Quantum Machine Intelligence (2022) 4: 11
https://doi.org/10.1007/s42484-022-00068-y

RESEARCH ARTICLE

Quantum reinforcement learning: the maze problem

Nicola Dalla Pozza[1,2,3] · Lorenzo Buffoni[2,3] · Stefano Martina[2,3] · Filippo Caruso[2,3,4]

Received: 5 November 2021 / Accepted: 5 April 2022 / Published online: 27 May 2022

Abstract

Quantum machine learning (QML) is a young but rapidly growing field where quantum information meets machine learning. Here, we will introduce a new QML model generalising the classical concept of reinforcement learning to the quantum domain, i.e. quantum reinforcement learning (QRL). In particular, we apply this idea to the maze problem, where an agent has to learn the optimal set of actions in order to escape from a maze with the highest success probability. To perform the strategy optimisation, we consider a hybrid protocol where QRL is combined with classical deep neural networks. In particular, we find that the agent learns the optimal strategy in both the classical and quantum regimes, and we also investigate its behaviour in a noisy environment. It turns out that the quantum speedup does robustly allow the agent to exploit useful actions also at very short time scales, with key roles played by the quantum coherence and the external noise. This new framework has the high potential to be applied to perform different tasks (e.g. high transmission/processing rates and quantum error correction) in the new-generation noisy intermediate-scale quantum (NISQ) devices whose topology engineering is starting to become a new and crucial control knob for practical applications in real-world problems. This work is dedicated to the memory of Peter Wittek.

Keywords Quantum walks · Reinforcement learning · Quantum machine learning · Maze

1 Introduction

The broad field of machine learning (Bishop 2011; Cover and Thomas 1991; Hastie et al. 2009) aims to develop computer algorithms that improve automatically through experience with lots of cross-disciplinary applications from domotics systems to autonomous cars, from face/voice recognition to medical diagnostics. Self-driving systems can learn from data, so as to identify distinctive patterns and make consequently decisions, with minimal human intervention. Its three main paradigms are *supervised learning*, *unsupervised learning* and *reinforcement learning* (RL). The goal of a supervised learning algorithm is to use an output-labeled dataset $\{x_i, y_i\}_{i=1}^N$, to produce a model that, given a new input vector x, can predict its correct label y. Unsupervised learning, instead, uses an unlabelled dataset $\{x_i\}_{i=1}^N$ and aims to extract some useful properties (patterns) from the single datapoint or the overall data distribution of the dataset (e.g. clustering). In reinforcement learning (Sutton and Barto 2018), the learning process relies on the interaction between an agent and an environment and defines how the agent performs his actions based on past experiences (episodes). In this process, one of the main problems is how to resolve the tradeoff between *exploration* of new actions and *exploitation* of learned experience. RL has been applied in many successful tasks, e.g. outperforming humans on Atari games (Mnih et al. 2015) and GO (Silver et al. 2016) and recently it is becoming popular in the contexts of autonomous driving (Kiran et al. 2020) and neuroscience (Botvinick et al. 2020).

In recent years, lots of efforts have been directed towards developing new algorithms combing machine learning and quantum information tools, i.e. in a new research field known as quantum machine learning (QML) (Schuld et al. 2015; Wittek 2014; Adcock et al. 2015; Arunachalam and de Wolf 2017; Biamonte et al. 2017), mostly in the supervised

✉ Filippo Caruso
filippo.caruso@unifi.it

[1] Scuola Normale Superiore, Piazza dei Cavalieri 7, I-56126 Pisa, Italy

[2] Department of Physics and Astronomy, University of Florence, via Sansone 1, I-50019 Sesto Fiorentino, Italy

[3] LENS - European Laboratory for Non-Linear Spectroscopy, via Carrara 1, I-50019 Sesto Fiorentino, Italy

[4] QSTAR and CNR-INO, I-50019 Sesto Fiorentino, Italy

(Neven et al. 2008; Mott et al. 2017; Lloyd et al. 2020; Martina et al. 2022) and unsupervised domain (Otterbach et al. 2017; Winci et al. 2020; Hu et al. 2019), both to gain an *advantage* over classical machine learning algorithms and to *control* quantum systems more effectively. Some preliminary results on QRL have been reported in refs. Dong and Chen (2005); Paparo et al. (2014) and more recently for closed (i.e. following unitary evolution) quantum systems in ref. Dunjko et al. (2016) where the authors have shown quadratic improvements in learning efficiency by means of a Grover-type search in the space of the rewarding actions. Similarly, ref. Saggio et al. (2021) have shown how to get quantum speedups in reinforcement learning agents. The setting of an agent acting on an environment, however, has a natural analogue in the framework of open quantum systems (Breuer and Petruccione 2002; Caruso et al. 2014), where one can embed the entire RL framework into the quantum domain, and this has not been investigated in literature yet. Moreover, one of the authors of this manuscript, inspired by recent observations in biological energy transport phenomena (Caruso et al. 2009), has shown in ref. Caruso et al. (2016) that one can obtain a very remarkable improvement in finding a solution of a problem, given in terms of the exit of a complex maze, by playing with quantum effects and noise. This improvement was about five orders of magnitude with respect to the purely classical and quantum regimes for large maze topologies. In the same work, their results were also experimentally tested by means of an integrated waveguide array, probed by coherent light.

Motivated by these previous works, here we define the building blocks of RL in the quantum domain but in the framework of open (i.e. noisy) quantum systems, where coherent and noise effects can strongly cooperate together to achieve a given task. Then, we apply it to solve the quantum maze problem that, being a very complicated one, can represent a crucial step towards other applications in very different problem-solving contexts.

