Offline Quantum Reinforcement Learning in a Conservative Manner

Zhihao Cheng[1], **Kaining Zhang**[1], **Li Shen**[2], **Dacheng Tao**[2, 1]

[1] The University of Sydney, Australia
[2] JD Explore Academy, China
zche3121@uni.sydney.edu.au, kzha3670@uni.sydney.edu.au, mathshenli@gmail.com, dacheng.tao@gmail.com

Abstract

Recently, to reap the quantum advantage, empowering reinforcement learning (RL) with quantum computing has attracted much attention, which is dubbed as quantum RL (QRL). However, current QRL algorithms employ an online learning scheme, *i.e.*, the policy that is run on a quantum computer needs to interact with the environment to collect experiences, which could be expensive and dangerous for practical applications. In this paper, we aim to solve this problem in an offline learning manner. To be more specific, we develop the first offline quantum RL (offline QRL) algorithm named CQ2L (Conservative Quantum Q-learning), which learns from offline samples and does not require any interaction with the environment. CQ2L utilizes variational quantum circuits (VQCs), which are improved with data re-uploading and scaling parameters, to represent Q-value functions of agents. To suppress the overestimation of Q-values resulting from offline data, we first employ a double Q-learning framework to reduce the overestimation bias; then a penalty term that encourages generating conservative Q-values is designed. We conduct abundant experiments to demonstrate that the proposed method CQ2L can successfully solve offline QRL tasks that the online counterpart could not.

Introduction

Reinforcement learning (RL), a machine learning paradigm that learns by trial and error, has recently made rapid progress in addressing complicated decision-making problems such as games (Mnih et al. 2015; Ye et al. 2019), robotics (Andrychowicz et al. 2020b), animation (Peng et al. 2018), etc. RL algorithms are emerging and can be further classified based on different standards. For example, according to whether a model is used during learning, RL can be divided into model-free RL and model-based RL (Moerland, Broekens, and Jonker 2020); according to how samples for training are obtained, RL could be categorized into, on-policy RL (Andrychowicz et al. 2020a), off-policy RL (Fujimoto, Meger, and Precup 2019), and offline RL (Levine et al. 2020). On-policy and off-policy RL are online, meaning that they require interactions with the environment for learning. In contrast, offline RL learns from a given dataset collected by another agent and thus does not demand online interactions. Nevertheless, despite the success of RL, training an RL algorithm from scratch suffers from high sample complexity (Yu 2018; Li et al. 2021). Researchers have proposed various methods to reduce the sample complexity of RL (Nachum et al. 2018; Seitzer, Schölkopf, and Martius 2021), among which quantum technologies are of great interest (Saggio et al. 2021).

Empowered by superposition and entanglement, the quantum system can represent and process an exponentially larger number of states with one operation compared to that in classical computing (Biamonte et al. 2017), leading to a possible exponential speedup for specific tasks (Gyongyosi and Imre 2019; Zhong et al. 2020; Arute et al. 2019). Machine learning (ML), a research domain that requests huge computing resources (Dean, Patterson, and Young 2018), has benefited from quantum computing to speed up ML algorithms (Biamonte et al. 2017; Nguyen et al. 2020). Among ML, enhancing RL with quantum technologies has attracted extensive attention, opening a new research field dubbed as quantum reinforcement learning (QRL) (Dong et al. 2008). Theoretical analyses have proved that QRL based on the Grover search can obtain a quadratic improvement compared to the classical RL regarding learning efficiency (Biamonte et al. 2017; Li et al. 2020; Saggio et al. 2021). However, the QRL methods that utilize Grover search are hard to be implemented in the noisy intermediate-scale quantum (NISQ) era (Hsiao et al. 2022). A promising solution is to build QRL with variational quantum circuits (VQCs), which are suitable for NISQ devices. Researchers have developed various QRL algorithms by employing VQCs, *e.g.*, on-policy methods (Jerbi et al. 2021) and off-policy approaches (Lockwood and Si 2020; Skolik, Jerbi, and Dunjko 2022; Franz et al. 2022).

However, current QRL algorithms rely on an online learning paradigm (Lockwood and Si 2020; Jerbi et al. 2021; Skolik, Jerbi, and Dunjko 2022), which means that they need to actively interact with environments to obtain new trials. Although online QRL has witnessed some successful applications, this online learning scheme imposes an obstacle to its applicability to complex real-world problems (Levine et al. 2020). There are three major reasons that can impair the applicability of online QRL. First, it is infeasible to online sample data for many cases because this could be expensive or dangerous (Wang et al. 2020). For example, in robotics, a sequence of improper actions during learning could irre-

vocably damage the robot. Second, effectively collecting large and diverse datasets for online QRL could be difficult in large-scale environments. In addition, the online learning framework generally adopts a hybrid quantum-classical structure (Jerbi et al. 2021), *i.e.*, some components of the online QRL algorithm are run on quantum computers while the remaining is implemented on classical computers. The hybrid quantum-classical structure functions at the expense of communication cost and simulation time (Lin, Wei, and Yao 2021; Liang et al. 2018), which could slow down the training process.

In this work, we propose offline quantum reinforcement learning (offline QRL) to bypass the drawbacks of online QRL. We employ variational quantum circuits (VQCs) to represent Q-value functions and enhance the expressivity with data re-uploading. To accommodate the property that the range of Q-value functions could be huge, a trainable scaling parameter is attached after the readout of VQCs. Considering the problem that Q-values are prone to overestimation with offline data, we utilize two different VQCs to alternatively update the Q-value target, which is referred to as double Q-learning and can reduce the maximization bias; then we design a penalty term to encourage learning conservative Q-values on out-of-distribution (OOD) data. Finally, we conduct experiments and demonstrate that our algorithm Conservative Quantum Q-learning (CQ2L) can satisfactorily solve the classical control tasks CartPole-v0, Acrobot-v1, and MountainCar-v0 (Brockman et al. 2016) with offline data. We summarize our contributions as follows:

- To our knowledge, we are the first to propose offline quantum reinforcement learning (offline QRL), which opens a new research topic in QRL.
- We conduct pilot studies and develop the first offline QRL algorithm CQ2L, which utilizes modified VQCs and learns in a conservative manner.
- Experiment results illustrate that CQ2L can successfully and effectively learn a control policy from offline data.

Related Work

Here, we briefly review reinforcement learning and quantum reinforcement learning.

