states $\rho(t_j)$ are valid density operators, throughout the entire evolution.

C. Application: Quantum Process Tomography

In this work we apply our QRN master equations and RNN quantum processes for quantum process tomography. We consider the following simple setup. We assume that the quantum system can be initialized in different initial states $\rho_\alpha(0)$ where $\alpha = 1, \ldots, N_{\rm I}$, for some number $N_{\rm I}$. For each initialization, we assume that it is possible to perform full-quantum tomography at some time steps $0 \leq t_j \leq T$ for $j = 1, \ldots, N_{\rm T}$ and some $N_{\rm T}$. After this procedure we are able to reconstruct the time evolutions $\rho_\alpha(t_j)$ for different times and initializations. Here we consider uniformly separated times where $t_j = j\Delta_T$ and $\Delta_T = T/N_{\rm T}$, though it is straightforward to generalize this procedure to non-uniform sequences. Each state tomography requires $O(d^2)$ measurements, where d is the dimension of the system's Hilbert space, so the total cost of reconstructing these sequences is $O(d^2 N_{\rm T} N_{\rm I})$. Once these sequences are obtained, we use them to train the neural networks.

We consider two cases. In the first one we train a RNN to learn the quantum state evolution, as shown in Fig. 2(b). Here we assume no knowledge about the system's Hamiltonian or interaction with the environment. Training is then performed by minimising a cost function

$$J_p = \frac{1}{N_{\rm I} N_{\rm T}} \sum_{\alpha=1}^{N_{\rm I}} \sum_{j=1}^{N_{\rm T}} \|\rho_\alpha(t_j) - \tilde{\rho}_\alpha(t_j)\| \,, \tag{10}$$

where $\rho_\alpha(t_j)$ are the training data, $\tilde{\rho}_\alpha(t_j)$ are the states outputted by the RNN, and $\| \cdot \|$ is any operator norm.

In the second application we aim at reconstructing $H^{LS}_{\leq t}$ and the recurrent Liouvillian operators $L^{\mu}_{\leq t}$ entering in the QRN master equation, where, on the other hand, we assume that the Hamiltonian H is known. The latter assumption can always be relaxed, as Hamiltonian evolution can be fully included into the correction Hamiltonian H^{LS}, which is learnt from the data. In the QRN master equation, the operators $H^{LS}_{\leq t}$ and $L^{\mu}_{\leq t}$ are obtained from the output of the RNN, as shown in Fig. 2(b). To train the RNNs to predict H^{LS} and L^{μ} given a starting state $\rho_\alpha(t = 0)$ we propose the use of the differential equation (4) to define a cost function.

To explain the idea, let us first consider the opposite scenario, where the differential equation (4), with all of its operators, is already known, while the states $\rho(t_j)$ are not. In this common case, the states $\rho(t_j)$ at different times are evaluated from the numerical integration of the master equation. The latter can be obtained with an n-order Runge-Kutta integrator [60] which, in general, can be formally written as

$$\rho_\alpha(t_{j+1}) = \rho_\alpha(t_j) + \Delta_T \mathcal{L}^{{\rm RK},n}_{\leq t_j}[\rho_\alpha(t_j)] \,, \tag{11}$$

where $\mathcal{L}^{{\rm RK},n}_{\leq t}$ is n-th order Runge-Kutta integration step, which can be explicitly obtained for any n (see e.g. Ref. [60]). For instance, to the first order $\mathcal{L}^{{\rm RK},1}_{\leq t}$ is simply $\mathcal{L}_{\leq t}$. To summarize, when $H^{LS}_{\leq t}$ and $L^{\mu}_{\leq t}$ are known, we can use a Runge-Kutta integrator to obtain the time evolution $\rho(t_j)$.

We now consider the opposite problem, namely where many time sequences $\rho_\alpha(t_j)$ are already known, while the operators $H^{LS}_{\leq t}$ and $L^{\mu}_{\leq t}$ in the QRN master equation are not. This

