IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 19 October 2024, accepted 19 November 2024, date of publication 27 November 2024, date of current version 9 December 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3506656

RESEARCH ARTICLE

Optimization of Reinforcement Learning Using Quantum Computation

ROOPA RAVISH, NISCHAL R. BHAT, N. NANDAKUMAR, S. SAGAR, SUNIL, AND PRASAD B. HONNAVALLI, (Member, IEEE)
Department of Computer Science and Engineering, PES University, Ring Road Campus, Bengaluru, Karnataka 560085, India
Corresponding author: Roopa Ravish (rooparavish@pes.edu)

ABSTRACT Exploring the convergence of quantum computing and machine learning, this paper delves into Quantum Reinforcement Learning (QRL) with a specific focus on Variational Quantum Circuits (VQC). Alongside a comprehensive examination of the field, the study presents a concrete implementation of QRL tailored to foundational gym environments. Leveraging principles from Quantum Computing, experiments reveal QRL's potential in enhancing reinforcement learning tasks within these environments, showcasing notable space optimization for RL models. Quantitative results indicate a reduction in the number of trainable parameters by up to 90% when compared to classical approaches, significantly improving efficiency, mainly due to quantum principles of superposition and entanglement. This highlights the promising implications of integrating quantum computing techniques to address challenges and advance capabilities in fundamental reinforcement learning scenarios. Furthermore, the study discusses the adaptability of QRL algorithms to diverse problem domains, suggesting their potential for scalability and applicability beyond simple environments. Such versatility underscores the robustness and practical relevance of QRL methodologies in real-world scenarios, positioning them as valuable tools for tackling complex reinforcement learning challenges.

INDEX TERMS Advantage actor critic, deep Q network, policy gradient, Q learning, quantum approximate optimization algorithm, quantum computing, reinforcement learning, variational quantum circuit.

I. INTRODUCTION

In recent times, Quantum Computing has emerged as a solution to optimize the existing working of various fields. We currently live in the NISQ (Noisy Intermediate-scale Quantum) era, which corresponds to shortage of qubits for large-scale applications and significant error rates for precision-related applications. However, progress has been made in several areas [1], [2], [3]. One of these fields is Machine Learning, where there is a scope for improvement, especially in terms of space and time consumption. Among the branches of Machine Learning, Reinforcement Learning is a fairly recent one, and has come into the spotlight to solve certain kinds of complex problem, which include robots and games. [4], [5], [6]

The associate editor coordinating the review of this manuscript and approving it for publication was Chi-Yuan Chen.

In the realm of decision-making studies, classical reinforcement learning (CRL) has played a pivotal role, finding extensive applications [7]. Meanwhile, quantum reinforcement learning (QRL) has exhibited remarkable performance in computer simulations but awaits experimental validation in large-scale applications or precision-based scenarios. Bridging this gap, our work introduces a practical QRL implementation tailored to foundational gym environments. Quantum principles like superposition and entanglement offer several benefits in quantum computing approaches, including a reduction in the number of training parameters, computational speedups, and enhanced exploration of the state space. By leveraging the advantages offered by quantum computation [8], our experiments unravel the potential of QRL to enhance reinforcement learning tasks within environments. Notable results, including the space optimization achieved in various works and in our experiments [9], underscore the promising implications of integrating quantum computing techniques. It is crucial to note that, despite extensive simulation, the experimental testing of QRL in large-scale applications or precision-based scenarios remains an unexplored frontier, though works exist where such concepts are visualized for real life scenarios, like Electric Vehicles or Communication 6G networks. [10].

In this article, we provide a brief overview of quantum reinforcement learning (QRL) algorithms. We go through many approaches to solve a particular task by using quantum algorithms and circuits. In order to understand the QRL, we first go through an overview of Classical Reinforcement and Quantum Computing in Section II. In Quantum Computing section, we provide introduction to the Quantum Gates and Variational Quantum Circuits (VQC). [11] Variational quantum circuits are a class of quantum circuits used in quantum computing for optimization and machine learning tasks [12], [13], [14], [15]. They play a significant role in hybrid quantum-classical algorithms, where both quantum and classical components work together to solve complex problems [16].

Section II also presents a brief literature review, which contains an overview of the methods employed in Quantum Reinforcement Learning tasks in various articles, referencing relevant literature and their methodologies. It also contains details of environments used and a discussion of the insights from the review.

This is followed by our implementation of QRL on fundamental environments, where the methods used, and experimental setup are stated in Section III. The Section IV displays the results and offers a discussion on the outcomes. The paper concludes with a comprehensive conclusion in Section V, framing the insights from the preceding sections as a standalone contribution.

Our primary contribution is to achieve significant space optimization for reinforcement learning tasks within these environments, demonstrating the potential of QRL to improve efficiency. The paper demonstrates this by training novel QRL agents to achieve space optimization while maintaining rewards comparable to Classical RL approaches. Additionally, we provide an extensive literature review that integrates various methodologies and approaches in QRL, highlighting the advantages of hybrid quantum-classical algorithms in optimizing space for reinforcement learning tasks.

II. BACKGROUND KNOWLEDGE AND LITERATURE REVIEW

A. CLASSICAL REINFORCEMENT LEARNING

Reinforcement Learning (RL) is a machine learning approach in which agents learn to make decisions by interacting with their environment to maximize cumulative rewards, as shown in Figure 1. Unlike supervised learning, which relies on labeled data, RL emphasizes learning from the consequences of actions taken in a dynamic environment [17]. The core objective is to find an optimal policy – a strategy that dictates

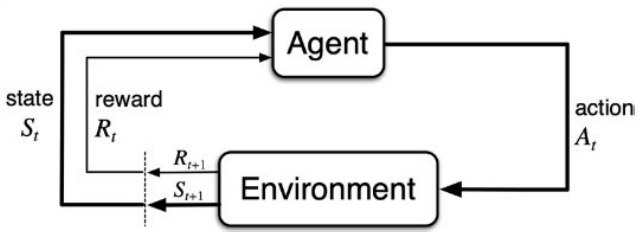


FIGURE 1. Interaction of the Agent with the Environment in a generic Reinforcement learning setup [21].

the best actions for maximizing long-term rewards [18]. This is often modeled using Markov Decision Processes (MDPs), which provide a framework for decision-making in uncertain settings [19]. RL methods generally fall into two categories: value-based, where the agent estimates future rewards (as seen in Q-learning), and policy-based, where the agent directly refines its decision-making strategy [20]. The Bellman equation further simplifies the calculation of state or action values by recursively breaking down the reward structure, enabling agents to make more efficient decisions based on expected future outcomes.

1) THE BELLMAN EQUATION

We use Bellman Equation to calculate state-action values [17]. It operates as follows: rather than computing the return for each state from the beginning, we can think of every state's value as:

R_{t+1} is the initial reward, and $R_{t+1} + (\gamma \cdot V(S_{t+1}))$ represents the discounted value of the subsequent state. Equation 1 goes as following

$$V_\pi(s) = \mathbf{E}_{[\pi]}\left[R_{t+1} + \gamma \cdot V_\pi(S_{t+1}) \mid S_t = s\right] \tag{1}$$

2) Q - LEARNING

Q-Learning is an off-policy value-based method that trains its value function using a value-based approach. Using an action-value function, the Q-learning algorithm identifies the value of a state and performs the corresponding action at that state [17]. Algorithm 1 explains the workflow for the Q-Learning. Q function uses Q-table for to get action for a corresponding state [22]. Q table has the value for each state-action pair. The optimal value function is defined by equation 2 below.

$$\pi^*(s) = \arg\max_a Q^*(s, a) \tag{2}$$

In the domain of reinforcement learning, several notable extensions and adaptations of the core concepts have emerged. One significant method is the actor-critic approach, which cleverly integrates ideas from both policy gradient [18] and value functions. Another refinement is the double Q-learning technique, which mitigates bias inherent in standard Q-learning by introducing an additional target action-value function.

Algorithm 1 Q-Learning Algorithm [22]

```
1:  procedure Q-Learning($\pi$, num_episodes, $\alpha$, $\{\epsilon_i\}$)
2:      Input:
            $\pi$: Target policy
            num_episodes: Number of episodes for learning
            $\alpha$: Learning rate
            $\{\epsilon_i\}$: Sequence of exploration parameters
3:      Output: value function $Q$ ($\approx q_\pi$ if num_episodes is large enough)
4:      Initialize $Q$ arbitrarily (e.g., $Q(s, a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$, and $Q(\text{terminal-state}, \cdot) = 0$)
5:      for $i \leftarrow 1$ to num_episodes do
6:          $\epsilon \leftarrow \epsilon_i$
7:          Observe $S_0$
8:          $t \leftarrow 0$
9:          repeat
10:             Choose action $A_t$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
11:             Take action $A_t$ and observe $R_{t+1}, S_{t+1}$
12:             $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$
13:             $t \leftarrow t + 1$
14:         until $S_t$ is terminal
15:     end for
16:     return $Q$
17: end procedure
```

3) DEEP Q NETWORK

Traditional Q-learning has difficulties when it comes to managing vast or continuous state spaces. Deep Q-Networks (DQN) are an extension of this technique. DQN uses deep neural networks to simulate the Q-value function, allowing it to scale to increasingly complex, high-dimensional settings, whereas Q-learning maintains a Q-value for every state-action pair in a table.

Experience replay, which randomly collects and saves previous interactions to break the association between consecutive encounters and increase stability, is one of DQN's main enhancements. Furthermore, DQN provides more stable target values during training by using a target network that updates less frequently. Because of these advancements, DQN is more reliable and effective than traditional Q-learning, especially when it comes to challenging tasks like visual or sequential decision-making, where DQN has demonstrated performance comparable to that of humans.

4) ADVANTAGE ACTOR CRITIC

Advantage Actor-Critic (A2C) is a reinforcement learning algorithm that combines elements of policy iteration and value iteration methods to efficiently train agents in sequential decision-making tasks. The key idea behind A2C is to have two components working in tandem: an actor and a critic. The actor is responsible for selecting actions, while the critic evaluates the chosen actions in terms of their advantage over alternative actions at each time step. The advantage function measures how much better a particular action is compared to the average expected return. By training the actor to maximize this advantage and simultaneously updating the critic to minimize the difference between the predicted and actual returns, A2C achieves a delicate balance between exploration and exploitation.

Unlike traditional policy gradient methods, A2C utilizes a value function (critic) to reduce the variance in the gradient estimates, leading to more stable and faster convergence during training. Moreover, A2C can be parallelized efficiently, allowing for accelerated learning by updating the policy and value function parameters concurrently across multiple environments. This parallelization makes A2C a computationally efficient and scalable algorithm, well-suited for a wide range of reinforcement learning applications.