Reinforcement Learning (RL) RL, one of the three domains of machine learning (ML), has attracted extensive interest due to its ability to solve complicated decision-making tasks with superhuman performance (Silver et al. 2018). RL can be further classified into various branches according to how it learns, *e.g.*, model-based (Kaiser et al. 2019) vs model-free (Schulman et al. 2017), policy-based (Schulman et al. 2015) vs value-based (Mnih et al. 2015), on-policy (Schulman et al. 2015) vs off-policy (Lillicrap et al. 2015), etc. A basic assumption underlying most RL algorithms is that agents can take exploratory actions and apply them to environments, where the online sampling scheme is referred to as online RL. However, online RL cannot utilize previously logged data like supervised learning (SL) and could be infeasible in safety-critical and cost-sensitive scenarios where agents are not allowed to make mistakes (Wang et al. 2020). To remove the need for online interactions, researchers propose offline RL (Levine et al. 2020; Wang et al. 2020; Fujimoto and Gu 2021; Kumar et al. 2020).

Quantum Reinforcement Learning (QRL) QRL is motivated by combining the merits of RL and quantum computing, which was first proposed by Dong et al. (2008). To speed up learning, Dong et al. (2008) represent states with quantum superposition and approximate value functions using Grover iteration (Grover 1996). Recently, researchers have employed variational quantum circuits (VQCs) to represent Q-value functions and propose variational-quantum deep Q-Networks (VQ-DQN) (Lockwood and Si 2020; Skolik, Jerbi, and Dunjko 2022; Franz et al. 2022). Lockwood and Si (2020) introduce two novel data encoding schemes for converting traditional data into quantum states and successfully solve the CartPole-v0 task. Skolik, Jerbi, and Dunjko (2022) design a new structure of VQCs with data re-uploading and trainable scaling and achieve similar performance to the conventional RL. Jerbi et al. (2021) utilize a similar VQC as in Skolik, Jerbi, and Dunjko (2022) to construct agent policies and train agents with the policy gradient algorithm REINFORCE. From the theoretical aspect, QRL has been proved to enjoy a quadratic speedup compared to classical RL (Biamonte et al. 2017; Dunjko, Taylor, and Briegel 2016; Wang et al. 2021). However, from the experimental aspect, there are some controversies over the speedup in QRL. Franz et al. (2022) conduct abundant experiments of VQ-DQN and argue that no measurable quantum advantage is observed, whereas the QRL agent in Hsiao et al. (2022) can achieve the same level of performance with much fewer parameters and samples compared against traditional RL that is built on multi-layer perceptrons (MLPs) (Kruse et al. 2022).

To the best of our knowledge, current QRL algorithms rely on interactions with environments to successfully learn a control policy. Nonetheless, the online learning paradigm can be prohibitive due to safety and cost concerns when applying current QRL algorithms to practical applications such as healthcare and robotics (Levine et al. 2020). In this paper, we aim to develop an offline QRL algorithm that can eliminate the demand for agent-environment interactions.

Backgrounds

Offline Reinforcement Learning (Offline RL)

Problem Statement of Online RL. Generic RL aims to control a dynamical system that is typically described by a Markov Decision Process (MDP) (Puterman 1994). An MDP is defined by a 5-tuple (S, A, T, r, γ), with the state space S ($s \in S$ denotes a state), the action space A ($a \in A$ represents an action), the probability distribution of environment transition dynamics $T = T(s'|s, a)$, the reward function $r : S \times A \to \mathbb{R}$, and a discount factor $\gamma \in [0, 1]$. Without loss of generality, we assume the reward function is bounded by $R_{\max}$, *i.e.*, $|r(s, a)| \leq R_{\max}$, $\forall (s, a) \in S \times A$. Under the online RL setting, agents can generate actions using a stochastic policy $\pi(a|s) : S \times A \to [0, 1]$ and apply sampled actions to the MDP. Trajectories $\tau = \{s_0, a_0, s_1, a_1, \cdots\}$ can be obtained by iteratively interacting with the MDP, where $s_0 \sim \rho_0(s_0)$, $\rho_0(\cdot)$ is the initial state distribution,

$a_t \sim \pi(a_t|s_t)$, $s_{t+1} \sim T(s_{t+1}|s_t, a_t)$, and t represents the timestep. RL aims to discover the optimal policy $\pi^*(a|s)$ by maximizing the expected discounted reward $J(\pi) = \mathbb{E}_{s_0,a_0,\cdots}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right]$ (Sutton, Barto et al. 1998).

In contrast to online RL, agents are not allowed to interact with environments to obtain new samples under the offline setting, making it challenging to estimate $J(\pi)$ and optimize policies consequently. Instead, only a static dataset $\mathcal{D} = \{(s_t^i, a_t^i, r_t^i, s_{t+1}^i)\}$ is provided to agents for learning in offline RL. The dataset $\mathcal{D}$ is composed of transition tuples generated with an unknown behavior policy π_β. We formally state the offline RL problem as follows.

Problem Statement of Offline RL. Provided with a fixed dataset $\mathcal{D}$, the goal of offline RL is to generate a near-optimal policy without further interactions with environments (Levine et al. 2020).

In addition to $J(\pi)$, the value function $V^\pi(s)$ and Q-value function $Q^\pi(s,a)$ are also commonly-used performance measures, whose definitions are:

$$V^\pi(s) := \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)|s_0 = s\right], \quad (1)$$

$$Q^\pi(s,a) := \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)|s_0 = s, a_0 = a\right]. \quad (2)$$

Basics of Quantum Computing

Qubits and Quantum Gates In quantum computing, the basic element is defined as a *qubit* (De Wolf 2019). A qubit $|\psi\rangle$ is a linear combination of the basis states $|0\rangle$ and $|1\rangle$, *i.e.*, $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $|0\rangle = (1,0)^\top$, $|1\rangle = (0,1)^\top$, $\alpha, \beta \in \mathbb{C}$, and $|\alpha|^2 + |\beta|^2 = 1$. The phenomenon that $|\psi\rangle$ could be $|0\rangle$ and $|1\rangle$ at the same time is termed quantum superposition. Similar to $|\psi\rangle$, we define an n-qubit quantum state as follows

$$\begin{aligned} |x\rangle =& x_1|\underbrace{0\cdots0}_{n}\rangle + x_2|0\cdots1\rangle + ... + x_{2^n}|1\cdots1\rangle \\ & s.t.\ \sum_{i=1}^{2^n}|x_i|^2 = 1 \text{ and } x_i \in \mathbb{C}. \end{aligned} \quad (3)$$