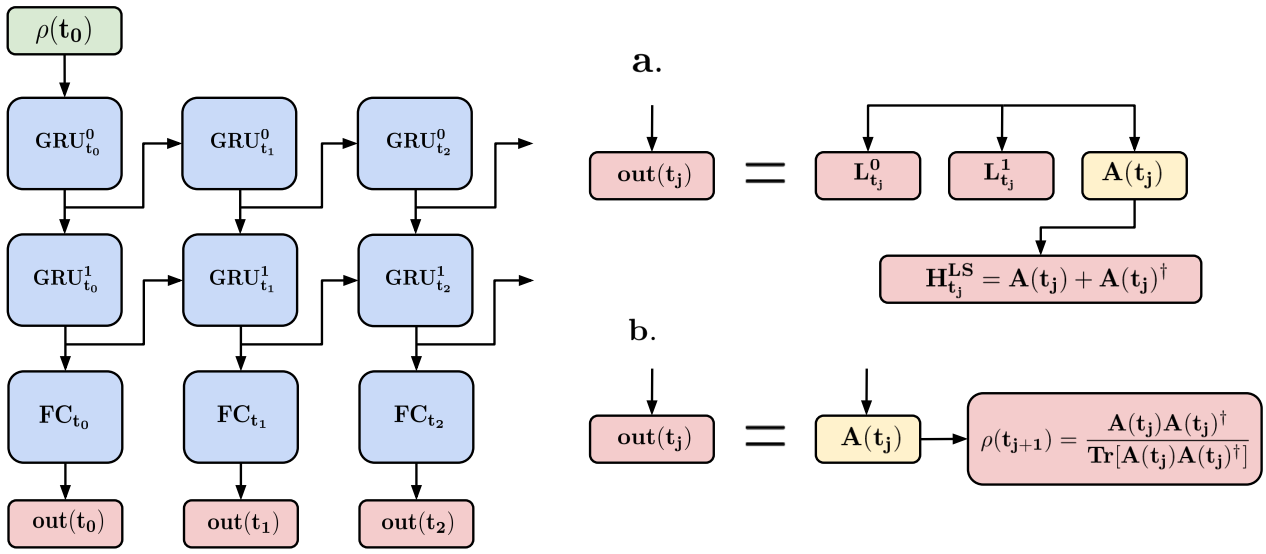

FIG. 2. **RNN architectures for open quantum systems**: schematic of a one-to-many deep RNN used to approximate the master equation and to model the non-Markovian dynamics. Both networks take as input $\rho(t_0)$ (green cell) and comprise of two GRU layers and a fully connected (FC) layer (blue cells). The yellow cells show the initial network output $A(t_j)$ before the post-processing that makes $H^{LS}_{t_j}$ Hermitian and $\rho(t_{j+1})$ a valid density operator. Panel **a.** shows the output for the QML master equation, i.e. the predicted value of H^{LS}_t and L^μ_t at intervals Δt (red cells). Panel **b.** shows the output for a RNN modelling the time evolution of a quantum state undergoing a non-Markovian quantum process, i.e. the predicted value of $\rho(t)$ at intervals Δt (red cells).

is like assuming that the solutions of a differential equation are known and from them we want to reconstruct the differential equation itself. Based on the analogy with numerical integration via Runge-Kutta, we propose to use the following cost function:

$$J = \frac{1}{N_{\rm I}N_{\rm T}} \sum_{\alpha=1}^{N_{\rm I}} \sum_{j=1}^{N_{\rm T}} \left\| \rho_\alpha(t_{j+1}) - \rho_\alpha(t_j) - \Delta_T \mathcal{L}^{{\rm RK},n}_{\leq t_j}[\rho_\alpha(t_j)] \right\|^2_F , \tag{12}$$

where $\|\cdot\|_F$ is the Frobenius norm. The intuitive idea behind the above cost function is that of making the measured data sequences as close as possible to those coming from the numerical solution of a master equation. Minimizing the cost function is then equivalent to finding the best QRN master equation compatible with the measured sequences $\rho_\alpha(t_j)$. It is expected that a higher order integrator (large n) performs better, especially for larger Δ_T, but requires heavier numerical computations.

IV. NUMERICAL EXPERIMENTS

A. Learning quantum state sequences

We mimic experimental data by numerically generating sequences of states, which are then used to train the neural network. To generate the training data, we consider a simple yet important model of spontaneous decay of a two-level system [48, 61], described by the master equation

$$\frac{\partial}{\partial t}\rho(t) = -i[\omega\sigma_z, \rho(t)] + \\ + \gamma(t)\left(\sigma^-\rho(t)\sigma^+ - \frac{1}{2}\{\sigma^+\sigma^-, \rho(t)\}\right), \tag{13}$$

where $\{x, y\} = xy + yx$, σ^α for $\alpha = x, y, z$ are the Pauli matrices, $\sigma^\pm = (\sigma^x \pm i\sigma^y)/2$, ω is the Rabi frequency of oscillations around the z axis. The parameter

$$\gamma(t) = \frac{2\gamma_0\lambda\sinh(\eta t/2)}{\eta\cosh(\eta t/2) + \lambda\sinh(\eta t/2)}. \tag{14}$$

is the decay rate with $\eta = \sqrt{\lambda^2 - 2\gamma_0\lambda}$. When $\lambda > 2\gamma_0$ the function $\gamma(t)$ is always positive, so Eq. (13) takes the GKSL form (1) and, as such, defines a Markovian evolution. On the other hand, for $\lambda < 2\gamma_0$ the function $\gamma(t)$ can be negative and the dynamics displays non-Markovian effects [48, 62, 63].