B. QUANTUM COMPUTING DEFINITIONS

Quantum computing fundamentally differs from classical computing by using quantum bits, or qubits, which leverage quantum phenomena such as superposition and entanglement. Unlike classical bits, which can only exist in discrete states of 0 or 1, qubits can exist in a superposition of both states simultaneously. This characteristic allows qubits to encode richer data, significantly increasing computational capacity. For example, a single qubit can represent states $|0\rangle$ and $|1\rangle$ simultaneously, described mathematically in Equation 3

$$|\Psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle \tag{3}$$

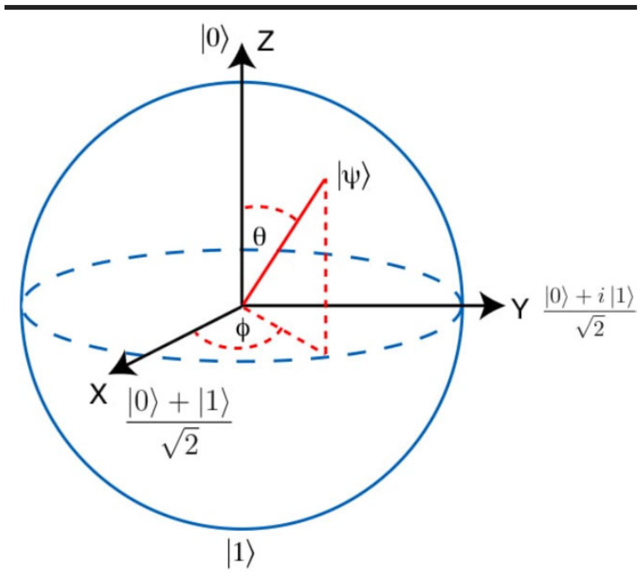
where α_0 and α_1 are complex amplitudes corresponding to the probability of measuring the qubit in state $|0\rangle$ or $|1\rangle$, respectively. Superposition enables quantum computers to perform multiple calculations at once, greatly enhancing processing efficiency. Every quantum state can be visualized as a point on a three-dimensional representation called the Bloch Sphere, as shown in the Figure 2.

Additionally, quantum gates manipulate qubits by adjusting their probabilities of states, allowing controlled transitions between superpositions. An essential feature is the entanglement gate, which creates a correlation between qubits such that the state of one qubit instantly affects the state of another, regardless of the distance separating them [23]. For instance, if two qubits are entangled in the state measuring one qubit will immediately determine the state of the other, showcasing the profound implications of quantum mechanics and enhancing the computational capabilities of quantum systems. An example for entangled state is shown in the Equation 4

$$|\Phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{4}$$

1) VARIATIONAL QUANTUM CIRCUITS

The hybrid quantum-classical technique known as the variational quantum circuit makes use of the advantages of



$|0\rangle$
Z
$|\psi\rangle$
θ
ϕ
Y
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
X
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
$|1\rangle$

FIGURE 2. Representation of quantum state on the Bloch Sphere. [24].

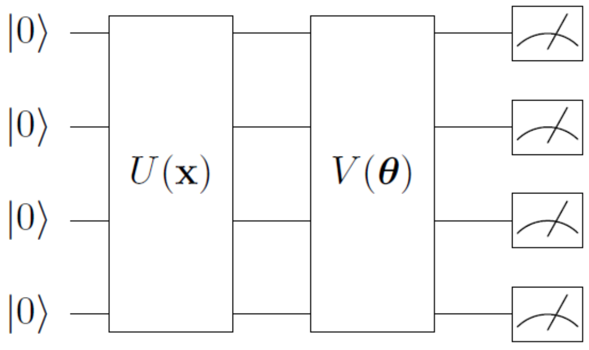


FIGURE 3. General form of VQC. U(x) stands for the Encoding layer, which helps to prepare the fixed input quantum state. V(θ) represents the Variational layer which is parameterized by a set of free parameters θ [28].

both quantum and classical computation [25]. It is a particular class of quantum circuit with controllable parameters that a classical computer iteratively optimizes. These variables are visible. In artificial neural networks, they serve as the weights. The flexibility in circuit depth and some noise resistance of the variational quantum circuit technique have been demonstrated. It consists of encoding layers and entanglement layers. VQCs, often referred to as parameterized quantum circuits (PQC) in the literature, are a unique class of quantum circuits that have tunable parameters. The optimization techniques used to train such parameters were created in the traditional ML communities. Gradient-based or gradient-free optimization are both options. Quantum machine learning examples can take advantage of VQC. [26], [27]. Figure 3 shows the general form of VQC.

The figure 4 shows the comparison between a Deep Neural Network and a Variational Quantum Circuit.

C. ARTICLES ON QRL

Quantum reinforcement learning is an emerging field that combines principles of quantum mechanics with reinforcement learning techniques. It aims to leverage the unique properties of quantum systems, such as superposition and entanglement, to enhance the efficiency and effectiveness of reinforcement learning algorithms [30].

The literature review includes several key articles on Quantum Reinforcement Learning (QRL), highlighting significant advancements in space optimization. A concise summary of each article is provided, offering an introductory understanding of their contributions. For a more detailed analysis and comprehensive review, please refer to the GitHub repository [31].

1) ENVIRONMENTS

In our survey of the literature, multiple environments have been utilized by authors to evaluate the performance of Quantum Reinforcement Learning (QRL) algorithms. The following are definitions for the most commonly discussed environments.

- **Acrobot**: An environment defined in the vertical plane, consisting of an arm with two links, resembling a double pendulum. The goal is to apply torque at the joint to make the lower link swing to a specified height [32].
- **Blackjack**: A card game environment where the agent aims to accumulate a score as close to 21 as possible without exceeding it. The agent receives rewards based on how well it achieves this goal.
- **Cognitive Radio**: The objective in this environment is for the agent to select one out of multiple available channels. The agent must choose an unoccupied channel to optimize communication [33].
- **Cartpole**: This is an inverted pendulum problem where a pole is attached to a cart moving horizontally. The agent's task is to balance the pole by controlling the cart's motion, preventing it from falling [34], [35].
- **Frozen Lake**: A maze-type environment where the agent must navigate a frozen lake to reach a goal. The challenge is to avoid slipping into holes while finding the optimal path [35], [36].
- **Lunar Lander**: The goal of this environment is to control a lunar lander and safely land it in a designated region. The agent must manage fuel and control the lander's velocity to succeed.
- **Minigrid**: A collection of grid-world environments, where the agent must navigate small, discrete grids. These environments test the agent's ability to find optimal paths and solve puzzles [37].
- **Mountain Car**: In this environment, the agent controls a car attempting to drive up a steep hill. The challenge is to adjust the car's acceleration to build enough momentum to reach the summit.
- **Robotic Arm**: A robotic arm environment where the agent learns to manipulate one or more links to perform

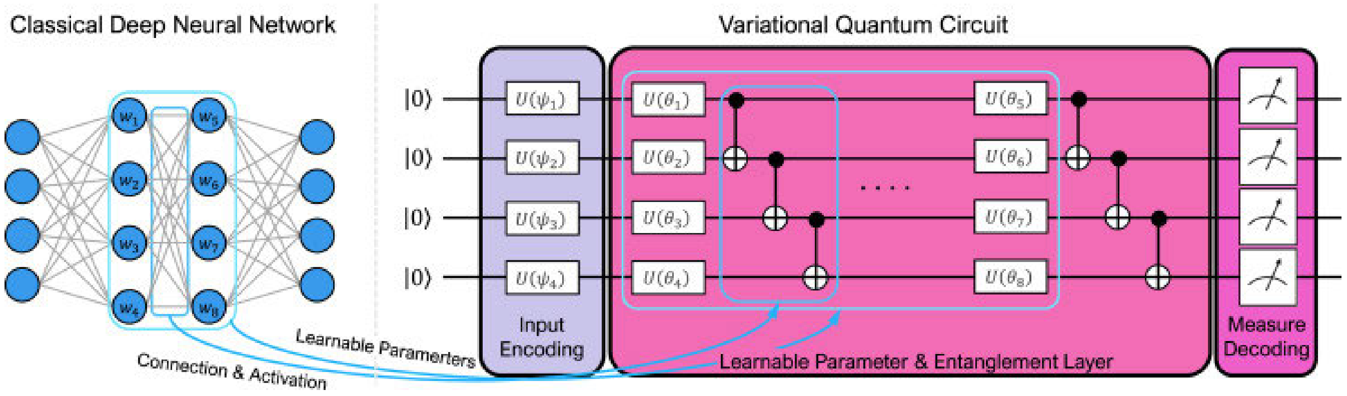


FIGURE 4. Comparison between the Deep Neural Networks and Variational Quantum Circuits. Here, the parameters in the quantum circuit are analogous to the weights of a neural network. [29].

specific tasks. These tasks may involve picking up objects, stacking, or reaching a target.

- **Turtlebot2**: This environment is used for robot navigation tasks. The agent, controlling the Turtlebot2, must learn to navigate through obstacles and reach predefined goals in various settings.

Most of the algorithms are online, which requires interacting with the environment on quantum computers. There are however examples of articles following offline learning [38]. Here are a few brief explanations of some quantum reinforcement learning algorithms:

2) QUANTUM REINFORCEMENT LEARNING USING VQC

Quantum Reinforcement Learning using Variational Quantum Circuits (VQC) is an approach that combines reinforcement learning with variational quantum algorithms. VQC is a framework that utilizes quantum circuits with trainable parameters to solve optimization problems. In the context of reinforcement learning, VQC can be used to learn and optimize policies for sequential decision-making tasks [39], [40].

a: VQC FOR DRL

The article by Chen et al. [41] presents a novel approach that combines Variational Quantum Circuit (VQC) with Deep Q-Network (DQN) to enhance the performance of Deep Reinforcement Learning (DRL) algorithms in two distinct environments. These environments are Frozen Lake [35], [36] and Cognitive Radio [33].

The results of the study demonstrate the effectiveness of the proposed approach on both Classical simulators and Quantum computers. Notably, there is a notable improvement in the efficiency of the reinforcement learning process, achieved by reducing the number of parameters required [41]. The combination of VQC and DQN holds significant promise for enhancing the efficiency and effectiveness of DRL algorithms in complex environments [42].

b: RL WITH QVC

The article from Lockwood et al. [43] highlights the application of Variational Quantum Circuits (VQC) in

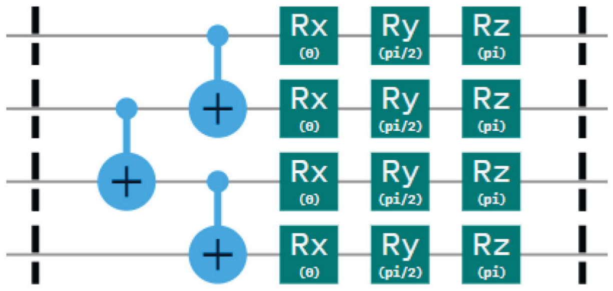


FIGURE 5. One layer of the QVC, composed of CNOT and parametrized rotation gates [43].

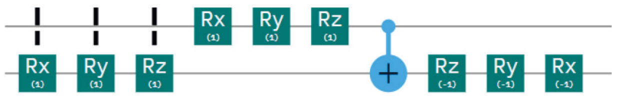


FIGURE 6. Quantum pooling operation(Single Pooling Operation [43]).

performing reinforcement learning, focusing on two specific environments: Cartpole [34], [35] and Blackjack [32]. The quantum circuits used in the articles can be seen in Figs. 5 and 6.