2 Reinforcement learning

In RL, the system consists of an agent that operates in an environment and gets information about it, with the ability to perform some actions in order to gain some advantage in the form of a reward. More formally, RL problems are defined by a 5-tuple $(S, A, P_{\cdot}(\cdot,\cdot), R_{\cdot}(\cdot,\cdot), \gamma)$, where S is a finite set of states of the agent, A is a finite set of actions (alternatively, A_s is the finite set of actions available from the state s), $P_a(s, s') = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$ is the probability that action a in state s at time t will lead to the state s' at time $t + 1$, $R_a(s, s')$ is the immediate reward (or expected immediate reward) received after transitioning from state s to state s', due to action a, and $\gamma \in [0, 1]$ is the discount factor balancing the relative importance of present and future rewards. In this setting, one can introduce different types of problems, based on the information one has at disposal. In *multi-armed bandit models*, the agent has to maximise the cumulative reward obtained by a sequence of independent actions, each of which giving a stochastic immediate reward. In this case, the state of the system describes the uncertainty of the expected immediate reward for each action. In *contextual multi-armed bandits*, the agent faces the same set of actions but in multiple scenarios, such that the most profitable action is scenario-dependent. In a *Markov decision process* (MDP), the agent has information on the state and the actions have an effect on the state itself. Finally, in *partially observable MDPs*, the state s is partially observable or unknown.

The goal of the agent is to learn a policy (π) that is a rule according to which an action is selected. In its most general formulation, the choice of the action at time t can depend on the whole history of agent-environment interactions up to t, and is defined as a random variable over the set of available actions if such choice is stochastic. A policy is called Markovian if the distribution depends only on the state at time t, with $\pi_t(a|s)$ denoting the probability to choose the action a from such state s, and if a policy does not change over time it is referred as stationary (Ghavamzadeh et al. 2015). Then, the agent aims to learn the policy that maximises the expected cumulative reward that is represented by the so-called value function. Given a state s, the value function is defined as $V^\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(Z_t)|Z_0 = (s, \pi(.|s))]$, where Z_t is a random variable over state-action pairs. The policy π giving the optimal value function $V^*(s) = \sup_\pi V^\pi(s)$ is the RL objective. It is known (Sutton and Barto 2018; Ghavamzadeh et al. 2015) that the optimal value function $V^*(s)$ has to satisfy the Bellman equation, i.e. $V^\pi(s) = R^\pi(s) + \gamma \int_S P^\pi(s'|s)V^\pi(s')ds'$. In deep RL, the policy is learned by a deep neural network whose objective function is the Bellman equation itself. The network starts by randomly exploring the space of possible actions and iteratively reinforcing its policy through the Bellman equation given the reward obtained after each action. A popular approach to transition from a purely random exploration to a conclusive reinforced policy can be achieved via an ε-greedy policy, which chooses a random action with probability $0 < \varepsilon \ll 1$ and the provisional optimal one with probability $1 - \varepsilon$. Furthermore, a smooth transition can be obtained with a time-dependent slow decay of the parameter ε. A pictorial view of the iterative process between the agent and the environment can be found in Fig. 1.

3 Quantum maze

Here we transfer the RL concepts into the quantum domain where both the environment and the reward process follow the laws of quantum mechanics and are affected by

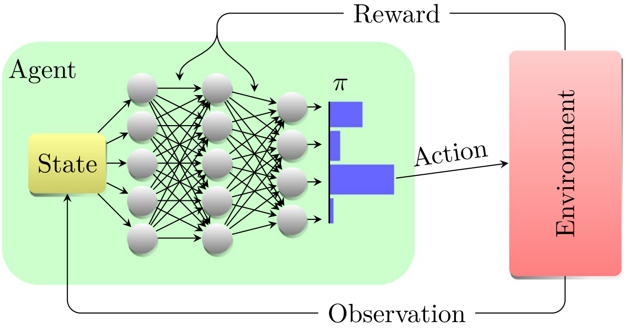

Fig. 1 Deep reinforcement learning scheme. A deep neural network learns the policy π that the agent uses to perform an action on the environment. A reward and the information about the new state of the system are given back to the agent that improves and learns its policy accordingly

both coherent and incoherent mechanisms. We consider, for simplicity, a quantum walker described by a qubit that is transmitted over a quantum network representing the RL environment. The RL state is the quantum state over the network, represented by the so-called density operator ρ. The RL actions are variations of the environment, e.g. its network topology, that will affect the system state through a noisy quantum dynamics. The reward process is obtained from the evolution of the quantum network and hence associated to some probability function to maximise. Following the results in ref. Caruso et al. (2016) and just to test this framework on a specific model, we consider a *perfect* maze, i.e. a maze where there is a single path connecting the entrance with the exit port. The network dynamics is described in terms of a stochastic quantum walk model (Whitfield et al. 2010; Caruso 2014), whose main advantage here is that, within the same model, it allows to consider a purely coherent dynamics (quantum walk), a purely incoherent dynamics (classical random walk), and also the hybrid regime where both coherent and incoherent mechanisms interplay or compete with each other. Although it is very challenging to make a fair comparison between QRL and RL as applied to the same task and it is out of the scope of this paper, the model we consider here allows us to have the non-trivial chance to analyse the performances of the classical and quantum RL models respectively but in terms of the same resources and degrees of freedom. Very recently we have also exploited this model to propose a new transport-based (neural network-inspired) protocol for quantum state discrimination (Dalla Pozza and Caruso 2020).

According to this stochastic quantum walk model, the time evolution t of the walker state ρ is governed by the following Lindblad equation (Lindblad 1976; Whitfield et al. 2010; Caruso 2014):

$$\frac{d\rho}{dt} = (1-p)\,\mathcal{L}_{QW}(\rho) + p\,\mathcal{L}_{CRW}(\rho) + \mathcal{L}_{exit}(\rho) \tag{1}$$