We use the above model to obtain training data for the RNN architecture shown in Fig. 2(b). The training sequence $\rho_\alpha(t_j)$, for $\alpha = 1, \ldots, N_{\rm I}$ has been obtained by first choosing a random initial state $\rho_\alpha(0)$ and then solving the master equation Eq. (13) to get the states $\rho_\alpha(t_j)$ at subsequent times, up to a maximal time t_{max}. These data sequences were then used to train a GRU neural network, by minimising the cost function (10). After training, we test the accuracy of the neural network by generating a new sequence of states $\rho_\beta(t_j)$, and the RNN prediction $\tilde{\rho}_\beta(t_j)$, for $\beta = 1, \ldots, N_{\rm P}$. As before, $\rho_\beta(t_j)$ is obtained by selecting a random initial state $\rho_\beta(0)$ and solving Eq. (13), possibly for longer times than t_{max}. On the other hand, the predicted evolution $\tilde{\rho}_\beta(t_j)$ is obtained by feeding the

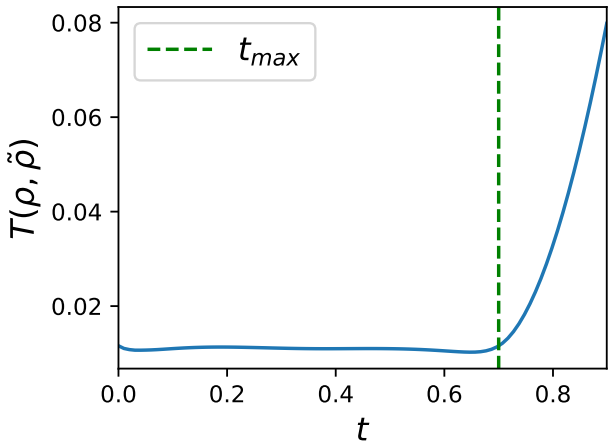

FIG. 3. **(EXP1) Learning the evolution of a noisy two level system:** the mean trace distance $T(\rho(t), \tilde{\rho}(t)) = \frac{1}{N_P} \sum_\beta D(\rho_\beta(t), \tilde{\rho}_\beta(t))$ between true states $\rho_\beta(t_j)$ and predicted states $\tilde{\rho}_\beta(t_j)$ as a function of time. The time t_{max} is the maximum time used during training. All the experiments run with the following parameters $\omega = 1$ for Eq. 13 and $\lambda = 2$, $\gamma_0 = 0.5$ for Eq. 14, and a discretisation interval of $\Delta t = 0.01$. The simulations run over 3000 training examples and predictions were tested over 1000 test examples. All the evolutions run for a time of $t_{max} = 0.7$.

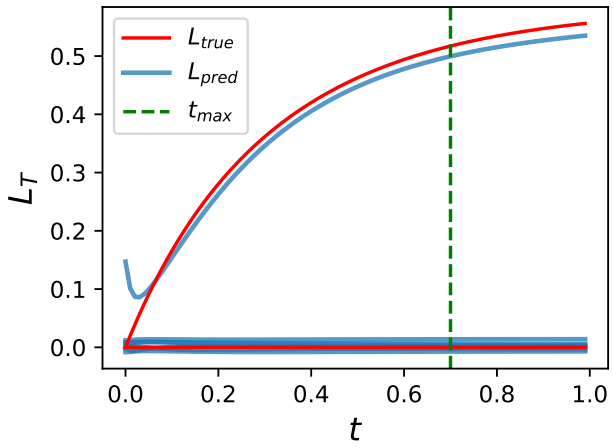

FIG. 4. **(EXP2) Learning the Lindblad operator describing the Markovian evolution of a two level system:** the learned and known entries of $\frac{1}{\alpha} \sum_\alpha L^{\mu \equiv 1}$ are compared for the Markovian evolution of a two-level system. We plot the time evolution of every entry of the matrix. Real and complex entries are plotted separately. All the experiments run with the following parameters $\omega = 1$ for Eq. (13) and $\lambda = 2$, $\gamma_0 = 0.5$ for Eq. (14), and a discretisation interval of $\Delta t = 0.01$. The simulations run over 1500 training examples and predictions were tested over 2500 test examples. All the evolutions run for a time of $t_{max} = 0.7$. The Frobenius norm squared was used as distance between the matrices.

initial state $\rho_\beta(0)$ to the RNN to get the entire temporal sequence. The two evolutions are compared with the trace distance $D(\rho, \tilde{\rho}) = \frac{1}{2} \mathrm{Tr} |\rho - \tilde{\rho}|$. In particular, we study the average trace distance $T(\rho(t), \tilde{\rho}(t)) = \frac{1}{N_P} \sum_\beta D(\rho_\beta(t), \tilde{\rho}_\beta(t))$ as a function of time.

In Fig. 3 we show a numerical solution of this numerical experiment (EXP1), where we can see that, once trained, the RNN is able to predict the evolution of a known starting state, up to the maximal training time t_{max}. On the other hand, and as expected, the accuracy of the RNN prediction rapidly deteriorates for $t > t_{max}$.