The results of the study demonstrate that the proposed VQC-based approach outperforms classical approaches, achieving a faster time to reach a reward threshold in both the Cartpole and Blackjack environments. This showcases the prospect of using VQC in enhancing the efficiency and effectiveness of reinforcement learning tasks. Overall, this survey highlights the successful application of VQC with DQN and DDQN algorithms in the context of reinforcement learning, providing valuable insights into the advantages of quantum-inspired techniques in complex environments such as Cartpole and Blackjack [42].

c: VQRL USING EVOLUTIONARY OPTIMIZATION

The article by Chen et al. [9] delves into the utilization of evolutionary optimization in Variational Quantum Reinforcement Learning (VQRL) problems. The goal of the article is to leverage evolutionary optimization concepts to tackle these challenges. The study focuses on two distinct environments:

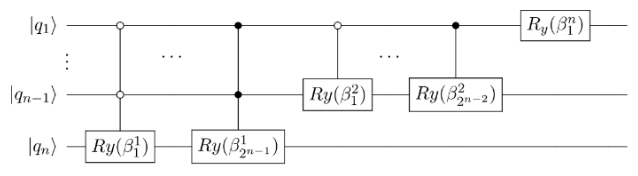


FIGURE 7. Amplitude encoding routine pre-inversion [9].

Cartpole [35] and Minigrid [37]. The amplitude encoding is used to convert the observation into amplitudes of quantum states, as shown in Figure 7

The results of the study indicate noteworthy findings. In the Cartpole problem, the proposed approach employs only 26 parameters, which is less compared to classical approaches. For the Minigrid problem, the article explores different grid dimensions and observes that the TN-VQC model with a chi value of 1 outperforms others in terms of requiring fewer generations to achieve higher scores. The results demonstrate the potential of these approaches in reducing parameter complexity and improving performance in complex reinforcement learning scenarios.

d: RL BASED VQC OPTIMIZATION FOR COMBINATORIAL PROBLEMS

The article by Khairy et al. [44] focuses on the application of Reinforcement Learning (RL) techniques in optimizing Variational Quantum Circuits (VQC) for combinatorial problems, specifically the Max-Cut problem on Erdos-Renyi graphs. The objective of the article was to achieve the maximum cut values in the given graphs. Performed on Erdos-Renyi graphs to obtain the Max-Cut. The methodology employed in this article revolves around the Quantum Approximate Optimization Algorithm (QAOA) [45], [46], [47].

The results of the study highlight several key observations. Increasing the depth of the QAOA circuit is found to enhance the approximation ratio, leading to improved performance [45]. The reward is obtained quickly, but a wiggly behavior is observed thereafter, indicating potential areas for future investigation and optimization.

e: ON THE USE OF QRL IN ENERGY-EFFICIENCY SCENARIOS

The article by Andres et al. [48] explores the application of Quantum Reinforcement Learning (QRL) in energy-efficient scenarios, with a focus on HVAC control, energy management in EV vehicles, and charging station optimization. Such environments have been extensively studied in several articles addressing Classical RL problems [49], [50], [51], [52], [53].The objective of the article is to investigate the benefits and limitations of QRL in solving energy-efficient problems [54].

The results obtained from the article are so: For HVAC, the computation time is larger for the model using VQC, which is a consequence of using quantum simulators. It however results in a better reward, with a significant difference in the rewards obtained in classical and quantum layer.

For EV vehicles, the quantum agent performed the management of energy better than the classical agent. However, the computation time is still a major overhead in quantum agent. VQC here overtakes the classical agent only at later time.

For charging stations, the quantum agent attains the optimal policy. Even though classical agent has faster convergence, it seems to get stuck in a local optimum.

f: COMPARING QHRL TO CLASSICAL METHOD

The article by Moll et al. [15] aims to compare Hybrid Quantum Reinforcement Learning (QHRL) with existing classical methods. The goal is to evaluate the VQ-DQN algorithm's performance in training the agent. The Frozen-lake scenario was employed, which is a basic maze style environment. This article considers four different models for comparison: Q Learning, DQN, non-pure VQ-DQN [41], and the VQ-DQN circuit proposed by the authors.

The results of the study indicate that the VQ-DQN circuit requires a lower number of parameters to train the agent compared to its classical equivalents. This finding suggests that the hybrid quantum RL approach offers advantages in terms of parameter efficiency.

g: ROBUSTNESS OF QRL UNDER HARDWARE ERRORS

The article by Skolik et al [55]. attempts to investigate the robustness of Quantum Reinforcement Learning (QRL) algorithms in handling shot noise, coherent errors, and incoherent errors [56]. The study focuses on two environments which uses VQC, namely Cartpole and the Travelling Salesman Problem, and implements Q Learning and Policy Gradient approaches for both environments [18].

The results demonstrate that both Q Learning and Policy Gradient methods exhibit robustness to the errors considered, suggesting the potential for running these algorithms on quantum computers. The article provides a valuable method for maintaining a certain level of robustness in such quantum systems, which is essential for practical applications.

h: PARAMETRIZED QUANTUM POLICIES FOR RL

This article by Jerbi et al. [57] explores the use of policies based on Parameterized Quantum Circuits (PQCs) for classical Reinforcement Learning (RL) environments like Acrobot, Cart Pole and Mountain Car.

They demonstrate that PQC policies can rival the performance of established Deep Neural Network (DNN) policies in benchmarking environments. Furthermore, the work demonstrates an empirical benefit of PQC rules over normal DNN policies in difficult RL tasks. The researchers also design environments where PQC policies outperform classical learners, incorporating the discrete logarithm problem, known for its quantum computational advantage. This research highlights the potential of quantum-inspired policies in RL settings.

The PQC traits employed by the authors increases the expressivity and adaptability of PQC policies (such as quantum classifiers), allowing them to train well in benchmarking environments that are comparable to those used by conventional DNNs.

i: INTRODUCTION TO QRL: THEORY AND PENNYLANE-BASED IMPLEMENTATION

The article by Kwak et al. [11] focuses on performing Quantum Reinforcement Learning (QRL) on the Cartpole environment. The objective is to introduce QRL theory and demonstrate its implementation using the Pennylane framework [58].

The results indicate that QRL with Variational Quantum Policies exhibits a lower number of parameters compared to classical approaches. However, handling noise introduced by quantum computation becomes challenging as the deviation of rewards is high.

j: QDRL FOR ROBOT NAVIGATION TASKS

The article by Heimann et al. [59] focuses on applying Quantum Deep Reinforcement Learning (QDRL) to a simulated robot task in the Turtlebot2 environment [60].

The results indicate that the quantum cases require an order of magnitude fewer parameters compared to their classical equivalent. However, the training time is the shortest in the classical approach. The second quantum case demonstrates improved training speed compared to the first quantum case and shows similar performance to the classical approach.

k: QUANTUM DEEP RECURRENT RL

The article by Chen et al. [28] discusses recurrent connections that store the memory of the past time steps. The article uses something called as quantum long short term memory (QLSTM) [61] to perform QRL on Cart Pole environment. The circuit used is displayed in Figure 8

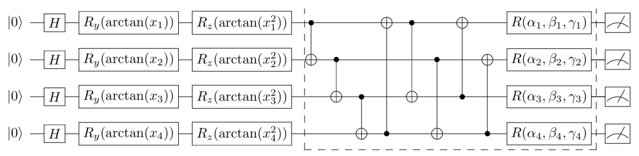


FIGURE 8. VQC Architecture for QLSTM used in the article [28].

In the Cart-Pole environment [35], the study compares different models for reinforcement learning (RL) agents. When the agent has complete access to the state of the environment, the Quantum Long Short-Term Memory (QLSTM) model with two Quantum Variational Circuit (VQC) layers outperforms the other studied configurations.

Quantum models typically outperform their classical counterparts in terms of both stability and average scores. Learning is slower and more difficult in a partially observable environment, where the agent only observes some information [62]. However, in terms of stability and performance, quantum models continue to outperform classical models. The classical LSTM with 16 hidden neurons, in particular, falters after 800 training events, but QLSTM agents remain stable.

l: UNENTANGLED QRL AGENTS IN THE OPENAI GYM

The article Hsiao et al. [5] aims to explore the use of Quantum Reinforcement Learning (QRL) agents that employ only single-qubit gates without entanglement. The objective is to leverage a novel approach called SVQC (Single qubit Variational Quantum Circuit) for QRL. Various environments are used such as Cartpole, Blackjack, Frozenlake, Acrobot, and Lunar Lander [35].

The results demonstrate that the proposed SVQC model achieves better rewards compared to existing VQC models. Additionally, the SVQC model exhibits improved convergence speed when compared to classical neural networks. Moreover, the SVQC model utilizes a smaller number of parameters while delivering superior performance. The experiments were conducted on a quantum simulator.

m: QRL IN CONTINUOUS ACTION SPACE

Wu et al. [63] propose a Quantum Reinforcement Learning (QRL) framework addressing both Continuous Action Space (CAS) [64] and Discrete Action Space (DAS) [65]. For CAS, they develop a quantum Deep Deterministic Policy Gradient (DDPG) algorithm, using state amplitude encoding to manage dimensionality challenges. Their classical simulations demonstrate the framework's effectiveness in quantum control problems like eigenstate preparation and state generation, particularly in low-dimensional systems.

The authors introduced a quantum "environment" register representing the RL environment and used Quantum Neural Networks (QNNs) to guide the system from its initial to target states through a sequence of unitary gates. They analyzed gate complexity for a single RL iteration and determined that the method's efficiency depends on whether Variational Quantum Circuits (VQCs) can be executed with an efficient gate complexity of poly(log N).

3) ARTICLES NOT USING VQC FOR QUANTUM REINFORCEMENT LEARNING

The below sections contain summaries of those articles which do not use VQC to implement a QRL pipeline. During the process of the literature review, we examined these articles to understand the different methodologies and their potential to solve QRL problems in different environments.

a: DRL CONTROL OF QUANTUM CARTPOLES

Wang et al. [66] apply deep reinforcement learning (DQN) to a quantum cartpole control problem, analogous to the classical cartpole setup. They utilize Q-learning, which estimates future rewards via the Q function.

The paper demonstrates that deep reinforcement learning (DRL) effectively solves quantum control problems, matching or surpassing optimal control strategies when no optimal solutions exist. The approach was tested on a harmonic oscillator with varying measurement strengths, and results confirmed its robustness, with gradual performance degradation as measurement efficiency decreased. Classical Linear Quadratic Gaussian (LQG) control performed well in cases where the system's state was Gaussian-like, meaning the states were normally distributed, which is easier for classical controllers to handle due to predictable behavior. However, in systems with non-Gaussian states—where the distribution is irregular and exhibits significant deviations from normality, such as in the cooling of quartic oscillators—LQG struggled. In contrast, DRL managed both Gaussian and non-Gaussian states effectively, demonstrating its versatility and superiority in handling a wider range of quantum control scenarios

b: MULTIQUBIT AND MULTILEVEL RL WITH QUANTUM TECHNOLOGIES

Lamata et al. [67] explore the application of quantum reinforcement learning (QRL) in multiqubit and multilevel systems. Their proposed QRL protocol involves encoding environmental data into register states, which then interact with the agent, leading to state changes in the agent.

Their research demonstrates that QRL can be applied across various quantum technologies, maintaining steady learning times and adaptability to imperfectly known environments. This approach is suitable for real-world applications, including superconducting circuits and trapped ions, paving the way for scalable quantum devices.

c: RL WITH NEURAL NETWORKS FOR QUANTUM FEEDBACK

Fosel et al. [68] present a network-based reinforcement learning (RL) approach to quantum feedback, focusing on developing new Quantum Error Correction (QEC) methods for few qubit systems under random noise and hardware constraints [69]. Their autonomous, human-guidance-free method employs a network agent to develop and modify feedback techniques based on assessment outcomes.