where $\mathcal{L}_{QW}(\rho) = -i[A, \rho]$ describes the coherent hoping mechanisms, $\mathcal{L}_{CRW}(\rho) = \sum_{i,j} L_{ij}\rho L_{ij}^{\dagger} - \frac{1}{2}\{L_{ij}^{\dagger}L_{ij}, \rho\}$ with $L_{ij} = (A_{ij}/d_j)|i\rangle\langle j|$ describes the incoherent hopping ones, while $\mathcal{L}_{exit}(\rho) = 2|n+1\rangle\langle n|\rho|n\rangle\langle n+1| - \{|n\rangle\langle n|, \rho\}$ is associated to the irreversible transfer from the maze (via the node n) to the exit (i.e., a sink in the node $n+1$). Here the maze topology is associated to the so-called adjacency matrix of the graph A, whose elements A_{ij} are 1 if there is a link between the node i and j, and 0 otherwise. Besides, d_j is the number of links attached to the node j, while $|i\rangle$ is the element of the basis vectors (in the Hilbert space) corresponding to the node i. The parameter p describes how much incoherent the walker evolution is. In particular, when $p = 1$ one recovers the model of a classical random walk, when $p = 0$ one faces with a quantum walk, while when $0 < p < 1$ the walker hops via both incoherent and coherent mechanisms (stochastic quantum walker). Let us point out that the complex matrix $\rho_{ij} \equiv \langle i|\rho|j\rangle$ contains the node (real) populations along the diagonal, and the coherence terms in the off-diagonal (complex) elements. More in general, in order to have a physical state, the operator ρ has to be positive semi-definite (to have meaningful occupation probabilities) and with trace one (for normalised probabilities). Hence, in this basis, only for a classical state ρ_{ij} is a fully diagonal matrix. Then, the escaping probability is measured as $p_{exit}(t) = 2\int_0^t \rho_{nn}(t')\mathrm{d}t'$. Ideally, we desire to have $p_{exit} = 1$ in the shortest time interval, meaning that with probability 1 the walker has left the maze.

In the RL framework, $\rho(t)$ is the state s_t evolving in time, the environment is the maze, and the objective function is the probability p_{exit} that the walker has exited from the maze in a given amount of time (to be maximised), or, in an equivalent formulation of the problem, the amount of time required to exit the maze (to be minimised). In this paper, we consider the former objective function. The actions are obtained by changing the environment, that is, by varying the maze adjacency matrix. More specifically, we consider three possible actions a performed at given time instants during the walker evolution: (i) building a new wall, i.e. A_{ij} is changed from 1 to 0 (removing a link); (ii) breaking through an existing wall, i.e. A_{ij} is changed from 0 to 1 (adding a new link); (iii) doing nothing (null action) and letting the environment evolve with the current adjacency matrix. The action (i) may allow the walker to waste time in dead-end paths, while the action (ii) may create shortcuts in the maze — see Fig. 2. Notice that the available actions a are indexed with the link to modify, so that the action space is discrete and finite. In the following, we set the total number of actions to be performed during the transport dynamics. In principle, one could add a penalty (negative term in the reward) in order to let the learning minimise the total number of actions (which might be energy consuming physical processes). The immediate reward $R_a(s, s')$ is the incremental probability that the walker has left the maze in the time interval Δt following the action a changing the state from $\rho(t)$ to $\rho(t + \Delta t)$. This is an MDP setting. The optimal policy π gives the optimal actions maximising the cumulative escaping probability. Besides, one could also optimise the noise parameter p but we have decided to keep it fixed and run the learning for each value of p in the range [0, 1].

This approach is slightly different from the scenario pictured in the traditional maze problem (classical RL). A classical educational example is provided, for instance, by a mouse (the agent) whose goal is to find the shortest route from some initial cell to a target cheese cell in a maze (the environment). The agent needs to experiment and exploit past experiences in order to achieve its goal, and only after lots of trials and errors it will solve the maze problem. In particular, it has to find the optimal sequence of states in which the accumulated sum of rewards is maximal, for instance considering a negative reward (penalty) for each move on free cells in the maze. This is indeed an MDP setting, where the possible actions are the agent moves (left, right, up, down). In our case, we face instead with a probability distribution to find the walker on the maze positions, while in the classical setting the corresponding state would be a diagonal matrix ρ_{ii} where only one element is equal to 1 and the others are vanishing. Our setup introduces an increased complexity with respect the classical case, in both the definition of the state and in the number of available actions. In addition, a quantum walker can move in parallel along different directions (quantum parallelism), as due to the quantum superposition principle in quantum physics, and interfere constructively or destructively on all maze positions, i.e. the quantum walker behaves as an electromagnetic or mechanical wave travelling through a maze-like structure (wave-particle duality). For these reasons, it is more natural to consider topology modifications (i.e. in the hopping rates described by A_{ij}) as possible actions. However, let us point out that changing the hopping rate is qualitatively similar to the process of forcing the walker to move more in one or in the other direction, hence mimicking the continuous-version of the discrete moves for the mouse in the classical scenario.

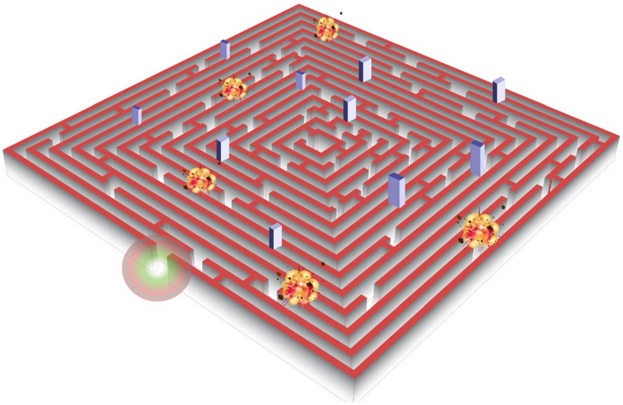

Fig. 2 Pictorial view of a maze where a classical/quantum walker can enter the maze from a single input port and escape through a single output port. In order to increase the escaping probability within a certain time, at given time instants, the RL agent can modify the environment (maze topology) breaking through existing walls and/or building new walls while the walker moves around to find the exit as quick as possible

4 Results

Within the setting described above, we set a time limit T for the overall evolution of the system and define the time instants $t_k = k\tau$, with $\tau = \Delta t = T/N$ and $k = 0, \dots N - 1$, when the RL actions can be performed. The quantum walker evolves according to Eq. 1 in the time interval between t_k and t_{k+1}. We then implement deep reinforcement learning with ε-greedy algorithm for the policy improvement, and run it with $N = 8$ actions and with 1000 training epochs (see Methods for more technical details). At each time instant t_k the agent can choose to modify whatever link in the maze, albeit we would expect its actions to be localised around the places where it has the chance to further increase the escaping probability. The ε-greedy algorithm implies that the agent picks either the action suggested by the policy with probability $1 - \varepsilon$ or a random action with probability ε. This method increases the chances of the policy to explore different strategies searching for the best one instead of just reinforcing a sub-optimal solution. The value of ε is slowly decreased during training so that, at the end, the agent is just applying the policy without much further exploration. This optimisation is repeated for different values of p and T in order to investigate their role in the learning process.