From Eq. (3), we see that the dimension of the space spanned by an n-qubit system is $M = 2^n$. We can use the tensor product $\otimes$ to generate higher-dimensional quantum systems $|x\rangle_{m+n} = |x\rangle_m \otimes |x\rangle_n$ with two subsystems $|x\rangle_m$ and $|x\rangle_n$, whose space dimensions are $m+n$, m, and n, respectively. In quantum mechanics, the other fundamental property is entanglement (Lockwood and Si 2020). We say a 2-qubit quantum state $|x\rangle$ is entangled if $|x\rangle$ cannot be obtained by the tensor product $|x_1\rangle \otimes |x_2\rangle$, where $|x_1\rangle$ and $|x_2\rangle$ are single-qubit states. Every qubit is independent of the others in an un-entangled multi-qubit state $|x\rangle$. By contrast, for an entangled quantum state, each qubit is perfectly correlated to the remaining, providing us a way to simultaneously modify multiple qubits in an operation.

In quantum computing, quantum gates are employed to perform logic operations on qubits (DiVincenzo 1998). For example, commonly-used single-qubit gates are the Hadamard gate, rotation gates, Pauli gates, and phase shift gates. In terms of the 2-qubit system, examples are CNOT and CZ gates. The effect of an n-qubit quantum gate is mathematically equal to a unitary matrix whose dimension is $2^n \times 2^n$. For instance, the unitary matrix of the Hadamard gate can be specified $H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$. To perform complex unitary operations, quantum circuits can be built by stacking elementary gates sequentially or in parallel. For more details on quantum background, please refer to De Wolf (2019).

Variational Quantum Circuits (VQCs) As explained above, a quantum circuit can perform a specific and fixed unitary operation after construction. To accommodate flexibility and learning ability, quantum circuits are equipped with trainable parameters, which are referred to as VQCs (Benedetti et al. 2019). Denote the parameter as μ, then a VQC can be specified as

$$f_\mu(x) = \langle 0^{\otimes n}| U^\dagger(x,\mu) O U(x,\mu) |0^{\otimes n}\rangle, \quad (4)$$

where x is the input, $U(x,\mu)$ represents a unitary operator which transforms input x with the learnable parameter μ, $U^\dagger(x,\mu)$ is the inverse of $U(x,\mu)$, and O stands for an observable for quantum measurement. Hence, VQCs, which operate on quantum systems, can be used to approximate functions by inputting and outputting classical data x and $y = f_\mu(x)$. Given the classical input x and output y, specific loss functions can be built according to tasks. Consequently, the parameter μ in VQCs can be updated with gradients obtained by utilizing the chain rule and parameter-shift rule (Benedetti et al. 2019). Besides, VQCs have the capability of serving as a universal approximator (Biamonte 2021), enabling us to solve complex tasks with VQCs.

Methodology

In this section, we provide a detailed explanation of the developed offline QRL algorithm, CQ2L. We begin with representing a Q-value function with VQCs, which are improved with data re-uploading and scaling parameters. Then, we discuss and validate the current problems of directly extending the existing off-policy QRL algorithm VQ-DQN to the offline setting. Finally, we present CQ2L with the desire to learn conservative Q-values using the double Q-learning framework and a penalty term.

Representing Q-value Functions with VQCs

Here, we aim to build a Q-value function $Q_\theta(s,a)$ using VQCs. The Q-value function $Q_\theta(s,a)$ maps the index (s,a) to a scalar, which can be subsequently used to choose actions. In an MDP, both the state s and action a are classical data, which cannot be directly processed by quantum computers. Hence, we first need to convert the classical data in the dataset $\mathcal{D}$ into quantum states. We choose to use the qubit encoding (Schuld and Killoran 2019), being efficient for the continuous space. Given a state $s = \{s^{(1)}, \ldots, s^{(n)}\} \in \mathbb{R}^n$, we denote its item as $s^{(i)} (0 \le i \le n)$. Then $s^{(i)}$ is first multiplied with the input scaling λ_i and then scaled to the range $[-\pi/2, \pi/2]$ by $\bar{s}^{(i)} = \arctan(\lambda_i \times s^{(i)})$. The input scaling parameter $\lambda = \{\lambda_1, \ldots, \lambda_n\}$ helps improve the capacity of VQCs and is trainable (Jerbi et al. 2021; Skolik, Jerbi, and Dunjko 2022). Consequently, the state s is converted into a quantum state

below

$$|\Psi(s)\rangle = \begin{pmatrix} \cos \bar{s}^{(1)} \\ \sin \bar{s}^{(1)} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \cos \bar{s}^{(n)} \\ \sin \bar{s}^{(n)} \end{pmatrix}. \quad (5)$$

After encoding, the information of s is uploaded and represented by rotations of qubits with the original information lost forever. A VQC that uploads the information of s once can just approximate simple functions like sine (Skolik, Jerbi, and Dunjko 2022) and is not enough to solve complex decision-making tasks. Therefore, we employ a data re-uploading structure for VQCs (Skolik, Jerbi, and Dunjko 2022), which uploads s to VQCs several times and stacks them sequentially with the variational part. The parameter of variational blocks is denoted as ϕ. Moreover, VQCs with the data re-uploading structure are able to achieve universal approximation for complex functions (Pérez-Salinas et al. 2020).

After uploading the state s and processing the information on quantum systems, we need to decode the Q-value from quantum states to classical ones with quantum measurement (De Wolf 2019). Using a quantum observable O_a, the Q-value function is represented as follows

$$Q_\theta(s,a) = \langle 0^{\otimes n}| U^\dagger(s,\mu) O_a U(s,\mu) |0^{\otimes n}\rangle. \quad (6)$$

The output range of Eq. (6) is bounded by the spectral norm of the quantum observable as $[-\|O_a\|, \|O_a\|]$. To cover the range of $Q(s,a) \in [-\frac{R_{\max}}{1-\gamma}, \frac{R_{\max}}{1-\gamma}]$, we further incorporate a trainable parameter ν_a after the measurement

$$Q_\theta(s,a) = \langle 0^{\otimes n}| U^\dagger(s,\mu) O_a U(s,\mu) |0^{\otimes n}\rangle \times \nu_a. \quad (7)$$

The parameter θ of our Q-value function $Q(s,a)$ contains three portions of parameters, *i.e.*, the input scaling λ, the variational part ϕ, and the output scaling $\nu = \{\nu_a | a \in A\}$.