Given the impossibility of classically simulating a full-scale quantum computer, modular approaches to quantum computation and devices are being developed. The hierarchical application of the quantum-module concept, with error-correction strategies applied to coupled modules, is theoretically feasible and well-suited to meet this challenge.

d: EXPERIMENTAL QUANTUM SPEED-UP RL AGENTS

Saggio et al. [70] explore reinforcement learning (RL) experiments using quantum computing to enhance communication channels between agents and the environment. While recent quantum-based studies focus on efficient decision-making algorithms, they have not achieved reductions in learning time until now.

This approach quantifies the reduction in learning time, allowing the agent to evaluate performance and shorten the learning period, outperforming agents using conventional communication. Additionally, new photonic circuit technology offers compactness, tunability, and low-loss transmission.

e: QRL VIA POLICY ITERATION

The article by Cherrat et al. [71] describes a quantum variant of the classical policy iteration technique that the researchers name Quantum Reinforcement Learning through Policy Iteration (QRL-PI). The Markov decision process (MDP) is a mathematical framework for modelling decision-making issues. The QRL-PI approach is meant to find optimum policies for MDPs.

The approach is effective in environments like FrozenLake and InvertedPendulum [35]. Further research is needed to explore environments where quantum linear algebra surpasses classical methods. Optimizing measurements and addressing inherent noise in quantum procedures are crucial. Future work may extend theoretical guarantees for quantum policy iteration variants, enhancing the understanding and applicability of QRL.

f: ROBUST OPTIMIZATION FOR QRL CONTROL USING PARTIAL OBSERVATIONS

Jiang et al. [62] discuss the role of quantum control in areas like quantum communication and scalable computing, citing applications such as State Steering and Quantum Approximate Optimization Algorithm (QAOA) [45], [46]. They address Hamiltonian uncertainty and control precision issues, proposing Reinforcement Learning (RL) as an alternative method. Their RL scheme relies solely on partial observations for control decisions, eliminating the need for extra quantum measurements in reward functions.

Their quantum RL algorithm calculates rewards based on partial observations, reducing measurement and computational costs significantly. Unlike methods relying on classical simulation or fidelity, it operates practically for near-term quantum devices, adapting to varying noise levels and initial states [42], [72].

4) SURVEY DISCUSSION

a: ADOPTION OF VARIATIONAL QUANTUM CIRCUITS (VQC)

Many studies in the Table 1 highlight the effectiveness of Variational Quantum Circuits (VQCs) in Quantum Reinforcement Learning (QRL) for solving complex environments. VQCs are particularly valuable for encoding intricate functions with fewer parameters compared to classical networks. Our implementation incorporates VQCs in both the policy network and the target network, aiming to leverage their parameter efficiency while ensuring a compact and effective model.

b: PARAMETER REDUCTION IN QUANTUM ARCHITECTURES

Several papers emphasize a significant reduction in the number of parameters needed for quantum agents compared

TABLE 1. Literature References Table. The table contains the details on the approach used for QRL, based on the Method used and if it is a Variational algorithm.

Sl No.	Citation	Type	Method	Variational Algorithms	Environments
1	[28]	Implementation	QLSTM	Yes	Cart Pole
2	[41]	Implementation	VQC, DQN	Yes	Frozen Lake, Cognitive Radio
3	[43]	Implementation	VQC, DQN	Yes	Cart Pole, Blackjack
4	[9]	Implementation	TN, VQC	Yes	Cart Pole, Minigrid
5	[44]	Implementation	QAOA, PPO	Yes	Erdos-Renyi Graph
6	[48]	Implementation	DQN, VQC	Yes	Eplus, Prius, Custom charging station environment
7	[15]	Implementation	DQN, VQC	Yes	Frozen Lake
8	[55]	Implementation	Q Learning, Policy Gradient	Yes	Cart Pole, Travelling Salesman Problem
9	[11]	Implementation	VQ Policy Circuit	Yes	Cart Pole
10	[59]	Implementation	DDQN	Yes	Turtlebot
11	[5]	Implementation	SVQC	Yes	Multiple gym environments
12	[67]	Theory	Quantum Circuit	No	General Environment
13	[68]	Implementation	Neural Network	No	Qubit system
14	[66]	Implementation	Deep QL, LQG Control	No	Quantum Cart Pole
15	[71]	Implementation	Quantum Policy Iteration	No	Frozen Lake, Inverted Pendulum
16	[57]	Implementation	Softmax, PQC	Yes	Cart Pole, Mountain Car, Acrobot
17	[63]	Implementation	Quantum DDPG	Yes	Continuous and Discrete action environments
18	[62]	Implementation	QAOA	No	Combinatorial Optimization
19	[70]	Implementation	Grover-like algorithm	No	Quantum Environment

to their classical counterparts. This advantage informs our design choices, allowing us to minimize training complexity and computational overhead without sacrificing performance. By utilizing VQCs, we aim to achieve faster training times and improved resource efficiency.

c: STABLE PERFORMANCE AND REWARD COMPARISONS

The literature indicates that quantum agents can achieve stable performance levels, often yielding rewards that are equal to or greater than those generated by classical reinforcement learning algorithms. We take this stability into account in our QRL implementation, focusing on environments where sustained performance is essential, and ensuring that our VQC-based approach can deliver consistent and high-quality results.

d: POLICY-BASED APPROACH WITH TARGET NETWORKS

Some of studies advocate for a policy-based reinforcement learning framework, especially when enhanced by the use of target networks. In our implementation, both the policy network and the target network are constructed using VQCs. This design promotes stable policy updates, as the target network provides a reliable reference point for optimizing the policy, while the quantum architecture enhances the exploration capacity of our agent.

e: HYBRID ARCHITECTURE

One of our approaches in the articles employs a hybrid Actor-Critic framework where the actor is quantum-based, utilizing VQCs for policy representation, while the critic remains classical. This structure allows the quantum actor to explore diverse policies effectively, while the classical critic refines the value estimates, improving the parameter values in the actor. This combination optimizes learning efficiency, leveraging the strengths of both quantum computation for exploration and classical methods for stability and performance enhancement.

In reviewing the existing literature on Quantum Reinforcement Learning (QRL), we identified several common obstacles that impact the effectiveness and efficiency of quantum agents. While QRL shows promise in solving complex environments with fewer parameters, certain limitations hinder its full potential. These challenges include:

f: LONGER TRAINING TIME FOR QUANTUM AGENTS

Quantum agents often require extended training periods due to the complexity of optimizing Variational Quantum Circuits (VQCs). This increases the computational resources needed to achieve convergence compared to classical methods.

g: UNSTABLE LEARNING DUE TO CIRCUIT DEPTH

Deeper quantum circuits can cause instability in the learning process, leading to oscillations or "wiggly" performance. Finding the right balance in circuit depth is critical for maintaining stable and effective learning.

h: IMPACT OF QUANTUM NOISE

Noise is a persistent issue in quantum computing, degrading quantum state accuracy and negatively impacting learning outcomes. Even small amounts of noise can lead to suboptimal policy updates and unpredictable performance.

Based on the conducted survey, we arrived at a collection of environments to perform QRL upon. Since VQC has been proven to be useful for solving the environments with fewer parameters, our implementations would involve them in order to achieve good performance and space optimization.

III. METHODOLOGY AND IMPLEMENTATION

This section contains information on our implementation of QRL on fundamental environments. We have used insights from the literature survey to arrive at novel methods to perform Reinforcement Learning.

A. FROZEN LAKE

The Frozen Lake is a well-known grid-world simulation that is frequently used in reinforcement learning. It depicts a frozen grid in which an agent navigates from a starting position to a goal while avoiding ice holes. The grid is made up of various types of cells, such as safe frozen surfaces, holes that lead to failure if stepped into, a starting point, and a goal. The agent can perform discrete actions such as moving up, down, left, or right. The difficulty stems from the environment's stochastic nature, as actions may not always lead in the intended direction due to slippery ice. Furthermore, the agent is at risk of falling into holes, making it critical to learn a policy that maximizes the probability of safely reaching the goal. We performed two different QRL methods for Frozen lake environment. These are

- Using VQC as a Policy to fill up the Q-Table.
- Using VQC as a Policy, where the Policy is optimized.

The first methodology involves using the VQC initialized with random parameters to fill up the cells of the Q-Table. This involves not actually optimizing the number of parameters, as we still have to find the values for every state-action pair.

The second method involves trying to learn the policy using VQC, by learning the optimal parameters for the results. The parameters in this case are the angle inputs to the rotation gates of the VQC, and corresponding with the number of layers and the number of qubits.

1) VQC WITH Q-TABLE

Using the Bellman equation, the algorithm here iteratively updates the Q-values based on observed rewards and estimates of future rewards. During training, the Q-table is gradually filled as the agent explores the environment. The agent chooses actions with the highest Q-values for a given state during decision-making.

We use the redefined Frozen Lake environment, accounting for the observation state size, where we map a single integer to an observation state of size 4. Here, we convert the state integer value to a bit representation array.

The VQC based Policy is assigned with random parameters. The input to the circuit is the state information, and the measurements correspond to the four possible actions in the frozen lake environment. The figure 9 describes the process applied.

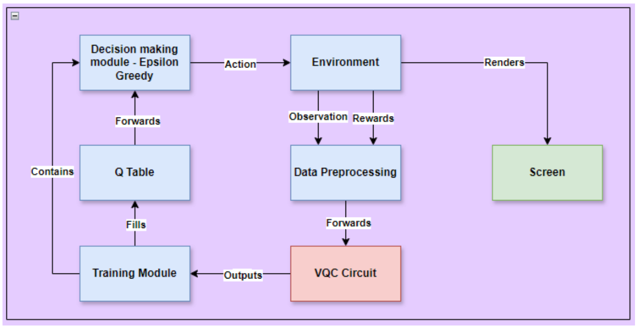


FIGURE 9. Workflow for Q Learning with VQC for the Frozen Lake Environment. We observe that the VQC Circuit behaves like an agent, receiving the processed observation and reward, and provides the best action based for the same.

This approach is model free, as we are not training a model to represent the environment. Instead, we let the agent explore the environment and allow it to fill the Q Table. The functionality of VQC here is to just return what action to take for a particular state. This approach is well-suited for the Frozen Lake environment, given its discrete observation and action spaces. In such cases, a Q-Table can effectively be utilized to model the agent's learning process.

Based on the measured output of the circuit, which is determined by the shape and the gates in the circuit, a decision is taken to select the action for the state. This requires mapping of each qubit to one of the four possible actions.

2) POLICY BASED METHOD USING VQC

The key concept in policy-based learning is to use a neural network to parameterize the policy. The network receives the state as input and produces a probability distribution over actions as output. Training entails adjusting the network's parameters to maximize the expected cumulative reward, typically using gradient ascent techniques. This method enables the learning of stochastic policies, which can be useful in dealing with exploration-exploitation trade-offs.