As shown in Fig. 3, there is a clear RL improvement for any value of p especially for large T (i.e. also large τ), while for small T it occurs mainly in the quantum regime (i.e. p going to 0) when the walker exploits coherent (and fast) hopping mechanisms. This is due to the fact that the classical random walker (without RL) moves very slowly and remains close to the starting point for small T, as reported in ref. Caruso et al. (2016). Repeating this experiment for 30 random 6×6 perfect mazes, we find a very similar behaviour — see also Fig. 6 in Methods where interestingly a dip in the cumulative reward enhancement is shown at around $p = 0.1$ where the interplay between quantum coherence and noise allows to optimise the escaping probability without acting on the maze (Caruso 2014). There it was very remarkable to observe that a small amount of noise allows the walker to both keep its quantumness (i.e. moving in parallel over

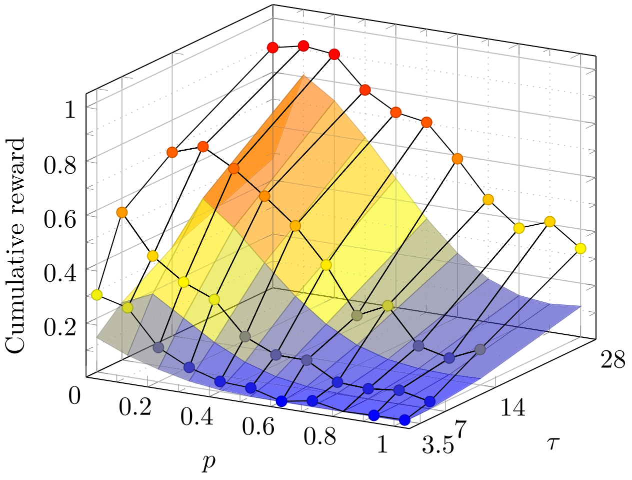

Fig. 3 Cumulative reward as a function of p and τ, for a given 6×6 perfect maze and $N = 8$ actions, equally spaced in time by the amount τ. The time unit is given in terms of the inverse of the sink rate set to 1. The dotted grid above represents the performance of the quantum walker after the training, while the coloured solid surface below is the baseline on the same maze with no actions performed by the agent (only free evolution). Repeating the training on over 30 random 6×6 mazes and averaging their performances for each (p, τ), we qualitatively obtain the same trend

the entire maze) and learn the shortest path to the exit from the maze.

Figure 4 shows an example of cumulative rewards obtained from the training of a network while the agent explores the space of the possible actions. Initially some random actions are performed, and soon the agent finds some positive reinforcement and learns to consistently apply better actions outperforming the case with no actions.

The proposed way of *learning* the best actions also comes with an intrinsic robustness to stochastic noise. Indeed this is a crucial property of RL-based approaches. In our case, we can suppose that we do not have perfect control on the system and there might be perturbations, for example in the timing at which the actions are effectively performed. These kinds of perturbations are in general detrimental for hard-coded optimisation algorithms, and we want to analyse how our QRL approach performs in this regard. To check this, we first train the agent in an environment with fixed τ and p. Afterwards, we evaluate the performance of the trained agent in an environment where the time τ at which the actions are performed becomes stochastic (noisy). This additional noise in the time is controlled by a parameter $0 \leq \eta \leq 1$ while the total time of the actions is kept fixed. In this setting, we observe a remarkable robustness of our agent that is capable of great generalisation and keep the cumulative reward almost constant despite of the added stochasticity. Indeed, in Fig. 5, we plot the average reward obtained by the agent in this stochastic environment over 100 different realisations of the noise. We can see that as we increase the parameter η our agent, on average, keeps the ability to find the correct actions in order to make the reward consistent even in a stochastic environment, even though it has not been retrained in the noisy setting. However, while the average reward remains stable, the difference between the minimum and maximum reward increases significantly as η increases.

The other tested scenario is the one in which, instead of taking the actions equally spaced in the total evolution time, we concentrate them all at the beginning or at the end of the evolution. This gives our agent a different environment at which it adapts once again implementing different strategies. Indeed, we find that our training method is applicable with success also in this more general scenario thus

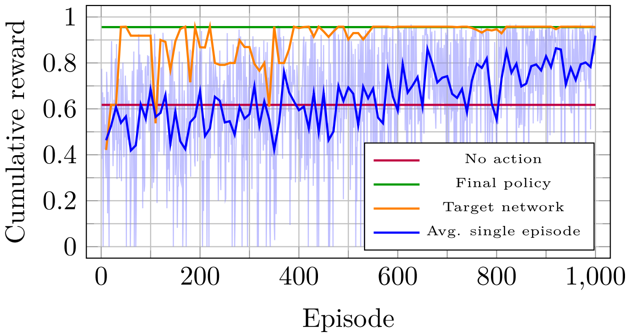

Fig. 4 Training curves for an agent performing RL actions for $p = 0.4, \tau = 28$, and $N = 8$ actions on a 6×6 perfect maze. The curves show the cumulative rewards from single episodes (light blue), ten-episode window average (dark blue) and for the target network (orange) — see RL optimisation in Methods. The two horizontal lines are the (constant) cumulative reward in the case of no RL actions (magenta) and for the final trained policy (green)