Current Problems

A minimalist way to enjoy the merits of offline QRL is by promptly extending current off-policy QRL algorithms to the offline setting. To be more specific, the exploratory data to update Q-values in off-policy RL, which are obtained by interacting with the environment, are replaced with offline data. Some researchers have demonstrated that off-policy RL algorithms could perform satisfactorily with carefully-collected offline data in some cases (Agarwal, Schuurmans, and Norouzi 2020; Yarats et al. 2022). However, when it comes to real-world applications, we have no way to control the sampling process of offline data but only the learning

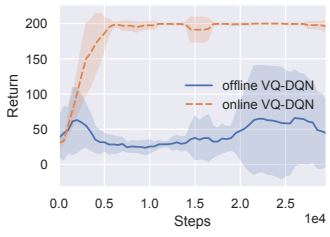


Figure 1: Performance of online and offline VQ-DQN.

Algorithm 1: Conservative Quantum Q-learning (CQ2L)

Input: Offline data $\mathcal{D}$, iteration number m, learning rate ξ_θ, and minibatch size N.
Parameter: Q-value functions with parameters θ^A and θ^B, where $\theta = (\lambda, \phi, \nu)$.
for $i = 1$ **to** m **do**
  Sample N transitions from offline dataset $\mathcal{D}$: $\{(s,a,r,s',d)\}_1^N \sim \mathcal{D}$
  Alternatively update parameters θ^A and θ^B using the loss function in Eq. (10)
end for

algorithm with provided data. Hence, to investigate whether it is feasible to extend current off-policy QRL algorithms to the offline setting, we conduct experiments using the off-policy QRL algorithm VQ-DQN in Skolik, Jerbi, and Dunjko (2022); Franz et al. (2022). On the CartPole-v0 task, two categories of experiments are carried out, *i.e.*, online VQ-DQN and offline VQ-DQN. The prefixes, online and offline, distinguish the training data for VQ-DQN. Online VQ-DQN interacts with the environment to sample exploratory off-policy data and updates parameters of its VQCs utilizing self-sampled data, whereas offline VQ-DQN merely uses offline data similar to Seno and Imai (2021) and does not require interactions. More details on offline data and hyperparameters are presented in the section of Experiments. To fairly and accurately measure the performance of online VQ-DQN and offline VQ-DQN during training, we greedily select actions in the environment according to their quantum Q-values and estimate the episode return with 10 episodes. Note that, the sampled data during testing are never used for training, and it is a common compromise to evaluate algorithms in offline RL (Fujimoto and Gu 2021).

The learning curves of online and offline VQ-DQN are presented in Figure 1. As we can see, online VQ-DQN can satisfactorily solve the CartPole-v0 task, which obtains a cumulative reward near 200 at the end of training. On the contrary, offline VQ-DQN has a relatively poor performance, achieving only about a quarter of the outcome of the online counterpart. We also conduct a hyperparameter search to find whether the failed learning of offline VQ-DQN is caused by unmatched hyperparameters. In particular, we adjust the target update interval, learning rates of VQCs, and batch size. Further experiments consistently confirm that VQ-DQN could not generate a successful control policy with offline data even if CartPole-v0 is one of the simplest tasks in OpenAI Gym (Brockman et al. 2016). The poor performance of offline VQ-DQN motivates us to develop a pure offline QRL algorithm, which can effectively learn from offline data.

Conservative Quantum Q-learning (CQ2L)

A major challenge in offline QRL is that it breaks the fundamental assumption that agents can sample data online. In other words, agents are supposed to learn a policy or value function from out-of-distribution (OOD) data, which is non-trivial. The distribution shift makes it hard to evaluate and consequently improve current Q-value functions, leading to

the extrapolation error (Kostrikov et al. 2021). Under the online setting, agents can actively inquiry environments for more data to correct the extrapolation error. However, for offline training, the extrapolation error means that agents could over-optimistically estimate Q-values for unseen state-action pairs, and optimizing policies according to the over-optimistic Q-value functions could lead to poor performance. Hence, we propose to suppress the over-optimistic problem in offline QRL by learning a conservative Q-value function. In particular, we improve the update process of VQ-DQN with a double Q-learning framework and a penalty term to reserve conservativeness in the update of Q-values.

We rewrite the update rule of offline VQ-DQN (Skolik, Jerbi, and Dunjko 2022) as follows

$$Q\left(s,a;\theta_{k+1}\right)\leftarrow$$
$$\underset{Q}{\operatorname{argmin}} \underset{(s,a,r,s')\sim\mathcal{D}}{\mathbb{E}} \left(r+\gamma\cdot\max_{\bar{a}\in A} Q(s',\bar{a};\theta_k)-Q(s,a)\right)^2,$$

where $Q(s,a;\theta_{k+1})$ and $Q(s,a;\theta_k)$ are Q-value networks that are constructed with VQCs. The target in VQ-DQN is defined as $Y_k^Q := r+\gamma\cdot\max_{\bar{a}\in A} Q(s',\bar{a};\theta_k)$. Updating $Q(s,a;\theta_k)$ using its own estimated target Y_k^Q could lead to training instability. To stabilize the learning process, a target network $Q(s,a;\theta_k^-)$ is introduced, whose parameter θ_k^- is updated to the current θ_k every fixed timestep (Mnih et al. 2015; Skolik, Jerbi, and Dunjko 2022). Hence, the update rule is modified as follows

$$Q\left(s,a;\theta_{k+1}\right)\leftarrow \underset{Q}{\operatorname{argmin}} \underset{(s,a,r,s')\sim\mathcal{D}}{\mathbb{E}} \left(Y_k^{\mathrm{DQN}}-Q(s,a)\right)^2,$$

where $Y_k^{\mathrm{DQN}} := r+\gamma\cdot\max_{\bar{a}\in A} Q(s',\bar{a};\theta_k^-)$. Although Y_k^{DQN} is calculated based on a target Q-value $Q(s,a;\theta_k^-)$, the parameter θ_k^- is still closely related to θ_k and could be unstable. Taking a max operation with estimated Q-values that are not accurate could lead to overestimation. What's worse, in offline RL, agents cannot online sample data and thus lose the chance to correct estimations, which further enlarges the overestimation and results in control failures.