In this case, the Policy to be trained is the VQC circuit. We can observe the circuit being used for Frozen Lake environment in the Fig 11. Using Pennylane functionality, we are able to define a Machine Learning model to act as the Policy of the agent, which contains of quantum circuits. A Replay memory buffer is used to store transitions which can be sampled for training. The epsilon value for exploration is subjected to Epsilon-Greedy approach, where the agent learns to exploit rather than explore as the training process continues. The training module employs two Variational Quantum Circuits (VQCs): one representing the policy network and the other serving as the target network. The policy network VQC is responsible for selecting actions based on the current state, guiding the agent's decision-making process. Meanwhile, the target network VQC provides stable reference values during training, helping to stabilize learning and improve convergence. This dual-VQC approach ensures

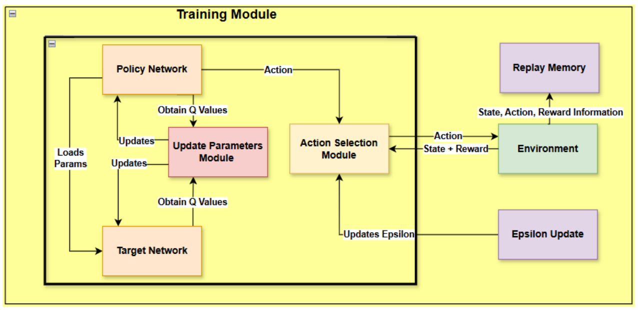


FIGURE 10. **Policy Based RL with VQC. Here, the Agent consists of both the Policy network and the Target network, each consisting a quantum circuit. Parameters of both the circuits are updated, and then an action is given as output.**

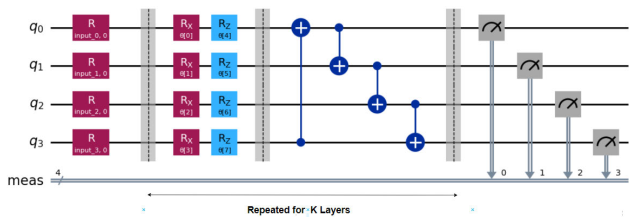


FIGURE 11. **Quantum Circuit used for both the Frozen Lake and Cart Pole environments, using 4 qubits. The key difference lies in the nature of the input and how the measurement maps to each of the possible actions in the respective environments. The first part is the encoding layer, where the preprocessed observation from the environment is fed into. Then comes the K layers of variational layers, denoted by rotational gates and entanglement gates. Finally, we measure each qubit to obtain the probabilities, based on which an action is chosen.**

more robust and efficient training in policy-based Quantum Reinforcement Learning scenarios. The Figure 10 displays the flow of this process.

For this example, we have used redefined Frozen lake environment. In this environment, the agent achieves a $+10$ reward for reaching the goal state. Reaching a hole state gives a negative reward of -1, and every step that does not lead to a terminal state results in a -0.05 reward. The idea behind these values is to allow the agent to reach the goal and also learn to do it in optimal number of steps. The original environment is sparse due to the nature of rewards, which is resolved by us redefining the rewards.

B. CART POLE ENVIRONMENT

A pole is attached to a cart in this two-dimensional simulation and the goal is to keep the pole from falling over by applying left or right forces to the cart. The environment's state is defined by four continuous variables: the cart's position, velocity, pole angle, and pole angular velocity. The cart can travel horizontally on a frictionless track and the pole can freely rotate. If the pole tilts beyond a certain angle or the cart moves outside predefined bounds, the episode ends. The goal is to devise a policy that applies actions to the cart in such a way that the pole remains upright for as long as possible, thereby challenging the agent to learn effective control strategies.

1) POLICY BASED METHOD USING VQC

The setup for this is similar to Frozen Lake Policy-Based method setup.

The important difference is the nature of the circuit. The Cart pole environment returns an observation of size 4. Therefore, we will require 4 qubits in quantum circuit. Since the Cart can take only 2 possible actions, we have to map measurements of 2 qubits for each action.

C. LUNAR LANDER ENVIRONMENT

The Lunar Lander simulation is a physics-based environment for testing and developing reinforcement learning algorithms. In this scenario, a lunar lander must safely descend from lunar orbit to the surface while contending with gravity, thrust, and limited fuel. The lander's position, velocity, angle, and angular velocity are all continuous variables that define the state of the environment. The lander can perform discrete actions, such as firing its engines to control descent and rotation. The primary goal is to gently land the spacecraft on a flat surface, with rewards and penalties based on factors such as fuel consumption, landing precision, and avoiding collisions.

Keeping the complexity of solving Lunar Lander environment in mind, we performed two different QRL methods for the Lunar Lander environment. These are

- Using VQC as a Policy, where the parameters are updated.
- Using VQC to implement an actor in Advantage Actor Critic, while using a Classical Critic.

The first methodology involves using training the DQN Policy network, by updating the parameters. The Policy is defined using Quantum Circuit, where the measurements are mapped to certain actions, based on the observation information. The idea is to improve the Policy network by training.

The second methodology involves using training both the Quantum Actor and the Classical Critic considering Advantage, to update the parameters of both the Actor and Critic to provide optimal performance. The Quantum Actor is implemented using a circuit, and returns a probability distribution of actions. The Classical Actor returns a value which is used to calculate the advantage.

1) POLICY BASED METHOD USING VQC

The setup for this is similar to Frozen Lake Policy Based method setup.

The important difference is the nature of the circuit. The Lunar Lander environment returns an observation of size 8. Therefore, we will require 8 qubits in the quantum circuit. Since the Cart can take only 4 possible actions, we have to map measurements of 4 qubits for each action.

2) HYBRID ADVANTAGE ACTOR CRITIC USING VQC

Advantage Actor Critic (A2C) is a powerful reinforcement learning algorithm that bridges the gap between policy-based and value-based methods. In A2C, an actor network develops

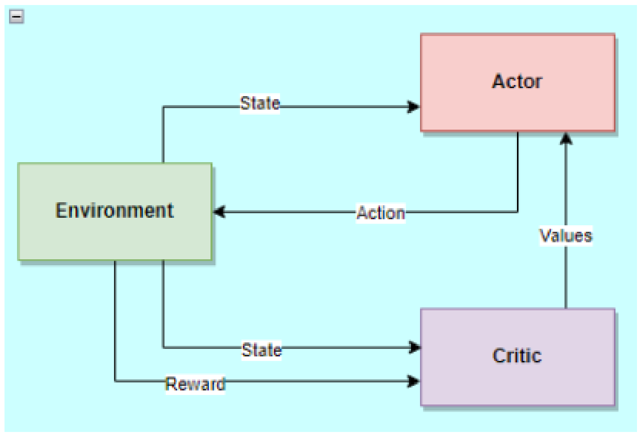


FIGURE 12. Workflow for A2C (Actor Advantage Critic) using VQC. Here the Actor contains a quantum circuit, meanwhile the Critic is purely classical.

policy by mapping states to actions, while a critic network assesses the current state's worth. The key innovation is the introduction of an advantage function, which compares the benefit of taking a specific action in a given state to the average action value.

A2C combines the strengths of both approaches by incorporating this advantage into policy gradient and value function updates, fostering stability and efficiency in learning. This integration is especially useful in environments with continuous action spaces, allowing A2C to effectively navigate complex decision landscapes. Figure 12 displays the A2C workflow.

The idea here was to use a Quantum Actor and a Classical Critic. The reasons for these are as follows:

- Having both Actor and Critic use Quantum Circuit will increase the complexity in terms of the time taken to train.
- Having a Quantum Actor allows us to use it as it is in the Testing Phase
- The Classical Critic allows obtaining a singular value as output, which is faster compared to Quantum analogue. In the latter case, we would have to measure the tensor product of all qubits, which is not very practical.

The Quantum Actor is implemented using a Variational Quantum Circuit (VQC) and is responsible for determining the optimal action in each state of the environment. The corresponding circuit is shown in Fig 13. The measurements correspond to the actions. This Classical Critic then obtains the rewards and the next state, which is processed, and a singular value is obtained. The output of the Critic is then used to train the Actor and obtain the parameters for the VQC circuit.

For this method, we had to redefine the reward structure for Lunar Lander. Lunar Lander on itself, is a difficult environment to solve due to the fact that it gets a large reward only when it lands perfectly in the landing pad without crashing. For the purpose of making it easier, we give higher priorities to events like leg in contact with the ground and landing perfectly.

The environment is also configured so that it learns to land in the landing pad area with both legs touching the ground, before the body touches the ground. Since the body touching the ground results in the end of the episode, we learn to prevent that by penalizing the agent for higher velocities.

D. PARAMETER SETTING

The parameter settings for the various approaches implemented in this article are displayed in the Table 2.

For environment-related parameters, we utilized the predefined settings from Gym. The observation size and action space size used are the default setting of the environments in Gym.

In the quantum circuit-related parameters, the number of qubits corresponded to the observation size, using 4 qubits for Frozen Lake and Cart Pole and 8 for Lunar Lander environments, facilitating effective state modeling. The number of layers was consistently set at 5 across all environments to balance expressiveness and computational efficiency.

For the remaining parameters, we experimented with various configurations to optimize performance. A higher learning rate (0.5) was chosen for Frozen Lake to enhance convergence speed, while smaller rates (0.001 to 0.005) were applied in more complex environments like Cart Pole, Lunar Lander for stability. The discount factor (γ) was uniformly set to 0.99 for all Policy based VQC approaches to prioritize long-term rewards. Other parameters, such as batch size, epsilon decay, and replay memory, were fine-tuned through experimentation to ensure effective learning and exploration.

E. COMPUTATIONAL RESOURCES

For the training of agents, both classical reinforcement learning (RL) and quantum reinforcement learning (Quantum RL) utilized distinct computational approaches but shared common resources.

For both classical RL and Quantum RL, we utilized the Python 3 Google Compute Engine backend available through Google Colab, which provided 12 GB RAM and GPU acceleration. This environment was essential for accelerating the training process. The training of both classical and quantum agents made use of the PyTorch and Gym libraries, providing a robust framework for reinforcement learning across both paradigms.

In the case of Quantum RL, we additionally used the PennyLane library to simulate quantum circuits and integrate quantum components with classical machine learning algorithms. The hybrid quantum-classical computations, performed using PennyLane, were executed on the same Colab environment, leveraging its computational resources to handle the quantum simulations.

For testing the quantum agents, we employed both the PennyLane Quantum Simulator and the IBMQ QASM Simulator

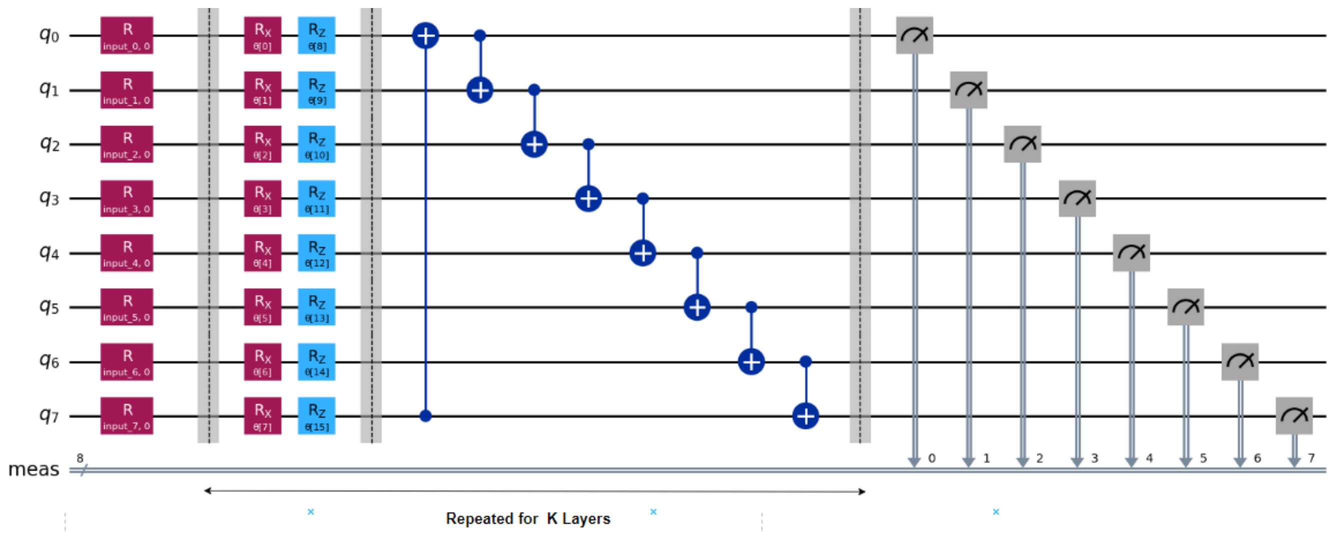


FIGURE 13. Quantum Circuit used for the Actor in Lunar lander environment. The circuit uses 8 qubits, which corresponds to the size of the observation obtained from the environment. The value of K chosen by us for our approach was 5.