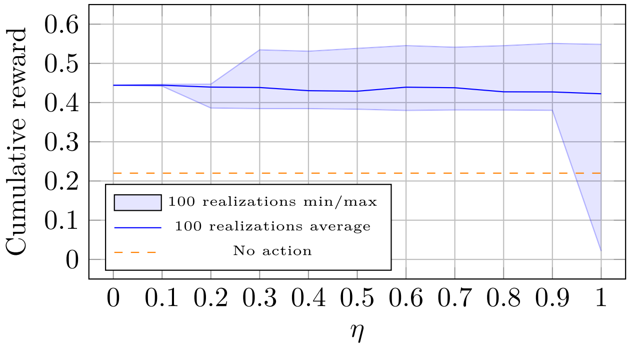

Fig. 5 Cumulative reward of an agent trained at $\tau = 14$ and $p = 0.4$ and deployed in a stochastic environment controlled by the parameter η. The solid line is the average reward obtained by the agent over 100 realisations of the noise, the shaded area represents the minimum and the maximum achieved reward and the dashed orange line is the baseline of the walker with no actions performed. While the average performance of the agent remains stable, the variance in the outcomes increases greatly as η increases

concluding our remarks on the robustness of the proposed QRL implementation.

A detailed discussion of the robustness analysis and all the aforementioned experiments can be found in the Appendix.

5 Methods

5.1 Quantum maze simulation

To simulate the stochastic quantum walk on a maze, we have used the popular QuTiP package (Johansson et al. 2012) for Python. In order to account for the actions performed by the agent at time instants $t_k = k\tau$ modifying the network topology and to evaluate the reward signal, we have wrapped the QuTiP simulator in a Gym environment (Brockman et al. 2016). Gym is a python package that has been created by OpenAI specifically to tackle and standardise reinforcement learning problems. In this way, we can apply any RL algorithms on our quantum maze environment. The initial maze could be randomly generated or loaded from a fixed saved adjacency matrix in order to account for both the reproducibility of single experiments and the averaging over different configurations.

5.2 RL optimisation

We have used a feed-forward neural network to learn the policy of our agent, following the Deep Q Learning approach (Stooke and Abbeel 2019), realised with the PyTorch package for python (Paszke et al. 2019). In this approach, at each iteration of the training loop defining a training epoch, a new training episode is evaluated by numerically solving Eq. 1 for the time evolution and employing an ε-greedy policy for the action selection. The new training episode is recorded in a fixed-dimension pool of recent episodes called replay memory, from which, after every new addition, a random batch of episodes is selected to train the policy neural network. The ε parameter is reduced at each epoch, in order to reduce the exploration of new action sequences and increase the exploitation of the good ones proposed by the policy neural network. Periodically, the policy neural network is copied in a target neural network, i.e. a trick used to reduce the instabilities in the training of the policy neural network. Figure 4 shows the reward of the training episodes, their ten-episode window average, the reward provided by target network, alongside the free evolution (no RL actions), and final reward (constant lines) provided by the trained target network.

Despite the relative simple architecture, we have found the training to be quite sensitive to the choice of learning hyper-parameters, such as the batch size of the training episodes per epoch, the replay memory capacity, the rate of target network update and the decay rate of ε in the ε-greedy policy. In particular, in Fig. 3 for each (p, τ), we run multiple independent hyper-parameter optimisations and training, employing the libraries Hyperopt (Bergstra et al 2013) and Tune (Liaw et al. 2018). Due to the small size of the networks, we were able to launch multiple instances of our training procedure using a single Quadro K6000 GPU. Figure 6 shows the mean cumulative reward improvement between the no-action strategy (only free evolution) and the trained strategy over 30 random perfect mazes (size 6 × 6) with $N = 8$ actions.

6 Discussion

To summarise, here we have introduced a new QML model bringing the classical concept of reinforcement learning into the quantum domain but also in presence of external noise. An agent operating in an environment does experiment and exploit past experiences in order to find an optimal sequence of actions (following the optimal policy) to perform a given task (maximising a reward function). In particular, this was applied to the maze problem where the agent desires to optimise the escaping probability in a given time interval. The dynamics on the maze was described in terms of the stochastic quantum walk model, including exactly also the purely classical and purely quantum regimes. This has allowed to make a fair comparison between transport-based RL and QRL models exploiting the same resources. We have found that the agent always learns a strategy that allows a quicker escape from the maze, but in the quantum case the walker is faster and can exploit useful actions also at very short

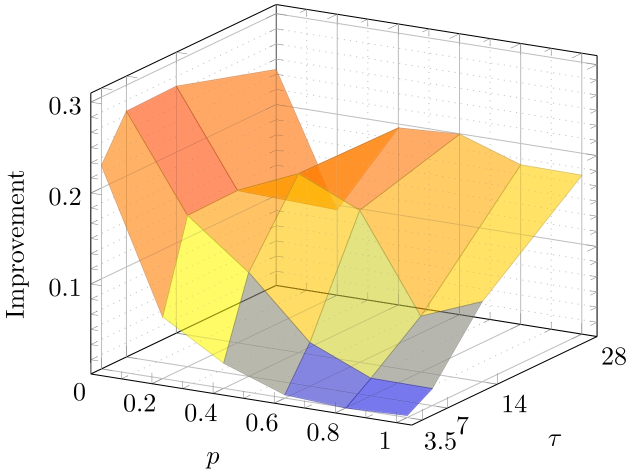

Fig. 6 Cumulative reward improvement over the no RL action (free evolution) dynamics as a function of p and τ, averaged over 30 random perfect mazes (6 × 6 size) and $N = 8$ (equally spaced in time) actions

times. Instead, in presence of a small amount of noise, the transport dynamics is already almost optimal and RL shows a smaller enhancement, hence further supporting the key role of noise in transport dynamics. In other words, some decoherence effectively reproduces a sort of RL optimal strategy in enhancing the transmission capability of the network. Moreover, the presence of more quantumness in our QRL protocol leads to have more robustness in the optimal reward with respect to the exact timing of the actions performed by the agent.