B. Learning the master equation: a simple case

In Fig. 4 we show the solution of a second numerical experiment (EXP2) obtained with the same training set of the first experiment (EXP1), discussed in Fig. 3. While EXP1 uses a RNN to model the entire quantum evolution, EXP2 uses the RNN to define a QRN master equation, following Sec. III C. Training is then performed by minimising the QRN cost function (12), where for simplicity we assume a first-order Runge-Kutta integrator ($n = 1$) and a single recurrent Lindblad operator ($\mu \equiv 1$). In EXP2 we have chosen a Markovian regime so that the entire evolution can be modeled via a GKSL master equation (1). In this regime, the RNN Lindblad operators can be approximated as a simple time-dependent function, so the minimisation of the cost function (12) is equivalent to learning standard Lindblad operators. In Fig. 4 we compare the predicted recurrent Lindblad operators L_μ as a function of time,

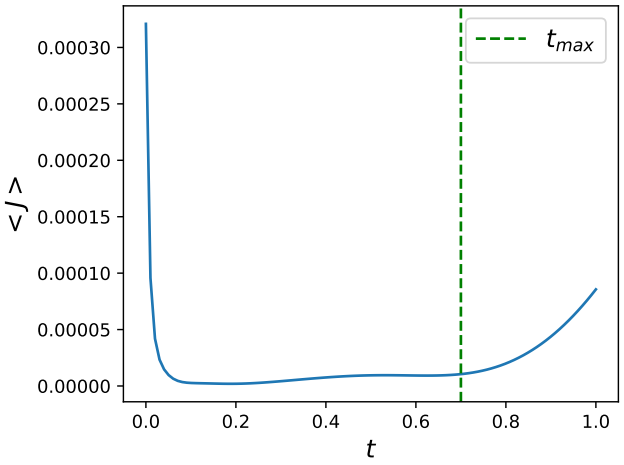

FIG. 5. **(EXP2) Average cost function value** (12) **for different times.** The parameters are the same of Fig. (4).

with respect to the real ones defined in Eq. (13). In the system under study all entries of the Lindblad operators are zero (hence the flat lines in the figure) apart from one value. As we can see, the prediction is remarkably accurate at all times, even beyond the training time $t > t_{max}$. The error in the prediction is shown in Fig. 5, by plotting the cost function (12) for different times. EXP2 shows that the RNN was able to learn the evolution of the Lindblad operator L_t for $t = 0.1$ to

$t_{max} = 0.7$. In addition, the RNN was able to predict L_t for $t > t_{max}$ which was the last time step used during training.

C. Learning the master equation: non-Markovian case

In this section we focus on the non-Markovian regime, where there are non-trivial memory effects that the RNN has to learn and reproduce. As in Sec. IV A, we consider state sequences generated numerically by solving a master equation. However, unlike our previous treatment, here we consider a more complicated non-Markovian model of the environment, which includes back-scattering effects. Back-scattering can refer, for instance, to a photon emitted to the environment that comes back at later times. As such, the information transferred to the environment is not completely lost. To model these non-Markovian effects we consider two qubits evolving with the following master equation

$$\frac{\partial}{\partial t}\rho_{12}(t) = -i[H_{12}, \rho_{12}(t)] + \\ + \sum_{i=1}^{2} \gamma_i(t)\left(\sigma_i^- \rho_{12}(t)\sigma_i^+ - \frac{1}{2}\{\sigma_i^+\sigma_i^-, \rho_{12}(t)\}\right), \quad (15)$$

where $\gamma_i(t)$ has the same functional form of Eq. (14) (but we denote the parameter γ_0 and λ with a superscript $\gamma_0^{(i)}$ and $\lambda^{(i)}$ that refers to qubit index) and the two-qubit Hamiltonian is

$$H_{12} = \omega\,\sigma_z \otimes I + c_1\,\sigma_x \otimes \sigma_x + c_2\,\sigma_y \otimes \sigma_y + c_3\,\sigma_z \otimes \sigma_z, \quad (16)$$

where $c_1 = 0.3242$, $c_2 = 0.6723$, and $c_3 = 0.1353$. We numerically solve Eq. (15) to get the data sequence $\rho_{12}^{\alpha}(t_j)$, where $\alpha = 1, \ldots, N_{\rm I}$ indexes the different solutions obtained with different initial states. From these two qubit solutions we define then the state sequences as $\rho_\alpha(t_j) = \mathrm{Tr}_2[\rho_{12}^{\alpha}(t)]$. In other terms, qubit 1 is the principal system, while qubit 2 is an ancillary system. Because of the coherent interaction H_{12} between qubit 1 and qubit 2, with this approach we can mimic the back-action of the environment onto the system.

We divide our numerical results into different experiments. In EXP3, shown in Fig. 6, we train a RNN to fully reproduce the entire quantum evolution, following the discussion of Sec. III B. We see that in spite of non-Markovian effects, the resulting error is comparable to that of the Markovian case (EXP1) shown in Fig. 3.