TABLE 2. Parameter Setting for all the different approaches implemented in the article.

Sl No.	Parameter	Frozen Lake VQC Q Learning	Frozen Lake VQC Policy Based	Cart Pole VQC Policy Based	Lunar Lander VQC Policy Based	Lunar Lander A2C (Actor)
1	Observation Size	1	1	4	8	8
2	Qubits	4	4	4	8	8
3	Actions Space size	4	4	2	4	4
4	Classical Measurement Size	4	4	2	4	4
5	Layers	5	5	5	5	5
6	Learning Rate (α)	0.5	0.01	0.001	0.005	0.005
7	Discount Factor (γ)	0.99	0.99	0.99	0.99	0.99
8	Training Batch Size	-	100	16	16	-
9	Epsilon Decay	-	0.99	0.99	0.99	-
10	Replay Memory	-	10000	10000	10000	-

(IBM Quantum Assembly Language). The IBMQ QASM Simulator offers a maximum memory capacity of 64 GB, which was utilized for more memory-intensive quantum simulations. In these quantum simulations, 1024 shots were used for measurement. A shot refers to a single execution of a quantum circuit, and multiple shots were necessary to estimate the probability distribution of measurement outcomes due to the inherent probabilistic nature of quantum computing.

IV. RESULTS AND DISCUSSION

The below sections discuss the results and provide discussion on our implementation of QRL along with a discussion on the literature review.

A. FROZEN LAKE ENVIRONMENT

The below sections discuss the results obtained for QRL on Frozen Lake environment.

1) VQC WITH Q TABLE

For 5 testing episodes, it is observed that the agent has learned to reach the goal state in 6 steps, which is the minimum number of steps required to traverse through the Frozen Lake. Considering that the original Frozen Lake environment gives reward only after reaching the goal state, the reward of +1 indicates that the agent traverses through the environment successfully.

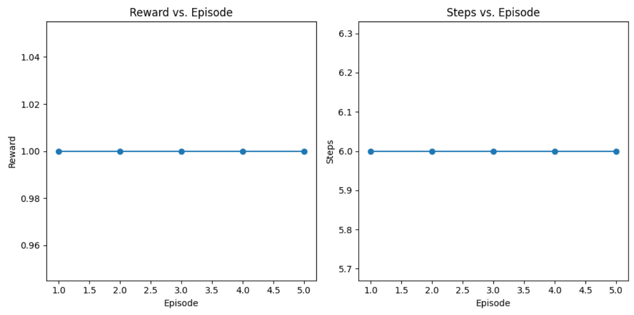
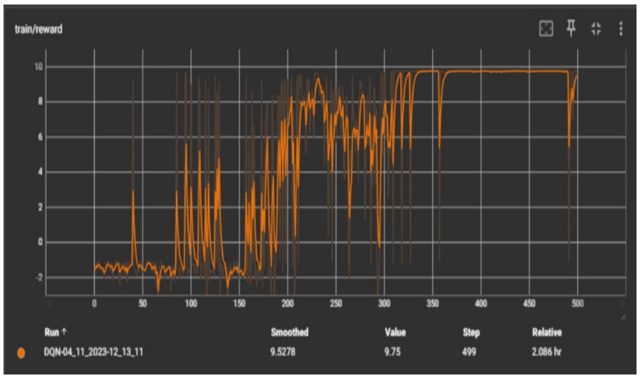


FIGURE 14. Q Learning with VQC Results for Frozen Lake Environment.

The Plots in Fig 14 show the testing results for the Q Learning setup for Frozen Lake

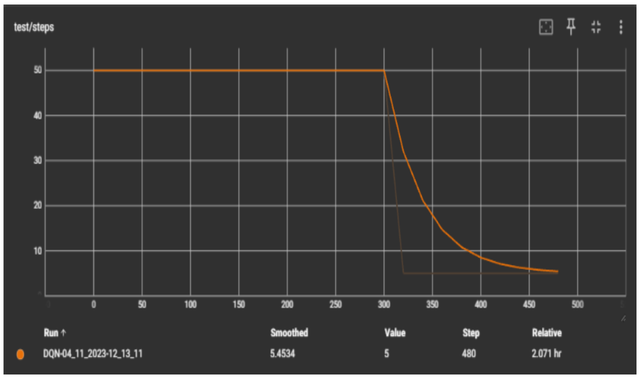
We observe that the agent has learned to reach the goal state in minimum number of steps (6) and obtain a reward of 1

Q-Learning is fairly easy to perform using the Q-Table, though it might require larger number of episodes. This is



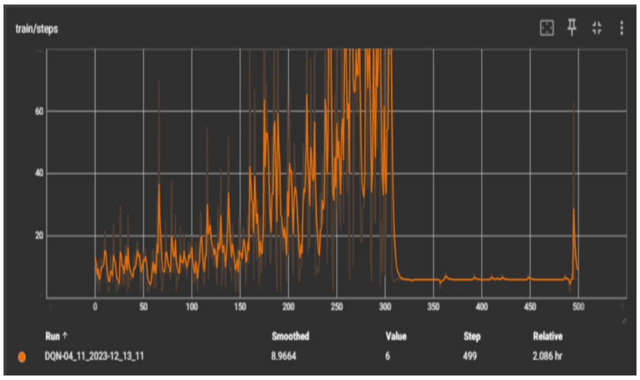
Run ↑	Smoothed	Value	Step	Relative
DQN-04_11_2023-12_13_11	9.5278	9.75	499	2.086 hr

FIGURE 15. Training reward vs episode plot for Frozen Lake.



Run ↑	Smoothed	Value	Step	Relative
DQN-04_11_2023-12_13_11	5.4534	5	480	2.071 hr

FIGURE 17. Test steps during training vs episode plot for Frozen Lake.



Run ↑	Smoothed	Value	Step	Relative
DQN-04_11_2023-12_13_11	8.9664	6	499	2.086 hr

FIGURE 16. Training steps vs episode plot for Frozen Lake.

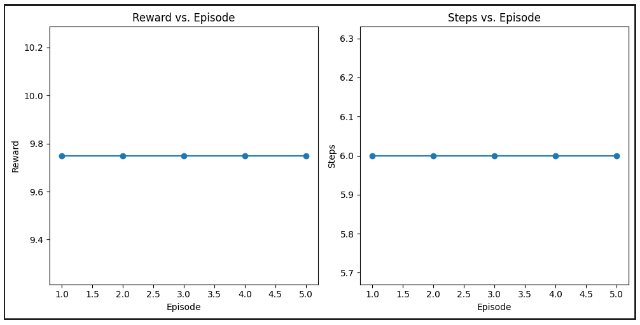


FIGURE 18. Testing phase results for Frozen Lake on Pennylane Quantum Simulator.

due to the fact that Q Learning is a model free approach. However, this method is not feasible for continuous state environments, because the memory required for Q Learning will be extremely large.

2) POLICY BASED METHOD USING VQC

This is performed for the redefined reward Frozen Lake environment, where the agent is penalized with negative rewards for states not leading to the goal. The plots represent both the training rewards and the testing rewards, where testing is done simultaneously with the training. From Figures 15. 16 and 17, we can observe that

- We observe that when we have trained the agent for 500 episodes, the model learns to give better rewards starting from about 175 episodes. This might be because the agent, in spite of lower epsilon value at this point, it has chosen exploration and reached the goal state. Following this, it tries to reach the goal multiple times, and eventually learns the best path.
- We observe that between episodes 175 and 300, the episodes last for larger number of steps, some taking up to 80 steps without reaching the destination. Also, just after 300 episodes, it tries to take more steps, implying that encountering the goal state was beneficial. The Agent is trying to search for Goal state again. Finally, the optimal number of steps is 6.
- From the Testing steps plot in 17, we observe that the agent eventually learns to reach the goal after 300 episodes of training. At the end, it manages to reach the destination using 6 steps.

Evaluation using Pennylane quantum simulator

We have used Pennylane quantum simulator to test the performance of the agent. Figure 18 shows the test steps and test rewards obtained during the evaluation phase.

We observe that the Agent performs optimally in 6 steps and obtains a reward of 9.75 for each test episode, for the Pennylane simulator.

Evaluation using IBMQ QASM simulator

We have used IBMQ Quantum simulator which uses Quantum Assembly Language to test the performance of the agent. Figure 19 shows the test steps and test rewards obtained during the evaluation phase for PennyLane simulator. The mean reward obtained is 9.75 and the number of steps taken is 6.

Using the IBM Quantum simulator, the agent is still able to reach the goal state every time, but it takes more steps than expected. The mean reward obtained is 9.6, and the mean number of steps taken is 9.

3) COMPARISON WITH CLASSICAL RESULTS

Since the goal of our project is to prove that one would require lesser number of parameters to train an RL agent

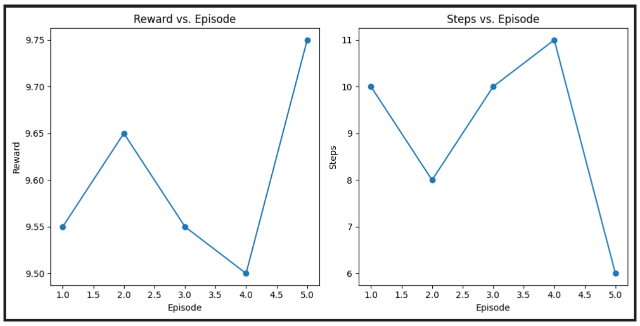


FIGURE 19. **Testing phase results for Frozen Lake on IBM Quantum Simulator.**

using Quantum Computation, it would make sense to compare it with the Classical equivalent of the Frozen Lake implementation. Table 3 below compares the number of trainable parameters in both the approaches

4) COMPARISON WITH EXISTING APPROACHES

Our approach differs from other approaches in the following ways:

- We have used a unique VQC circuit to model the Frozen Lake environment, as shown in the Fig 11.
- Comparing the number of parameters required to train the Agent with existing works, such as [15] and [41], we observe that our agent contains 40 trainable parameters.
- The article by Chen et al. [41], uses 28 parameters to obtain the results, where the training was done for 500 episodes in both the article and in our approach.
- The article by Moll et al. [15], uses 40 parameters to train the agent, but the runtime of the training is 21hrs, meanwhile our training process takes 2hrs to be trained.