To this end, we develop CQ2L in order to suppress the overestimation in offline learning, which is presented in Algorithm 1. CQ2L achieves a double overestimation reduction with a double Q-learning framework and a conservative Q-value update.

Double Q-learning Instead of using a target network $Q(s,a;\theta_k^-)$, we define two independent sets of parameters θ_k^A and θ_k^B. Then the target Y_k^{DoubleQ} for updating the parameter θ_k^A is specified as

$$Y_k^{\mathrm{DoubleQ}} := r+\gamma\cdot Q(s',\operatorname*{argmax}_{\bar{a}\in A} Q(s',\bar{a};\theta_k^A);\theta_k^B). \quad (8)$$

From Eq. (8), the action $\bar{a}$ which can obtain the maximum Q-value is still selected based on $Q(s',\bar{a};\theta_k^A)$. However, the Q-value in the target Y_k^{DoubleQ} is not calculated based on θ_k^A but θ_k^B. The independent parameter θ_k^B outputs an unbiased Q-value estimation for updating $Q(s,a;\theta_k^A)$ (Van Hasselt, Guez, and Silver 2016). The other parameter θ_k^B is adjusted with Eq. (8) by symmetrically exchanging the roles of θ_k^A and θ_k^B. The double Q-learning update framework reduces the maximization error and thus lowers the overestimation error.

Conservative Q-value Update With the double Q-learning framework, $Q(s,a;\theta_k^A)$ can be offline updated according to

$$Q\left(s,a;\theta_{k+1}^A\right)\leftarrow \underset{Q}{\operatorname{argmin}} \underset{(s,a,r,s')\sim\mathcal{D}}{\mathbb{E}} \left(Y_k^{\mathrm{DoubleQ}}-Q(s,a)\right)^2.$$

Note that samples (s,a,r,s') for updating Q-value functions are generated from a provided dataset $\mathcal{D}$. And data in the dataset $\mathcal{D}$ are collected with an unknown behavior policy $\pi_\beta(a|s)$. This is substantially different from off-policy QRL, which samples data using an epsilon-greedy policy based on its Q-values. With the maximization operation in Y_k^{DoubleQ}, Q-values are prone to be overestimated. Then agents could be biased to select actions that are overestimated but not executed by $\pi_\beta(a|s)$, which brings distribution shift and could further damage the RL performance. To suppress the overestimated error and distribution shift, we add a penalty term to form a conservative update target

$$\begin{aligned} &Q\left(s,a;\theta_{k+1}^A\right)\leftarrow \\ &\underset{Q}{\operatorname{argmin}}\,\alpha\cdot\Big(\underset{s\sim\mathcal{D},a\sim\mu(a|s)}{\mathbb{E}} Q(s,a;\theta_k^A)-\underset{(s,a)\sim\mathcal{D}}{\mathbb{E}} Q(s,a;\theta_k^A)\Big) \\ &+\underset{(s,a,r,s')\sim\mathcal{D}}{\mathbb{E}} \left(Y_k^{\mathrm{DoubleQ}}-Q(s,a)\right)^2, \end{aligned} \quad (9)$$

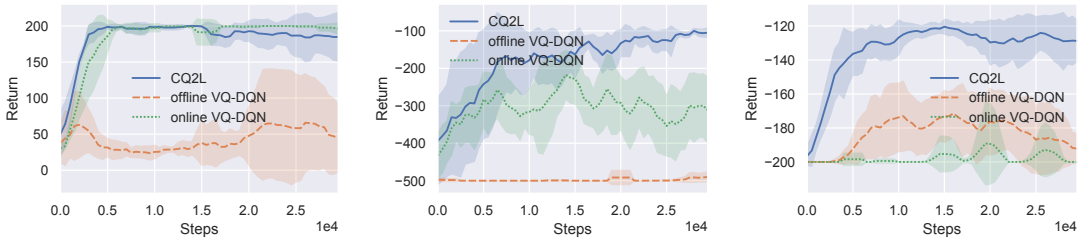
where $\mu(a|s)$ is another policy, and α is a scalar. The term $\mathbb{E}_{s\sim\mathcal{D},a\sim\mu(a|s)}Q(s,a;\theta_k^A)$ penalizes the Q-values to be conservative on out-of-distribution (OOD) state-actions pairs, whereas $-\mathbb{E}_{(s,a)\sim\mathcal{D}}Q(s,a;\theta_k^A)$ encourages to improve Q-values on data in the dataset $\mathcal{D}$. When the policy $\mu(a|s)$ exactly equals the agent policy π, the update in Eq. (9) lower bounds the true value function of π (Kumar et al. 2020). Since there are no explicit policies in value-based RL, we manually construct a Softmax policy based on $Q(s,a;\theta_k^A)$ such that $\mu(a|s)\propto\frac{1}{Z}\exp(Q(s,a;\theta_k^A))$, where Z represents the normalizing factor. Hence, the update target in Eq. (9) can be simplified as follows

$$\begin{aligned} &Q\left(s,a;\theta_{k+1}^A\right)\leftarrow \\ &\underset{Q}{\operatorname{argmin}}\,\alpha\underset{s\sim\mathcal{D}}{\mathbb{E}}\left(\log\sum_{a\in A}\exp(Q(s,a;\theta_k^A))-\underset{a\sim\pi_\beta}{\mathbb{E}} Q(s,a;\theta_k^A)\right) \\ &+\underset{(s,a,r,s')\sim\mathcal{D}}{\mathbb{E}} \left(Y_k^{\mathrm{DoubleQ}}-Q(s,a)\right)^2, \end{aligned} \quad (10)$$

which is the core update rule of CQ2L.

Experiments

In this section, we start with a description of experiment settings including the OpenAI Gym environment (Brockman et al. 2016), offline data, and implementation details. Next, we present the results of CQ2L and comparisons against the other baselines.



(a) CartPole-v0 (b) Acrobot-v1 (c) MountainCar-v0

Figure 2: Learning curves of CQ2L and the other baselines. We evaluate each algorithm with 5 random seeds with x-axis denoting iteration steps and y-axis representing episode return.

Experiment Settings

Environments Based on Skolik, Jerbi, and Dunjko (2022); Jerbi et al. (2021), we select three OpenAI classic control tasks CartPole-v0, Acrobot-v1, and MountainCar-v0. The states and actions of all environments are continuous and discrete, respectively. For example, in Acrobot-v1, the state is 6-dimensional which consists of sines, cosines, and angular velocities of the angles of two joints, while the action is a discrete number $0/1/2$. The action 0, 1, or 2 will correspondingly apply torques -1, 0, or $+1$ $N \cdot m$ to the actuated joint between the two links. The agent should swing up the robot links in the shortest possible time in Acrobot-v1. In contrast, under the CartPole-v0 task, agents are required to keep the pole upright as long as possible. Next, we create offline data for these three environments and conduct offline QRL.