Finally, let us discuss how to possibly implement the RL actions in the maze problem from the physics point of view. In ref. Caruso (2014), one of us has shown that one can design a sort of noise mask that leads to a transport behaviour as if one had modified the underlying topology. For instance, dephasing noise can open shortcuts between non-resonant nodes, and Zeno-like effects can suppress the transport over a given link, hence mimicking the two types of RL actions discussed in this paper. As future outlooks, one could test these theoretical predictions via atomic or photonic experimental setups or even on the new-generation NISQ devices whose current technologies today allow to engineer complex topologies and modify them in the same time scale of the quantum dynamics while also exploiting some beneficial effects of the environmental noise that cannot be suppressed.

Appendix 1. Generalisation of the neural network training

After training the learning algorithm, we have verified its generalisation properties on unseeing parameter pairs $(p,\ \tau)$. Namely, we have applied the neural network $\mathcal{N}$ trained for (p', τ') on all the $(p,\ \tau)$ grid. An example of the cumulative reward obtained from this comparison is depicted in Fig. 7, where we represent with a coloured surface the performance of the free evolution, and in a black mesh surface the cumulative reward of the neural network trained for $p' = 0,\ \tau' = 14$. The figure, to be compared with Fig. 3 of the main text, is qualitatively similar, meaning that a single neural network is able to generalise the behaviour of other networks trained for each $(p,\ \tau)$. Note that the optimal sequences proposed by $\mathcal{N}$ are indeed different depending on $(p,\ \tau)$ (though they may share similar patterns), and that in general the optimal sequences of actions are optimal only locally. We have tested this latter hypothesis running all the optimal sequences proposed by all the trained neural networks for all grid points $(p,\ \tau)$. The cumulative reward obtained is plotted in Fig. 8, where we can observe that a small number of optimal sequences cover all the grids (the same sequence is related to the same colour of the marker). Despite not being an exhaustive check for all the possible sequences, this gives evidence that a sequence is optimal only locally.

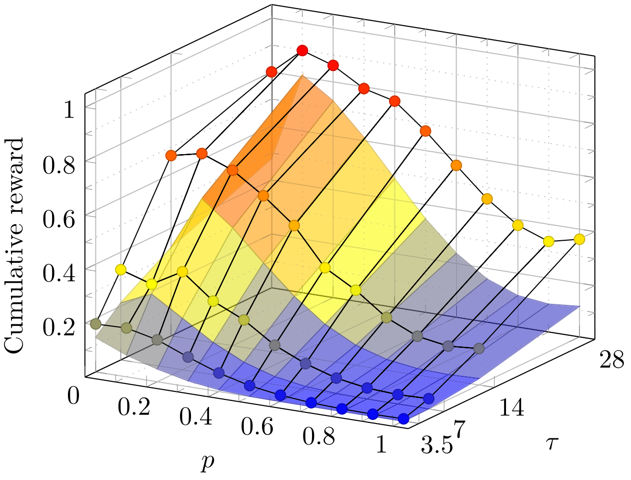

Fig. 7 Cumulative reward of a neural network $\mathcal{N}$ trained for $p' = 0, \tau' = 14$ and then tested on all the (p, τ) grid (black dotted mesh). The solid coloured surface gives the cumulative reward of the free evolution

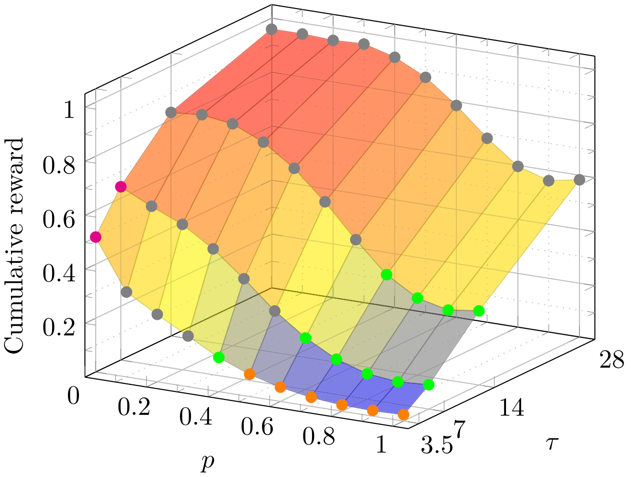

Fig. 8 Cumulative reward obtained maximising for each (p, τ) the cumulative reward from all the optimal sequences suggested by all the trained neural networks. Of all the sequences, four are sufficient to give the maximum cumulative reward, and are identified by the coloured markers (grey, magenta, green and orange)

Appendix 2. Robustness

To further test the robustness of our trained agent, we checked its performances in a stochastic environment where the time interval between the actions can fluctuate. The agent is thus forced to adapt its strategy to the new environment and, as we can observe, in Fig. 9 it does this surprisingly well. The agents have been first trained in a noiseless environment with $\tau = 14$ and $p \in [0, 1]$. The additional noise in the time is controlled by a parameter $0 \leq \eta \leq 1$ while the total time of the actions is kept fixed. In detail, for a set of $N = 8$ actions, we sample 8 random numbers in the interval $[-\eta\tau,\ \eta\tau]$ obtaining a noise vector $\bar{\eta}$, which is then averaged to zero in order to keep the total time of the walker constant. This vector gives the variation to apply to each time instant t_k where the actions are performed. In Fig. 9, we plot the average reward obtained by the agent in this noisy environment over 100 different realisations of the noise. We can see that as we increase the noise parameter η our agent keeps the ability to find the correct actions in order to make the reward consistent even in a noisy environment, even though it has not been retrained in the noisy setting. This analysis on the robustness to noise in time further proves the capability of our approach to generalise well to different environments.