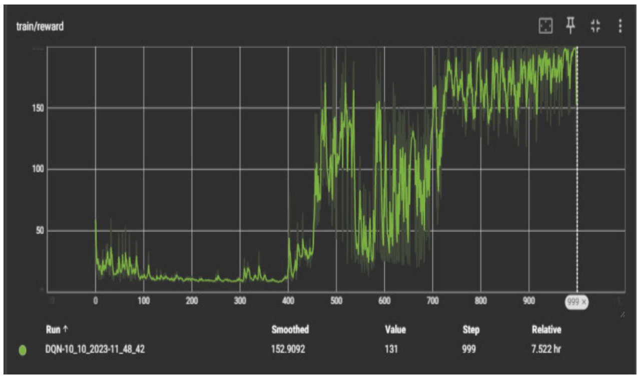
B. CART POLE ENVIRONMENT

The below sections discuss the results obtained for QRL on Cart Pole environment.

1) POLICY BASED METHOD USING VQC

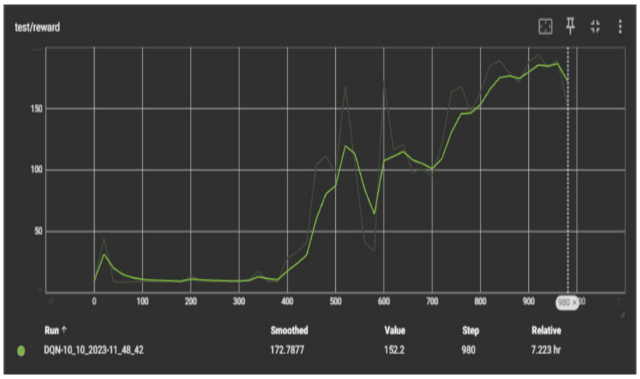
The Cart Pole environment uses the environment provided by gym, where the objective is to keep the pole upright. Figure 20 gives the following results for training phase. The plots represent both the training rewards and the testing rewards 21, where testing is done simultaneously with the training.

We can observe that this iteration performs really well. We can clearly see that it is slowly learning to get better rewards. The reward obtained in the last episode is 131, which is a good performance by the agent. Post 400 episodes, the Agent comes out of the flat region of rewards and tries to explore the unexplored possibilities. Between 500 and 700 episodes, we can observe the reward varies, and also it slowly learns to obtain better rewards.



```
Run ↑                      Smoothed   Value  Step  Relative
DQN-10_10_2023-11_48_42    152.9092   131    999   7.522 hr
```

FIGURE 20. **Training reward against number of episodes plot for Cart Pole.**



```
Run ↑                      Smoothed   Value  Step  Relative
DQN-10_10_2023-11_48_42    172.7877   152.2  980   7.223 hr
```

FIGURE 21. **Testing reward during the Training Process against number of episodes plot for Cart Pole.**

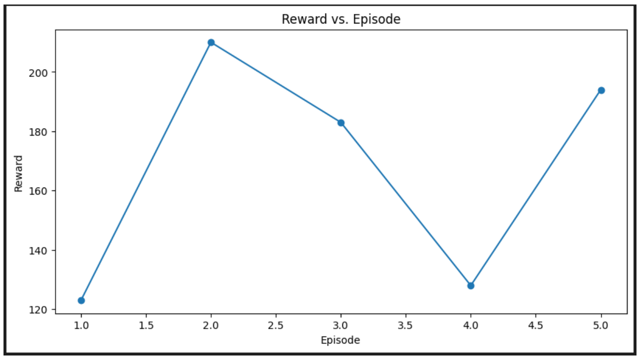


FIGURE 22. **Testing Phase rewards for Cart Pole on Pennylane Quantum Simulator.**

During the process of testing while training as in 21, we observe that the agent follows a similar trend compared to training rewards plot. The test reward at the last epoch is 152.2.

Figures 22 and 23 display the outputs for the evaluation phase.

Evaluation using Pennylane quantum simulator

We can observe that the rewards obtained are in excess of 100, reaching about 180 reward thrice for 5 episodes in testing phase. The mean reward is 167.6.

TABLE 3. Comparison of various approaches for RL on Frozen Lake. Here the first two approaches are classical approaches using different number of nodes in hidden layers. For a 4 × 4 frozen lake, the observation can be recorded using 4 bits, and the agent can perform 4 different actions. For the quantum case, as in Fig 11, we use 2 rotational gates per layer per qubit.

Model used	Calculation	Total parameters	Mean Reward
DQN with 3 layers, hidden = 16	4 * 16 + 16 * 16 + 16 * 4	384	9.75
DQN with 3 layers, hidden = 32	4 * 32 + 32 * 32 + 32 * 4	1280	9.75
Policy Based VQC-DQN with 5 layers	5 * 2 * 4	40	9.75

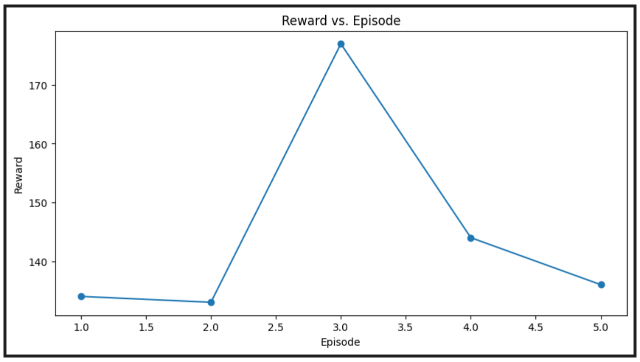


FIGURE 23. Testing Phase rewards for Cart Pole on IBM Quantum Simulator.

Evaluation using IBM QASM simulator

In the IBM quantum simulator, we observe that the model does not get as good results as compared to Pennylane simulator, with only one episode returning a reward greater than 170. The mean reward is 144.8.

2) COMPARISON WITH CLASSICAL RESULTS

Since the goal of our project is to prove that one would require lesser number of parameters to train an RL agent using Quantum Computation, it would make sense to compare it with the Classical equivalent of the Cart Pole implementation. Table 4 below compares the number of trainable parameters in both the approaches.

In the classical approaches, we observe for the first case, the mean reward obtained is 146. The average performance of our quantum agent is a little better compared to the first case, while using almost 90% less parameters to train the agent. The testing rewards for this approach can be viewed in Fig 24.

In the second classical implementation with hidden layer of size 64 nodes, the Mean reward obtained is 500, which is the maximum reward obtainable in the Cart Pole environment. The performance of the Quantum is much lesser than the the classical case. However, there is scope for improvement in performance by increasing the number of layers or gates in the quantum setup. The testing rewards for the classical implementation can be viewed in Fig 25.

3) COMPARISON WITH EXISTING APPROACHES

Comparing our approach with other approaches

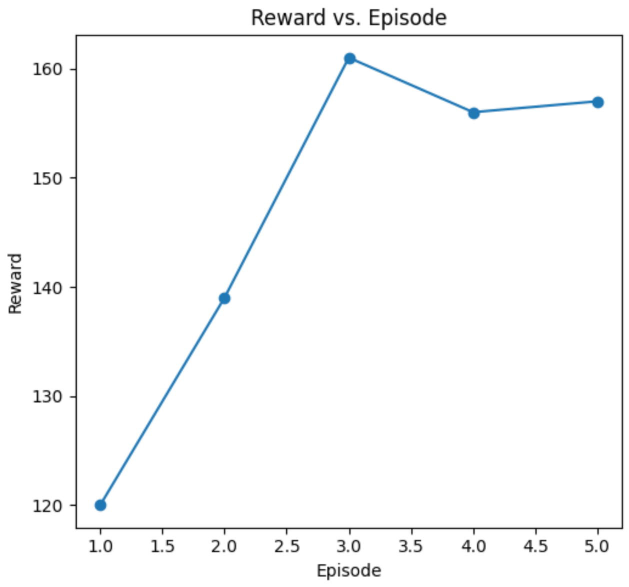


FIGURE 24. Testing Phase rewards for CartPole Classical Case 1.

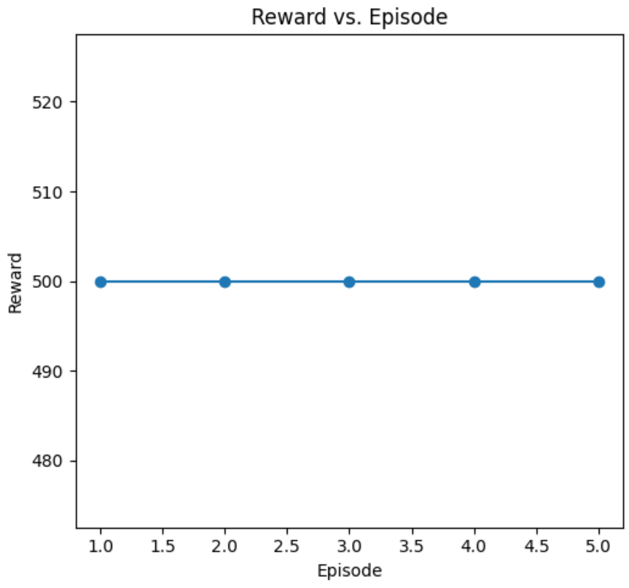
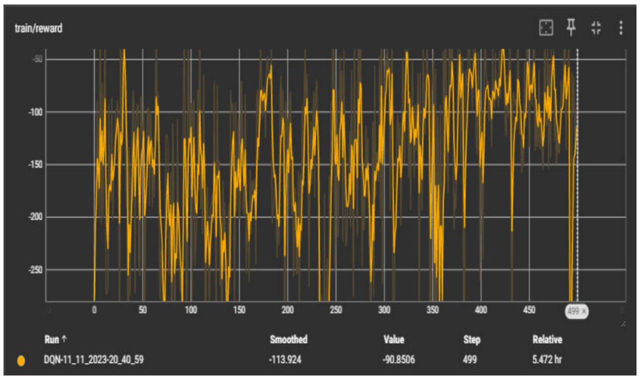


FIGURE 25. Testing Phase rewards for CartPole Classical Case 2.

- The VQC we have used to model the Cartpole environment is unique to our approach, as shown in Fig 11.
- The approach used in Lockwood et al. [43] implements an VQC agent with training parameters in the order of

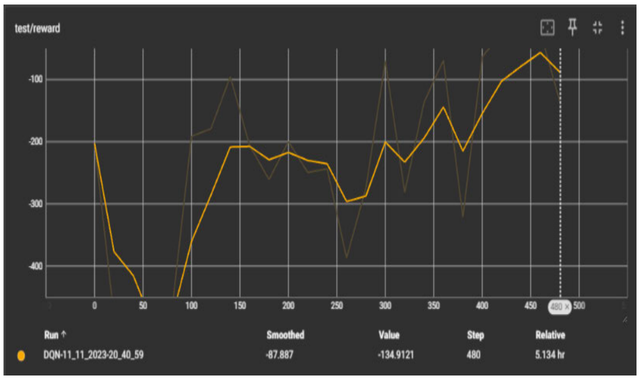
TABLE 4. Comparison of various approaches for RL on Cart Pole. Here the first two approaches are classical approaches using different number of nodes in hidden layers. For a Cart Pole, the observation size is 4, and the agent can perform 2 different actions. For the quantum case, as in Fig 11, we use 2 rotational gates per layer per qubit.