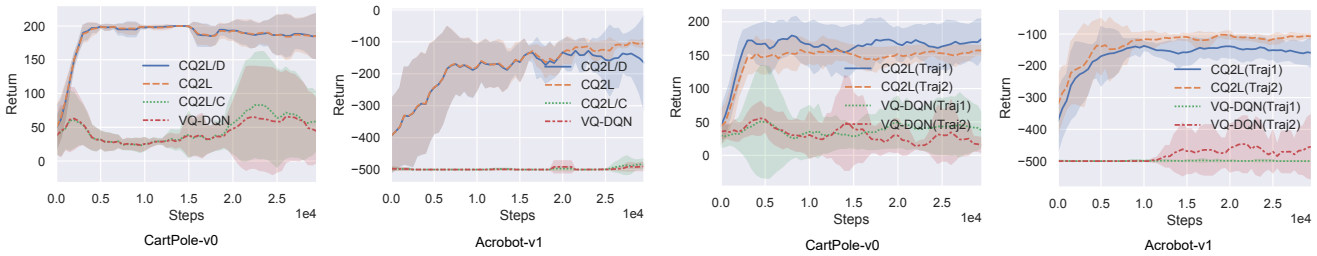
Offline Data In offline RL, the training data are provided by other agents rather than samples collected with their own policies. To evaluate the offline QRL algorithm, we create offline data for CartPole-v0, Acrobot-v1, and MountainCar-v0 in a similar way as d3rlpy (Seno and Imai 2021). In particular, we use a DQN agent to online sample data from the environment, whose learning rate is 0.001, target update interval is 100, and policy is constant epsilon-greedy with $\epsilon = 0.3$. The sampled data are recorded in a replay buffer with length 1×10^6 and then saved for offline QRL. The logged data contain tuples of $(s_t, a_t, r_t, s_{t+1}, d)$, and d indicates whether an episode ends. We refer readers to Seno and Imai (2021) and their codes for more details.

Implementation Details We implement the CQ2L algorithm according to Skolik, Jerbi, and Dunjko (2022); Jerbi et al. (2021); Seno and Imai (2021), in which Tensorflow Quantum (Broughton et al. 2020) and Cirq (Hancock et al. 2019) are used to simulate quantum states. The quantum Q-value functions are mainly adopted from Skolik, Jerbi, and Dunjko (2022); Jerbi et al. (2021), while the double Q-learning framework and conservative update are inspired by Seno and Imai (2021). In contrast to Skolik, Jerbi, and Dunjko (2022); Jerbi et al. (2021) which train policies once every fixed episodes, we sample a mini-batch of data and carry out gradient descent once with fixed steps. The reason is that the episode length of tested environments varies widely, leading to unfairness when evaluating algorithms with fixed episodes. For example, for CartPole-v0, the shortest episode length is about 10, whereas the longest reaches 200. Besides, we cannot define an episode in offline QRL since the agent does not interact with the environment. Due to the training scheme switching from fixed episodes to fixed steps, the parameters of VQ-DQN are slightly modified to fairly compare the online version against the offline one.

In experiments, we use VQCs with 5 layers to represent Q-value functions. There are 4, 6, and 2 qubits of VQCs for CartPole-v0, Acrobot-v1, and MountainCar-v0, respectively. According to the feasible actions in CartPole-v0, Acrobot-v1, and MountainCar-v0, we choose quantum observables $[Z_0Z_1, Z_2Z_3]$, $[Z_0, Z_1, Z_2]$, and $[Z_0, Z_0 * Z_1, Z_1]$, where Z_i is the readout of a Pauli Z gate on the ith qubit. We introduce the ansatz of employed VQCs. Input data are encoded with X rotation gates, while the variational part includes X, Y, and Z rotation gates. Note that the $\arctan$ operation is not enabled during encoding since the input data are relatively small. Qubits are entangled in a circular topology. The variational part, entanglement, and data encoding are repeated several times, which is then measured by Pauli Z gates to denote Q-values. The learning rates for VQCs' parameters $\xi_\theta = [\xi_\lambda, \xi_\phi, \xi_\nu]$ are 0.001, 0.001, and 0.1, respectively. The target Q-value Y_k^{DoubleQ} is calculated with the discount factor $\gamma = 0.99$ and then forwarded into a Huber loss (Akkaya and Pınar 2020). For every iteration, we sample data from $\mathcal{D}$ with a batch size of 16 and update $Q(s, a)$ utilizing an Adam optimizer (Kingma and Ba 2014). For the online VQ-DQN, we use similar parameters despite that VQ-DQN uses a buffer to store interactions and updates $Q(s, a)$ every 10 interactions. Since all tasks are not so complex, we do not use the entire logged data $\mathcal{D}$ that is huge in amount but just select one trajectory to train CQ2L and offline VQ-DQN.

Results

The learning curves of CQ2L, offline VQ-DQN, and online VQ-DQN are presented in Figure 2. It is clear that our algorithm CQ2L is able to solve the classical tasks and can significantly outperform offline VQ-DQN. For example, in CartPole-v0, CQ2L stabilizes at an episode return of 184 at the end of the training, whereas offline VQ-DQN achieves about 40. The poor performance of VQ-DQN with offline data demonstrates that it is not feasible to directly extend off-policy QRL algorithms to the offline setting. In terms of the online version, CQ2L performs marginally worse than



(a) Test on modifications (b) Various offline data

Figure 3: Ablation study of CQ2L.

online VQ-DQN in CartPole-v0 but noticeably surpasses it in Acrobot-v1 and MountainCar-v0. Besides, under Acrobot-v1 which is harder than CartPole-v0, some oscillations are observed during the learning of online VQ-DQN. Online VQ-DQN can solve CartPole-v0 but fails to solve Acrobot-v1 and MountainCar-v0 as we first validated. By contrast, CQ2L could use existing large and diverse datasets, leading to better Q-values to resolve the two tasks.