In Fig. 7 we use the same training data of EXP3 to run a new experiment (EXP4) where we train a QRN master equation, by minimising the cost function (12). We consider two cases: in the first one the RNN outputs a single recurrent Lindblad operator $L^{\mu\equiv 1}$. In the second one, the RNN outputs both model both a recurrent Lindblad operator $L^{\mu\equiv 1}$, and the renormalized Hamiltonian, namely the Lamb-shift term H^{LS}. We note that, with a single recurrent Lindblad operator, the error is slightly larger than that of the Markovian case, shown in Fig. 4. However, the resulting error is very low when we include also the correction Hamiltonian H^{LS}. Comparing Fig. 7 with the Markovian case, Fig. 4, one can see that the error grows faster in the non-Markovian regime after the training

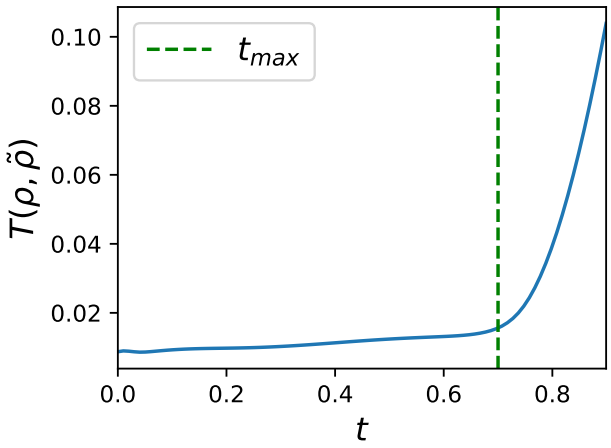

FIG. 6. **(EXP3) Learning the evolution of a noisy two level non-Markovian system:** the mean trace distance $T(\rho(t), \tilde{\rho}(t)) = \frac{1}{N_{\rm P}} \sum_\beta D(\rho_\beta(t), \tilde{\rho}_\beta(t))$ between true states $\rho_\beta(t_j)$ and prediction states $\tilde{\rho}_\beta(t_j)$ as a function of time. The time t_{max} is the maximum time used during training. The experiments run with the following parameters $\lambda^{(1)} = 2$, $\gamma_0^{(1)} = 0.5$ and $\lambda^{(2)} = 1$, $\gamma_0^{(2)} = 0.2$ for Eq. (15), $\omega = 1$ for Eq. (16), and a discretisation interval of $\Delta t = 0.01$. The simulations run over 3000 training examples and predictions were tested over 1000 test examples. All the evolutions run for a time of $t_{max} = 0.7$.

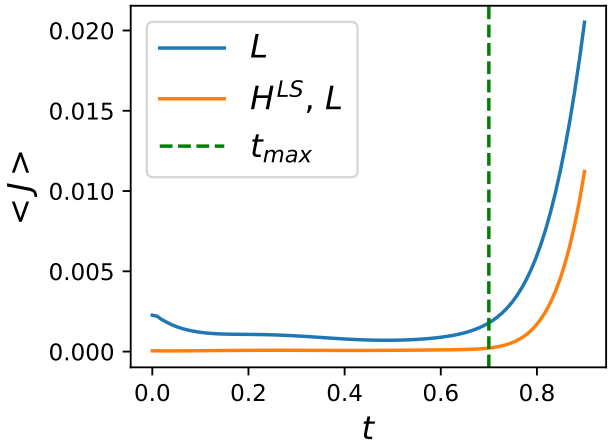

FIG. 7. **(EXP4) Learning a non-Markovian memory kernel:** average value of the cost function $\langle J \rangle$ on unseen examples as a function of time for a learned memory kernel with only the Lindblad Operator $L^{\mu}_{\leq t}$ and with both the Lindblad Operator and the Lamb-shift Hamiltonian $H^{LS}_{\leq t}$. The parameters are the same of Fig. 6. The Frobenius norm squared was used as distance between the matrices.

time $t_{\rm max}$. Overall, in Fig. 7 we see that the RNN which learned both the recurrent Lindblad operator and the Lamb-shift Hamiltonian performed best, and was better able to predict the memory kernel at times greater than those seen during training.

Finally, in Fig. 8 we run a different numerical experiment (EXP5). In EXP5 the training set is composed by data se-

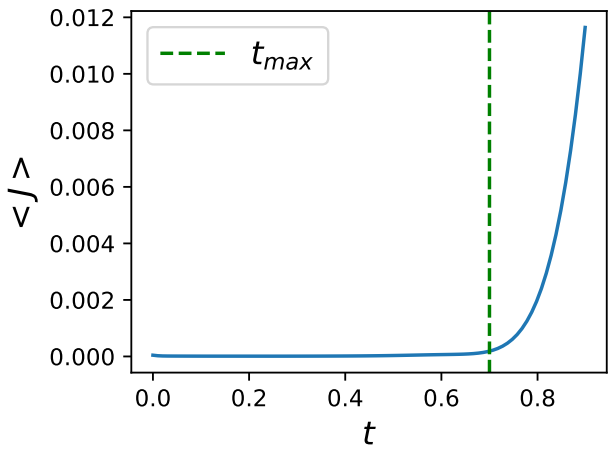

FIG. 8. **(EXP5) Learning a non-Markovian master equation with different ω values:** average value of the cost function $\langle J \rangle$ on unseen examples as a function of time for a learned memory kernel with both the Lindblad operator $L^{\mu}_{\leq t}$ where $\mu = 2$ and the Lamb-shift Hamiltonian $H^{LS}_{\leq t}$. t_{max} specifies the maximum time in the range of times used for training. All the experiments run with the following parameters $\lambda^{(1)} = 2$, $\gamma^{(1)}_0 = 0.5$ and $\lambda^{(2)} = 1$, $\gamma^{(2)}_0 = 0.2$ for Eq. (15), ω uniformly sampled in the interval [0.5, 1.5] for Eq. (16), and a discretisation interval of $\Delta t = 0.01$. The simulations run over 3000 training examples and predictions were tested over 1000 test examples. All the evolutions run for a time of $t_{max} = 0.7$. The Frobenius norm squared was used as distance between the matrices.