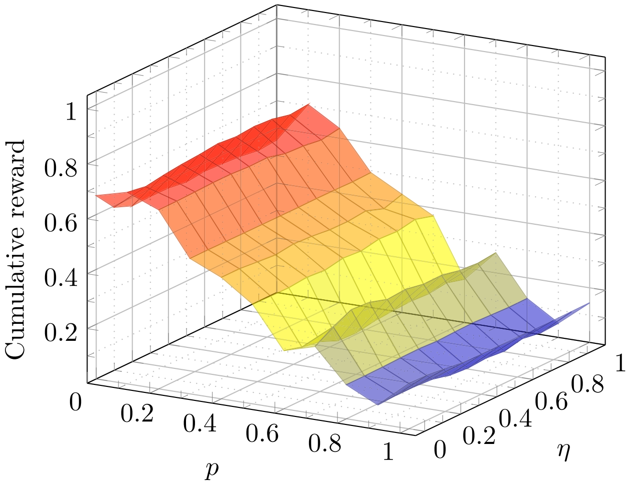

Fig. 9 Cumulative reward of an agent trained at $\tau = 14$ and deployed in a noisy environment where the noise is controlled by the parameter η. The reported reward is the average performance of the agent over 100 different realisations of the noise

Appendix 3. RL actions timing

Finally, we analyse the scenario when one introduces a transient time before or after the set of equally time-spaced actions. Namely, we consider a total time evolution of $T = 8 \times 28 = 224$, and split it in $T = T_1 + T_2 + T_3$ where T_1 is a transient time with free evolution before applying the actions, $T_2 = N \times \tau$ is the time interval applying the actions spaced by τ, and T_3 is a transient time of free evolution after the actions. The results in Fig. 10 show that our training method is applicable also in this more general scenario, and we can also observe the role of the time instant to perform the action. In fact, accumulating the actions at the beginning (Fig. 10b) and at the end (Fig. 10a) of the dynamics seems to lead to a suboptimal strategy, where the improvements are more difficult to occur. Of course, the extreme case of performing the actions at the very end shows no improvement with respect to the no-action strategy (Fig. 10a for large T_1). We also find that for low values p (meaning more quantumness of the walker) the time at which we do the actions is clearly less important than for large values of p, where a different timing of the actions can result in a drastic reduction of the improvements over the baseline. This result proves the robustness of the quantum regime with respect to the classical one.

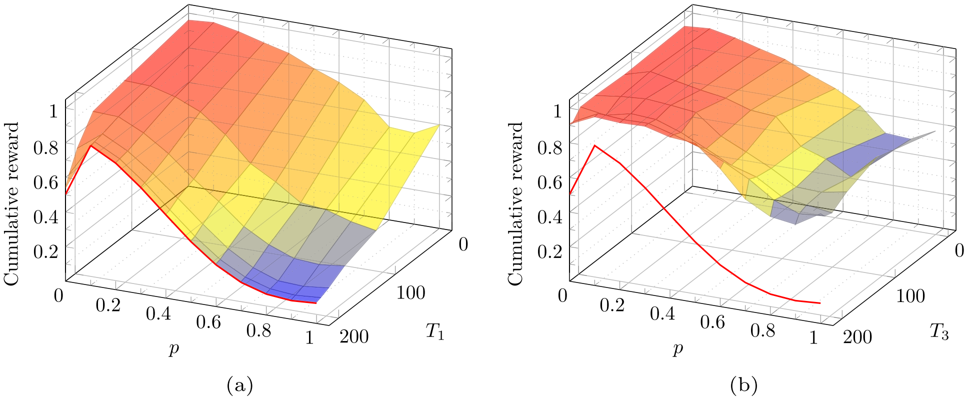

Fig. 10 Cumulative reward for a fixed total time evolution $T = T_1 + T_2 + T_3 = 8 \times 28 = 224$, with T_1 the free evolution interval before the application of the actions, $T_2 = 8 \times \tau$ the interval where the actions spaced by τ are performed (τ is evaluated given the values T, T_1, T_3), and T_3 the free evolution interval after the actions. The time unit is given in terms of the inverse of the sink rate set to 1. The cumulative reward for the no-actions strategy (only free evolution in T) is drawn in a red solid line. **a** Cumulative reward as a function of p and T_1 with $T_3 = 0$. Notice that in the limit where $T_1 = 224$ the actions are packed at the end of the time evolution where they become irrelevant, thus recovering the no-action case. **b** Cumulative reward as a function of p and T_3 with $T_1 = 0$

Acknowledgements This work has much benefited from the initial contribution of Peter Wittek and very stimulating discussions with him before he left us, and is dedicated to his memory. In February 2019, during a workshop on “Ubiquitous Quantum Physics: the New Quantum Revolution I” at ICTP in Trieste, Peter and F.C. had indeed started a new collaboration focused on the development of the new concept of quantum reinforcement learning that, unfortunately alone, we have carried out later and finally discussed in this paper.

Author contribution N.D.P. and F.C. led and carried out the theoretical work. N.D.P., L.B. and S.M. performed the numerical simulations and optimisations. F.C. conceived and supervised the whole project. All authors contributed to the discussion, analysis of the results and the writing of the manuscript.

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement. This work was financially supported from Fondazione CR Firenze through the project QUANTUM-AI, the European Union’s Horizon 2020 research and innovation programme under FET-OPEN Grant Agreement No. 828946 (PATHOS), and from University of Florence through the project Q-CODYCES.

Data availability The code supporting the results of the paper is freely available at https://github.com/Buffoni/quantum_maze_learning.

Declarations This work does not involve human participants and presents no ethical concerns.

Conflict of interest The authors declare no competing interests.