Ablations

In this part, we conduct ablation experiments to give more insights into why CQ2L works. First, we would investigate the role of double Q-learning and the conservative update in CQ2L. In particular, we disable these two modifications and measure the performance of agents. Second, we also carry out experiments to test the impact of offline data. The learning curves of ablation studies are shown in Figure 3.

From Figure 3(a), we can conclude that both the framework of double Q-learning and the conservative update play an important role in the success of CQ2L. We use the legend CQ2L/x to denote that the modification x is disabled, and x could be D or C, where D and C are the initials of double Q-learning and conservative update. For example, CQ2L/D means that the double Q-learning framework is disabled. We can see that in CartPole-v0, no noticeable performance improvement is observed by adding the double Q-learning framework. By contrast, in Acrobot-v1, CQ2L performs much better than CQ2L/D. Under both tasks, the conservative updating technique can significantly improve the performance of learning from offline data. Figure 3(b) presents the results of CQ2L and offline VQ-DQN with different trajectories. For instance, CQ2L(Traj1) means that the offline data used is the first trajectory. These learning curves demonstrate that CQ2L can consistently outperform offline VQ-DQN regardless of the offline data.

Comparison to Classical RL

The fundamental motivation for developing QRL is to enjoy the potential advantage of quantum computing over its classical counterpart. Here, we conduct experiments in CartPole-v0 to investigate whether some advantages (including sample efficiency (Hsiao et al. 2022) and the reduction in parameters (Chen et al. 2020)) of QRL could be observed. We mainly compare CQ2L with the classical CQL, whose policy is represented by MLPs. To conduct a fair comparison, we

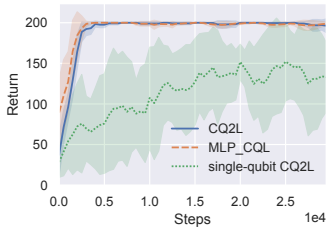


Figure 4: Performance of QRL vs classical RL.

use a VQC with 3 layers (62 parameters in total) for CQ2L, whereas for classical CQL, an MLP using 2 hidden layers with 5 nodes in each layer is adopted (67 parameters in total). Thus, CQ2L and classical RL have a similar number of parameters. Besides, we also replace the VQCs in CQ2L with single-qubit VQCs adopted from Hsiao et al. (2022) and term it as single-qubit CQ2L. The experimental results are shown in Figure 4. From the figure, we notice that CQ2L could achieve comparable performance to the classical one with a similar amount of parameters. Besides, no significant advantages in the sample efficiency or the parameter size are observed, although these two advantages were claimed in Lockwood and Si (2020); Hsiao et al. (2022). It may indicate that the current structure of VQCs or the limited number of qubits is not sufficient to exhibit quantum advantages for QRL, and further effort is needed.

Conclusion

In this work, we concentrate on how to effectively conduct quantum reinforcement learning (QRL) with offline data. To the best of our knowledge, we develop the first offline QRL algorithm CQ2L, whose Q-value functions are represented by enhanced VQCs. To suppress the overestimation induced by offline data, CQ2L employs a double Q-learning framework and maintains a conservative Q-value update by imposing an additional penalty term. The proposed algorithm, CQ2L, can effectively learn from offline data and outperform its online counterparts. Besides, CQ2L demonstrates its ability on achieving similar performance to the MLP-based RL on basic control tasks. In the future, we would theoretically analyze whether CQ2L is able to utilize the quantum advantage and try to achieve quantum speedups in experiments.

Acknowledgements

Mr Zhihao Cheng and Mr Kaining Zhang are supported by ARC FL-170100117 and IC-190100031.

References

Agarwal, R.; Schuurmans, D.; and Norouzi, M. 2020. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, 104–114. PMLR.

Akkaya, D.; and Pınar, M. Ç. 2020. Minimizers of sparsity regularized Huber loss function. *Journal of Optimization Theory and Applications*, 187(1): 205–233.

Andrychowicz, M.; Raichuk, A.; Stańczyk, P.; Orsini, M.; Girgin, S.; Marinier, R.; Hussenot, L.; Geist, M.; Pietquin, O.; Michalski, M.; et al. 2020a. What matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv:2006.05990*.

Andrychowicz, O. M.; Baker, B.; Chociej, M.; Jozefowicz, R.; McGrew, B.; Pachocki, J.; Petron, A.; Plappert, M.; Powell, G.; Ray, A.; et al. 2020b. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1): 3–20.

Arute, F.; Arya, K.; Babbush, R.; Bacon, D.; Bardin, J. C.; Barends, R.; Biswas, R.; Boixo, S.; Brandao, F. G.; Buell, D. A.; et al. 2019. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779): 505–510.

Benedetti, M.; Lloyd, E.; Sack, S.; and Fiorentini, M. 2019. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology*, 4(4): 043001.

Biamonte, J. 2021. Universal variational quantum computation. *Physical Review A*, 103(3): L030401.

Biamonte, J.; Wittek, P.; Pancotti, N.; Rebentrost, P.; Wiebe, N.; and Lloyd, S. 2017. Quantum machine learning. *Nature*, 549(7671): 195–202.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. *arXiv preprint arXiv:1606.01540*.

Broughton, M.; Verdon, G.; McCourt, T.; Martinez, A. J.; Yoo, J. H.; Isakov, S. V.; Massey, P.; Halavati, R.; Niu, M. Y.; Zlokapa, A.; et al. 2020. Tensorflow quantum: A software framework for quantum machine learning. *arXiv preprint arXiv:2003.02989*.

Chen, S. Y.-C.; Yang, C.-H. H.; Qi, J.; Chen, P.-Y.; Ma, X.; and Goan, H.-S. 2020. Variational quantum circuits for deep reinforcement learning. *IEEE Access*, 8: 141007–141024.

De Wolf, R. 2019. Quantum computing: Lecture notes. *arXiv preprint arXiv:1907.09415*.

Dean, J.; Patterson, D.; and Young, C. 2018. A new golden age in computer architecture: Empowering the machine-learning revolution. *IEEE Micro*, 38(2): 21–29.

DiVincenzo, D. P. 1998. Quantum gates and circuits. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969): 261–276.

Dong, D.; Chen, C.; Li, H.; and Tarn, T.-J. 2008. Quantum reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(5): 1207–1220.

Dunjko, V.; Taylor, J. M.; and Briegel, H. J. 2016. Quantum-enhanced machine learning. *Physical review letters*, 117(13): 130501.