quences where the qubit frequency ω in Eq. (16) is not fixed, but rather uniformly sampled from 0.5 to 1.5. The sampled frequency is used as an extra input to the RNN. This corresponds to the experimentally relevant case where the qubit frequency can be externally tuned to a known value. Uncertainty about this frequency can be estimated from the correction Hamiltonian H^{LS}, which is learned from the data. In Fig. 8 we see that the error in EXP5 is remarkably low. Based on the success of this numerical experiment, we propose the following general strategy to introduce prior knowledge about the system. We can define a RNN where the known properties of the system, e.g. its Hamiltonian, are added as an extra input. Training is performed using datasets of quantum state sequences and their respective Hamiltonian, where the latter is sampled from the space of experimentally relevant Hamiltonians. The remarkable accuracy shown Fig. 8 suggests that this procedure forces the RNNs to better explore the manifold of quantum state sequences, to produce a more accurate and robust prediction.

V. CONCLUSIONS AND PERSPECTIVES

We have studied quantum state evolution using RNNs. We have shown that even when the system is interacting with a complicated surrounding environment, RNNs offer an accurate and robust tool for modeling and reconstructing the quantum evolution, both in the Markovian and non-Markovian regime, and even when there are back-scattering effects from the environment.

We have introduced two approaches for modelling open quantum systems with RNNs. In the first one, a deep RNN is trained to learn the entire quantum evolution, namely to reproduce the time sequence $\rho(t_j)$ given an initial state $\rho(0)$. In the second approach, we use a deep RNN to define a non-Markovian master equation where the memory kernel takes a convenient mathematical form, namely that of the GKSL equation. In our master equation the non-Markovian memory effects are taken into account by the structure of the RNN cells. The observed success of our approaches stems from the ability of RNNs to learn temporal sequences, where the future depends on the entire past.

Many extensions of our work are possible. Many-body systems with exponentially large Hilbert spaces could be considered by using RNNs which output a compressed representation of the state, such as tensor networks [64, 65], restricted Boltzmann machines [8] or variational autoencoders [16]. Although RNNs proved to be a powerful tool for modelling open quantum systems, the machine learning literature offers a number of other methods, such as Kalman Filters or models with Gaussian Process transitions, that appear particularly promising. For example, Gaussian Process State Space Models [66] allow one to model prior information on the system and return Bayesian estimates of the uncertainties. Both these features are desirable in a quantum context: prior knowledge on the system, such as the form of the noise-free Hamiltonian, may be used to further reduce the complexity of the model, while approximate values for the uncertainties might enable better control of experimental inaccuracies.

Finally, in more physical terms, it would be interesting to study what happens when *partial* information about the system is available. An experimentally relevant case is when the initial state $\rho(0)$ is known, but one has access to a limited set of expectation values $\langle A_k \rangle_{\rho(t_j)}$, where the observables A_k are not enough to tomographically reconstruct the states $\rho(t_j)$. This possibility may be considered, using our framework, by introducing a cost function between expectation values, rather than between density operators. A further challenge is then to model the disturbance of the measurement onto the system, namely the wave function collapse. This can be done using the process tensor formalism [44], which provides an avenue for generalising our approach in the presence of feedback. It would be interesting to study the performance of RNNs to model these experimentally relevant quantum evolutions.

ACKNOWLEDGMENTS

The authors would like to thank A.D. Ialongo for valuable discussions. L.B. is supported by the UK EPSRC grant EP/K034480/1. E.G. is supported by the UK EPSRC grant EP/P510270/1. A.R. is supported by an EPSRC DTP Scholarship and by QinetiQ. S.S. is supported by the Royal Society, EPSRC, the National Natural Science Foundation of China, and the grant ARO-MURI W911NF17-1-0304 (US DOD, UK MOD and UK EPSRC under the Multidisciplinary University

Research Initiative). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. Part of this work was conducted while A.R. and S.S. were at the Institut Henri Poincaré in Paris.

Appendix A: Gated Recurrent Unit

The GRU state s_t is a linear interpolation of the previous state s_{t-1} and a candidate state $\tilde{s}_t$. The candidate state is a function of the cell input x_t and the previous state given by:

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z s_{t-1} + b_z) \,, \\ r_t &= \sigma(W_r x_t + U_r s_{t-1} + b_r) \,, \\ \tilde{s}_t &= \tanh(W x_t + U(r_t \circ s_{t-1}) + b) \,, \\ s_t &= (1 - z_t) \circ s_{t-1} + z_t \circ \tilde{s}_t \,, \end{aligned} \tag{A1}$$

where $\circ$ denotes the entry-wise (Hadamard) product, σ denotes the sigmoid function, x_t is the input to the cell and W, U and b are trainable parameters, which do not explicitly depend on t. GRU cells can be stacked to form a deep GRU network. The input x_t^j for a depth j cell at time t is the state from the cell in the previous layer s_t^{j-1}.