References

Adcock J, Allen E, Day M, Frick S, Hinchliff J, Johnson M, Morley-Short S, Pallister S, Price A, Stanisic S (2015) Advances in quantum machine learning. arXiv:1512.02900

Arunachalam S, de Wolf R (2017) A survey of quantum learning theory. arXiv:1701.06806

Bergstra J, Yamins D, Cox D (2013) Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. Dasgupta S, McAllester D (eds.) Proceedings of the 30th International Conference on Machine Learning. Proceedings of Machine Learning Research. PMLR, Atlanta, Georgia, USA. 28:115–123. http://proceedings.mlr.press/v28/bergstra13.html

Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S (2017) Quantum machine learning. Nat 549:195–202. https://doi.org/10.1038/nature23474

Bishop CM (2011) Pattern recognition and machine learning, 1st ed. 2006. corr. 2nd printing 2011 edition edn. Springer, New York

Botvinick M, Wang JX, Dabney W, Miller KJ, Kurth-Nelson Z (2020) Deep reinforcement learning and its neuroscientific implications. Neuron 107(4):603–616. https://doi.org/10.1016/j.neuron.2020.06.014

Breuer HP, Petruccione F (2002) The theory of open quantum systems. Oxford University Press

Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, Zaremba W (2016) Openai gym. arXiv:1606.01540

Caruso F (2014) Universally optimal noisy quantum walks on complex networks. New J Phys 16(5):055015. https://doi.org/10.1088/1367-2630/16/5/055015

Caruso F, Chin AW, Datta A, Huelga SF, Plenio MB (2009) Highly efficient energy excitation transfer in light-harvesting complexes: the fundamental role of noise-assisted transport. J Chem Phys 131(10):09–612

Caruso F, Giovannetti V, Lupo C, Mancini S (2014) Quantum channels and memory effects. Rev Mod Phys 86(4):1203

Caruso F, Crespi A, Ciriolo AG, Sciarrino F, Osellame R (2016) Fast escape of a quantum walker from an integrated photonic maze. Nat Commun 7:11682. https://doi.org/10.1038/ncomms11682

Cover TM, Thomas JA (1991) Information theory and statistics. Wiley series in telecommunications, Wiley, New York

Dalla Pozza N, Caruso F (2020) Quantum state discrimination on reconfigurable noise-robust quantum networks. Phys Rev Res 2:043011. https://doi.org/10.1103/PhysRevResearch.2.043011

Dong DCZ, Chen C (2005) Quantum reinforcement learning. Notes Comput Sci 3611:686–689

Dunjko V, Taylor JM, Briegel HJ (2016) Quantum-enhanced machine learning. Phys Rev Lett 117(13):130501. https://doi.org/10.1103/physrevlett.117.130501

Ghavamzadeh M, Mannor S, Pineau J, Tamar A (2015) Bayesian reinforcement learning: a survey. Found Trends Mach Learn 8(5–6):359–483. https://doi.org/10.1561/2200000049

Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: data mining, inference, and prediction. Springer

Hu L, Wu SH, Cai W, Ma Y, Mu X, Xu Y, Wang H, Song H, Song Y, Deng DL, Zou CL et al (2019) Quantum generative adversarial learning in a superconducting quantum circuit. Sci Adv 5(1):2761

Johansson JR, Nation PD, Nori F (2012) QuTiP: an open-source Python framework for the dynamics of open quantum systems. Comput Phys Commun 183(8):1760–1772

Kiran BR, Sobh I, Talpaert V, Mannion P, Sallab AAA, Yogamani S, Pérez P (2020) Deep reinforcement learning for autonomous driving: a survey. https://doi.org/10.1109/TITS.2021.3054625

Liaw R, Liang E, Nishihara R, Moritz P, Gonzalez JE, Stoica I (2018) Tune: a research platform for distributed model selection and training. arXiv:1807.05118

Lindblad G (1976) On the generators of quantum dynamical semigroups. Commun Math Phys 48(2):119–130

Lloyd S, Schuld M, Ijaz A, Izaac J, Killoran N (2020) Quantum embeddings for machine learning. arXiv:2001.03622

Martina S, Buffoni L, Gherardini S, Caruso F (2022) Learning the noise fingerprint of quantum devices. Quantum Mach Intell 4:8. https://doi.org/10.1007/s42484-022-00066-0

Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G et al (2015) Human-level control through deep reinforcement learning. Nat 518(7540):529–533

Mott A, Job J, Vlimant JR, Lidar D, Spiropulu M (2017) Solving a higgs optimization problem with quantum annealing for machine learning. Nat 550(7676):375

Neven H, Denchev VS, Rose G, Macready WG (2008) Training a binary classifier with the quantum adiabatic algorithm. arXiv:0811.0416

Otterbach J, Manenti R, Alidoust N, Bestwick A, Block M, Bloom B, Caldwell S, Didier N, Fried ES, Hong S et al (2017) Unsupervised machine learning on a hybrid quantum computer. arXiv:1712.05771

Paparo GD, Dunjko V, Makmal A, Martin-Delgado MA, Briegel HJ (2014) Quantum speedup for active learning agents. Phys. Rev. X 4:031002. https://doi.org/10.1103/PhysRevX.4.031002

Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S (2019) PyTorch: an imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A, d' Alché-Buc F, Fox E, Garnett R (eds.) Advances in Neural Information Processing Systems. Curran Associates Inc. 32:8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Saggio V, Asenbeck BE, Hamann A, Strömberg T, Schiansky P, Dunjko V, Friis N, Harris NC, Hochberg M, Englund D et al (2021) Experimental quantum speed-up in reinforcement learning agents. Nat 591(7849):229–233

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

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D (2016) Mastering the game of Go with deep neural networks and tree search. Nat 529(7587):484–489. https://doi.org/10.1038/nature16961

Stooke A, Abbeel P (2019) rlpyt: a research code base for deep reinforcement learning in PyTorch. arXiv:1909.01500>

Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT press

Whitfield JD, Rodríguez-Rosario CA, Aspuru-Guzik A (2010) Quantum stochastic walks: a generalization of classical random walks and quantum walks. Phys. Rev. A 81:022323. https://doi.org/10.1103/PhysRevA.81.022323

Winci W, Buffoni L, Sadeghi H, Khoshaman A, Andriyash E, Amin MH (2020) A path towards quantum advantage in training deep generative models with quantum annealers. Mach Learn Sci Technol 1(4):045028. https://doi.org/10.1088/2632-2153/aba220

Wittek P (2014) Quantum machine learning: what quantum computing means to data mining. Academic Press