Franz, M.; Wolf, L.; Periyasamy, M.; Ufrecht, C.; Scherer, D. D.; Plinge, A.; Mutschler, C.; and Mauerer, W. 2022. Uncovering Instabilities in Variational-Quantum Deep Q-Networks. *arXiv preprint arXiv:2202.05195*.

Fujimoto, S.; and Gu, S. S. 2021. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2052–2062. PMLR.

Grover, L. K. 1996. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 212–219.

Gyongyosi, L.; and Imre, S. 2019. A survey on quantum computing technology. *Computer Science Review*, 31: 51–71.

Hancock, A.; Garcia, A.; Shedenhelm, J.; Cowen, J.; and Carey, C. 2019. Cirq: A Python Framework for Creating, Editing, and Invoking Quantum Circuits. https://github.com/quantumlib/Cirq. Accessed: 2022 March 24.

Hsiao, J.-Y.; Du, Y.; Chiang, W.-Y.; Hsieh, M.-H.; and Goan, H.-S. 2022. Unentangled quantum reinforcement learning agents in the OpenAI Gym. *arXiv preprint arXiv:2203.14348*.

Jerbi, S.; Gyurik, C.; Marshall, S.; Briegel, H.; and Dunjko, V. 2021. Parametrized quantum policies for reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 28362–28375.

Kaiser, L.; Babaeizadeh, M.; Milos, P.; Osinski, B.; Campbell, R. H.; Czechowski, K.; Erhan, D.; Finn, C.; Kozakowski, P.; Levine, S.; et al. 2019. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kostrikov, I.; Fergus, R.; Tompson, J.; and Nachum, O. 2021. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, 5774–5783. PMLR.

Kruse, R.; Mostaghim, S.; Borgelt, C.; Braune, C.; and Steinbrecher, M. 2022. Multi-layer perceptrons. In *Computational Intelligence*, 53–124. Springer.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191.

Levine, S.; Kumar, A.; Tucker, G.; and Fu, J. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.

Li, G.; Shi, L.; Chen, Y.; Gu, Y.; and Chi, Y. 2021. Breaking the sample complexity barrier to regret-optimal model-free reinforcement learning. *Advances in Neural Information Processing Systems*, 34.

Li, J.-A.; Dong, D.; Wei, Z.; Liu, Y.; Pan, Y.; Nori, F.; and Zhang, X. 2020. Quantum reinforcement learning during human decision-making. *Nature human behaviour*, 4(3): 294–307.

Liang, J.; Makoviychuk, V.; Handa, A.; Chentanez, N.; Macklin, M.; and Fox, D. 2018. Gpu-accelerated robotic simulation for distributed reinforcement learning. In *Conference on Robot Learning*, 270–282. PMLR.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Lin, X.; Wei, Z.; and Yao, P. 2021. Quantum and Classical Hybrid Generations for Classical Correlations. *IEEE Transactions on Information Theory*, 68(1): 302–310.

Lockwood, O.; and Si, M. 2020. Reinforcement learning with quantum variational circuit. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 16, 245–251.

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

Moerland, T. M.; Broekens, J.; and Jonker, C. M. 2020. Model-based reinforcement learning: A survey. *arXiv preprint arXiv:2006.16712*.

Nachum, O.; Gu, S. S.; Lee, H.; and Levine, S. 2018. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31.

Nguyen, N. H.; Behrman, E. C.; Moustafa, M. A.; and Steck, J. E. 2020. Benchmarking Neural Networks For Quantum Computations. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7): 2522–2531.

Peng, X. B.; Kanazawa, A.; Malik, J.; Abbeel, P.; and Levine, S. 2018. Sfv: Reinforcement learning of physical skills from videos. *ACM Transactions on Graphics (TOG)*, 37(6): 1–14.

Pérez-Salinas, A.; Cervera-Lierta, A.; Gil-Fuster, E.; and Latorre, J. I. 2020. Data re-uploading for a universal quantum classifier. *Quantum*, 4: 226.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. USA: John Wiley & Sons, Inc., 1st edition. ISBN 0471619779.

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

Schuld, M.; and Killoran, N. 2019. Quantum machine learning in feature Hilbert spaces. *Physical review letters*, 122(4): 040504.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *International conference on machine learning*, 1889–1897.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Seitzer, M.; Schölkopf, B.; and Martius, G. 2021. Causal influence detection for improving efficiency in reinforcement learning. *Advances in Neural Information Processing Systems*, 34.

Seno, T.; and Imai, M. 2021. d3rlpy: An Offline Deep Reinforcement Learning Library. *arXiv preprint arXiv:2111.03788*.

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419): 1140–1144.

Skolik, A.; Jerbi, S.; and Dunjko, V. 2022. Quantum agents in the gym: a variational quantum algorithm for deep q-learning. *Quantum*, 6: 720.

Sutton, R. S.; Barto, A. G.; et al. 1998. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Van Hasselt, H.; Guez, A.; and Silver, D. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Wang, D.; Sundaram, A.; Kothari, R.; Kapoor, A.; and Roetteler, M. 2021. Quantum algorithms for reinforcement learning with a generative model. In *International Conference on Machine Learning*, 10916–10926. PMLR.

Wang, Z.; Novikov, A.; Zolna, K.; Merel, J. S.; Springenberg, J. T.; Reed, S. E.; Shahriari, B.; Siegel, N.; Gulcehre, C.; Heess, N.; et al. 2020. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33: 7768–7778.

Yarats, D.; Brandfonbrener, D.; Liu, H.; Laskin, M.; Abbeel, P.; Lazaric, A.; and Pinto, L. 2022. Don't Change the Algorithm, Change the Data: Exploratory Data for Offline Reinforcement Learning. *arXiv preprint arXiv:2201.13425*.

Ye, D.; Liu, Z.; Sun, M.; Shi, B.; Zhao, P.; Wu, H.; Yu, H.; Yang, S.; Wu, X.; Guo, Q.; et al. 2019. Mastering Complex Control in MOBA Games with Deep Reinforcement Learning. *arXiv preprint arXiv:1912.09729*.

Yu, Y. 2018. Towards Sample Efficient Reinforcement Learning. In *IJCAI*, 5739–5743.

Zhong, H.-S.; Wang, H.; Deng, Y.-H.; Chen, M.-C.; Peng, L.-C.; Luo, Y.-H.; Qin, J.; Wu, D.; Ding, X.; Hu, Y.; et al. 2020. Quantum computational advantage using photons. *Science*, 370(6523): 1460–1463.