The GRU state s_t can be thought of as a mixture of the previous state s_{t-1} and the candidate state $\tilde{s}_t$. The weighting of the mixture is controlled by z_t which is known as the *update gate*. We can see that if the entries of z_t are zero then the candidate state is ignored and the previous state becomes the new state.

The candidate state $\tilde{s}_t$ is a function of the previous state s_{t-1} and the current input x_t. The relative contribution of the previous state to the candidate state is controlled by r_t which is known as the *reset gate*.

A more detailed discussion of GRU cells can be found in Ref. [55].

Note that the output of the network is a real vector. In order to encode complex matrices into the RNN we use the following procedure. Assume that we want to encode a matrix $M \in \mathbb{C}^{m \times m}$. M can be decomposed into a real and imaginary part $M = M_{\mathrm{Re}} + iM_{\mathrm{Im}}$. We require the output of the RNN to be a vector $o \in \mathbb{R}^{2m^2}$ such that the first m^2 elements encode in row-major order the entries of M_{Re} and the second m^2 elements encode in row-major order the entries of M_{Im}.

All GRU networks used in this work consisted of 2 GRU layers with output dimension = 40, followed by 1 fully connected layer. The Adam optimizer was used for training with initial learning rate 0.01 [67] and batches of 32 examples. Training was stopped after each example was seen 60 times. In EXP2 the fully connected weights were regularized using weight decay [68], with the L2 penalty factor set to 0.001. The networks were implemented using the Keras 2.2.0 and Tensorflow 1.8 frameworks [69, 70]. Network weights were initialized using the Keras defaults.

[1] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag, Berlin, Heidelberg, 2006).
[2] Y. LeCun, Y. Bengio, and G. Hinton, Nature **521**, 436 (2015).
[3] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, Physical Review Letters **120**, 024102 (2018).
[4] P. T. Komiske, E. M. Metodiev, and M. D. Schwartz, Journal of High Energy Physics **2017**, 110 (2017).
[5] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Nature **549**, 195 (2017).
[6] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, Proc. R. Soc. A **474**, 20170551 (2018).
[7] G. Carleo and M. Troyer, Science **355**, 602 (2017).
[8] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, Nature Physics , 1 (2018).
[9] E. P. van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nature Physics **13**, 435 (2017).
[10] J. Gray, L. Banchi, A. Bayat, and S. Bose, arXiv preprint arXiv:1709.04923 (2017).
[11] J. Carrasquilla and R. G. Melko, Nature Physics **13**, 431 (2017).
[12] P. Mehta and D. J. Schwab, arXiv preprint arXiv:1410.3831 (2014).
[13] Y. Levine, D. Yakira, N. Cohen, and A. Shashua, CoRR, abs/1704.01552 (2017).
[14] Y. Levine, O. Sharir, N. Cohen, and A. Shashua, arXiv preprint arXiv:1803.09780 (2018).
[15] J. Chen, S. Cheng, H. Xie, L. Wang, and T. Xiang, Physical Review B **97**, 085104 (2018).
[16] A. Rocchetto, E. Grant, S. Strelchuk, G. Carleo, and S. Severini, npj Quantum Information **4**, 28 (2018).
[17] L. Banchi, N. Pancotti, and S. Bose, npj Quantum Information **2**, 16019 (2016).
[18] L. Innocenti, L. Banchi, A. Ferraro, S. Bose, and M. Paternostro, arXiv preprint arXiv:1803.07119 (2018).
[19] L. Eloie, L. Banchi, and S. Bose, Phys. Rev. A **97**, 062321 (2018).
[20] U. Las Heras, U. Alvarez-Rodriguez, E. Solano, and M. Sanz, Physical Review Letters **116**, 230504 (2016).
[21] E. Zahedinejad, J. Ghosh, and B. C. Sanders, Physical Review Applied **6**, 054005 (2016).
[22] G. Verdon, J. Pye, and M. Broughton, arXiv preprint arXiv:1806.09729 (2018).
[23] E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. G. Green, and S. Severini, arXiv preprint arXiv:1804.03680 (2018).
[24] A. A. Melnikov, H. P. Nautrup, M. Krenn, V. Dunjko, M. Tiersch, A. Zeilinger, and H. J. Briegel, Proceedings of the National Academy of Sciences , 201714936 (2018).
[25] W. Huggins, P. Patel, K. B. Whaley, and E. M. Stoudenmire, arXiv preprint arXiv:1803.11537 (2018).
[26] M. Benedetti, E. Grant, L. Wossnig, and S. Severini, arXiv preprint arXiv:1806.00463 (2018).
[27] M. Bukov, A. G. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, arXiv preprint arXiv:1705.00565 (2017).
[28] M. Y. Niu, S. Boixo, V. Smelyanskiy, and H. Neven, arXiv preprint arXiv:1803.01857 (2018).
[29] H.-P. Breuer and F. Petruccione, *The theory of open quantum*

systems (Oxford University Press on Demand, 2002).
[30] A. Rivas and S. F. Huelga, *Open Quantum Systems* (Springer, 2012).
[31] M. August and X. Ni, Physical Review A **95**, 012335 (2017).
[32] M. Ostaszewski, J. Miszczak, P. Sadowski, and L. Banchi, arXiv preprint arXiv:1803.05193 (2018).
[33] V. Gorini, A. Kossakowski, and E. C. G. Sudarshan, Journal of Mathematical Physics **17**, 821 (1976).
[34] G. Lindblad, Communications in Mathematical Physics **48**, 119 (1976).
[35] F. Ciccarello, G. Palma, and V. Giovannetti, Physical Review A **87**, 040103 (2013).
[36] M. Ziman, P. Štelmachovič, V. Bužek, M. Hillery, V. Scarani, and N. Gisin, Physical Review A **65**, 042105 (2002).
[37] V. Giovannetti and G. M. Palma, Physical Review Letters **108**, 040401 (2012).
[38] T. Rybár, S. N. Filippov, M. Ziman, and V. Bužek, Journal of Physics B: Atomic, Molecular and Optical Physics **45**, 154006 (2012).
[39] R. McCloskey and M. Paternostro, Physical Review A **89**, 052120 (2014).
[40] S. Campbell, F. Ciccarello, G. M. Palma, and B. Vacchini, arXiv preprint arXiv:1805.09626 (2018).
[41] F. C. Binder, J. Thompson, and M. Gu, Physical Review Letters **120**, 240502 (2018).
[42] J. Cerrillo and J. Cao, Physical Review Letters **112**, 110401 (2014).
[43] F. A. Pollock and K. Modi, arXiv preprint arXiv:1704.06204 (2017).
[44] F. A. Pollock, C. Rodríguez-Rosario, T. Frauenheim, M. Paternostro, and K. Modi, Physical Review A **97**, 012127 (2018).
[45] M. Howard, J. Twamley, C. Wittmann, T. Gaebel, F. Jelezko, and J. Wrachtrup, New Journal of Physics **8**, 33 (2006).
[46] J. Yuen-Zhou, J. J. Krich, M. Mohseni, and A. Aspuru-Guzik, Proceedings of the National Academy of Sciences **108**, 17615 (2011).
[47] B. Bellomo, A. De Pasquale, G. Gualdi, and U. Marzolino, Journal of Physics A: Mathematical and Theoretical **43**, 395303 (2010).
[48] H.-P. Breuer, E.-M. Laine, J. Piilo, and B. Vacchini, Reviews of Modern Physics **88**, 021002 (2016).
[49] A. Ishizaki and G. R. Fleming, The Journal of chemical physics **130**, 234111 (2009).
[50] L. Banchi, G. Costagliola, A. Ishizaki, and P. Giorda, The Journal of chemical physics **138**, 05B609_1 (2013).
[51] S. Mukamel, *Principles of nonlinear optical spectroscopy*, 6 (Oxford University Press on Demand, 1999).
[52] W.-M. Zhang, P.-Y. Lo, H.-N. Xiong, M. W.-Y. Tu, and F. Nori, Physical Review Letters **109**, 170402 (2012).
[53] S. Jang and R. J. Silbey, The Journal of chemical physics **118**, 9312 (2003).
[54] D. Chruściński and A. Kossakowski, Physical Review Letters **104**, 070406 (2010).
[55] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, arXiv preprint arXiv:1406.1078 (2014).
[56] S. Hochreiter and J. Schmidhuber, Neural computation **9**, 1735 (1997).
[57] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, arXiv preprint arXiv:1412.3555 (2014).
[58] Z. C. Lipton, J. Berkowitz, and C. Elkan, arXiv preprint arXiv:1506.00019 (2015).
[59] B. Vacchini, A. Smirne, E.-M. Laine, J. Piilo, and H.-P. Breuer, New Journal of Physics **13**, 093004 (2011).
[60] E. Süli and D. Mayers, *An Introduction to Numerical Analysis* (Cambridge University Press, 2003).
[61] H.-P. Breuer, B. Kappler, and F. Petruccione, Physical Review A **59**, 1633 (1999).
[62] D. Chruściński and S. Maniscalco, Physical Review Letters **112**, 120404 (2014).
[63] B. Bylicka, D. Chruściński, and S. Maniscalco, Scientific reports **4**, 5720 (2014).
[64] J. Cui, J. I. Cirac, and M. C. Banuls, Physical Review Letters **114**, 220601 (2015).
[65] G. Vidal, Physical Review Letters **101**, 110501 (2008).
[66] R. Frigola, Y. Chen, and C. E. Rasmussen, in *Advances in Neural Information Processing Systems* (2014) pp. 3680–3688.
[67] D. P. Kingma and J. Ba, arXiv preprint arXiv:1412.6980 (2014).
[68] A. Krogh and J. A. Hertz, in *Advances in neural information processing systems* (1992) pp. 950–957.
[69] F. Chollet *et al.*, "Keras," `https://keras.io` (2015).
[70] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, in *OSDI*, Vol. 16 (2016) pp. 265–283.