Model used	Calculation	Total parameters	Mean Reward
DQN with 3 layers, hidden = 32	4 * 32 + 32 * 32 + 32 * 2	1216	146
DQN with 3 layers, hidden = 64	4 * 64 + 64 * 64 + 64 * 2	4480	500
Policy Based VQC-DQN with 5 layers	5 * 2 * 4	40	168



	Run ↑	Smoothed	Value	Step	Relative
●	DQN-11_11_2023-20_40_59	-113.924	-90.8506	499	5.472 hr

FIGURE 26. Training Phase rewards against number of episodes plot for Lunar Lander.



	Run ↑	Smoothed	Value	Step	Relative
●	DQN-11_11_2023-20_40_59	-87.887	-134.9121	480	5.134 hr

FIGURE 27. Testing Phase rewards while training against number of episodes plot for Lunar Lander.

first power of 10. The agent is trained for 1000 episodes and the maximum reward obtained is 125, which means that the pole was upright for 125 steps.

- Our approach uses 40 parameters, which is comparable to the approach by [43]. Our agent is trained for 1000 episodes and the average rewards obtained is 144.8.

C. LUNAR LANDER ENVIRONMENT

The below sections discuss the results obtained for QRL on Lunar Lander environment.

1) POLICY BASED METHOD USING VQC

This method uses the default environment provided by gym module. Figure 26 shows the training phase results, and figure 27 displays the testing rewards obtained during the process of training.

The training reward during the last episode is −90.85. This might be due to the complexity of the problem, the fact that we were using 8 qubits and 3 rotation gates per layer per qubit. This might have resulted in an unstable learning, considering the fact that 8 qubits measurements result in a probability distribution of 256 values.

Looking at the figure 27, we observe that the testing rewards keeps increasing, but the reward does not seem to increase beyond 0. Due to the complexity of an eight-qubit circuit, the agent does not seem to learn to land properly in the flagged region.

The agent is not able to learn very well this environment using this method. The lander almost never lands perfectly. The lunar lander environment also gives better rewards only for perfect landing.

2) ADVANTAGE ACTOR CRITIC USING VQC

This method uses the redefined environment, where we have reconfigured it to obtain better training of the agent. It differs from ordinary environment in two aspects

- Penalty for high vertical velocity
- Episode ends if the Lander lands with both legs in landing pad.

The assumption is that the Lander will learn to land properly with a lower vertical velocity.

Figure 28 shows the plot for the training phase

As we can observe for 1000 episodes, the combination of Quantum Actor and Classical Critic seems to do very well, starting from −600 reward, and obtaining rewards greater than 200. The main reason for this could be the Classical Critic, which provides a good value for the purpose of parameter updates. We can observe the testing results in Figures 29 and 30

We tested both policy based VQC-DQN and hybrid A2C models for Lunar Lander. Due to the complexity of the environment, VQC-DQN did not perform well, leading us to choose hybrid A2C for improved stability and convergence. However, for other environments, policy based VQC-DQN was sufficient and demonstrated good performance, so A2C was not used.

Evaluation using Pennylane quantum simulator

We can observe that the for 5 episodes, 3 of them land successfully, and the other two gain a small negative reward.

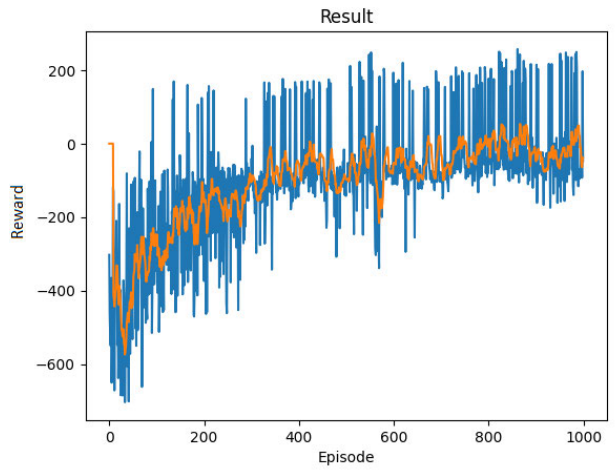


FIGURE 28. Training Phase rewards against number of episodes plot for Lunar Lander.

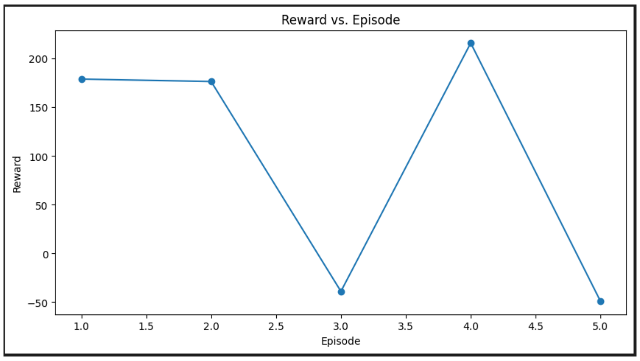


FIGURE 29. Testing Phase results for Lunar Lander on Pennylane quantum simulator.

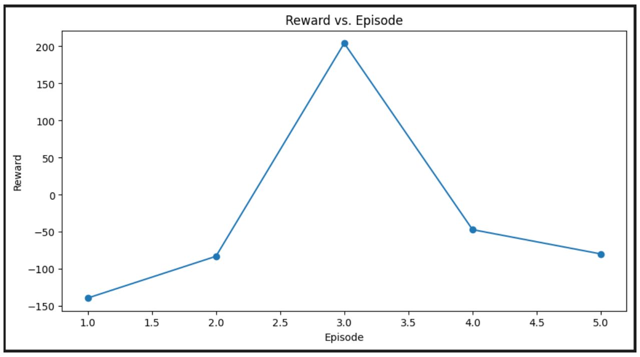


FIGURE 30. Testing Phase results for Lunar Lander on IBM quantum simulator.

This could mean that they did not crash-land and also landed outside the landing zone. The model performs fairly well when evaluated. The mean reward is 96.73.

Evaluation using IBM quantum simulator

On the IBM simulator, we obtain negative rewards for most of the episodes. Only once does it land perfectly on Episode 3. The first two episodes seem to result in crash of the lander, where the last two seem to land out of the landing pad. The mean reward is −28.87.

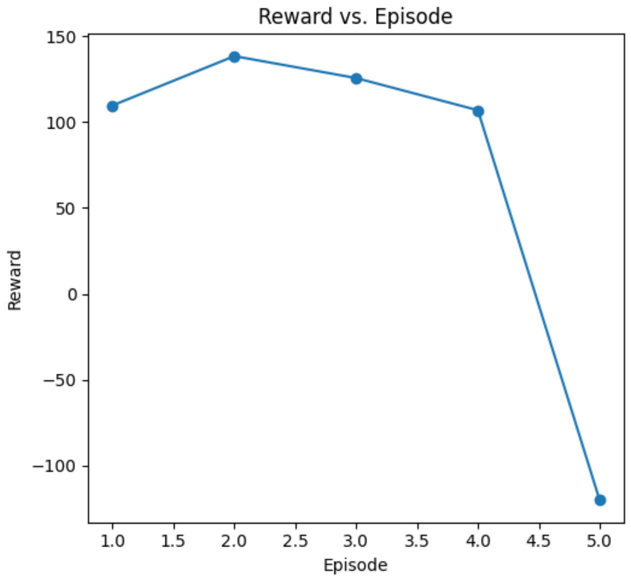


FIGURE 31. Testing phase rewards for lunar lander classical case.

3) COMPARISON WITH CLASSICAL RESULTS

Since the goal of our project is to prove that one would require lesser number of parameters to train an RL agent using Quantum Computation, it would make sense to compare it with the Classical equivalent of the Lunar Lander implementation for A2C method. The test rewards obtained from the classical approach can be observed in the Fig 31. Table 5 below compares the number of trainable parameters in both the approaches

As we observe here, the Hybrid Quantum Actor and Classical Critic approach gives a better performance compared to the classical approach with lesser number of parameters. This is a result of using qubits and entanglement in the quantum circuit. The Classical Critic also helps for us to obtain the value quickly so that parameters of Actor can be improved upon.

4) COMPARISON WITH EXISTING APPROACHES

An A2C approach using VQC is first suggested by our article for solving the Lunar Lander environment. This has to do with a complexity of Lunar lander having a state size of 8 and due to it being a continuous environment with 4 possible actions. The circuit for the same can be seen in Fig 13.

The article Hsiao et al. [5] proposed a single qubit gates approach using 124 parameters (24 for angle and 100 for output scaling). Compared to this, our Actor VQC circuit uses 80 parameters to train the agent.

D. LIMITATIONS AND CHALLENGES

We observed that, when testing the agent on the IBM Quantum Simulator, the mean reward across episodes and the agent's consistency in reaching the goal state in the Lunar

TABLE 5. Comparison of the performance and trainable parameters between the Pure Classical A2C and Hybrid A2C. For the classical actor and critic in both cases, there are 2 hidden layers of 128 and 256 nodes respectively. However critic returns a single value, meanwhile the classical actor returns a tensor of size 4 since there are four possible actions. For the quantum actor, as shown in Fig 13 we use 2 rotational gates per layer per qubit.

Model used	Actor Parameters	Critic Parameters	Total Parameters	Mean Reward
Pure Classical A2C	34816	34048	68864	72.08
Hybrid A2C	80	34048	34128	96.73

Lander environment were notably lower compared to the Pennylane Quantum Simulator. A similar trend is expected when testing the agent on the IBM Quantum Computer. One of the primary challenges is the susceptibility of current quantum systems to noise and high error rates during computations. Near-term quantum devices are particularly vulnerable to decoherence and gate errors, which can significantly degrade the performance of quantum algorithms, leading to inaccurate results. This is especially critical in reinforcement learning, where precision is essential for effective policy optimization and decision-making.

To address these limitations, various noise reduction and error mitigation techniques have been developed. Methods such as Probabilistic Error Cancellation (PEC), Zero Noise Extrapolation (ZNE), Matrix-free Measurement Mitigation (M3), and Twirled Readout Error eXtinction (TREX) show promise in reducing errors during computation. However, each of these approaches introduces its own overheads and varying levels of accuracy, which must be carefully considered in practical implementations [73], [74], [75], [76], [77].

V. CONCLUSION

In conclusion, the article delves into the exciting realm of Quantum Reinforcement Learning (QRL), spotlighting its potential to revolutionize machine learning and optimization processes. By exploring the confluence of quantum computing and reinforcement learning, we have unearthed both the promise and challenges inherent in this cutting-edge field.

Throughout the literature review, we have identified and discussed key QRL concepts, methodologies, prominent algorithms, and potential applications in various domains. While challenges such as quantum hardware limitations persist, collaboration among researchers from various disciplines is pivotal in harnessing QRL's transformative potential. As quantum technology continues to advance, these barriers may diminish, paving the way for more extensive practical applications of QRL in the future.

In the Frozen Lake environment, both the standard Q-learning with VQC and the Policy Based VQC DQN methods give good optimal results, solving the environment in minimal number of steps (being 6) and achieving the maximum possible reward (which is 9.75). This was both in the original environment and the reward shaped modified environment. The Q-Table in Q-learning being model free performs well, meanwhile the Policy Based VQC DQN method consumes fewer parameters compared to the classical analogues.

In the Cart Pole environment, the method of Policy Based VQC DQN method gave good results, achieving rewards in excess of 150. This means that the agent has learned to move the Cart in such a way that the Pole remains upright for a longer period of time. Even though the environment is a continuous state environment, the results seem to be very good for the number of gates used and the number of trainable parameters.

In the Lunar Lander environment, using Policy Based VQC DQN was not very yielding. The agent failed to make the Lander land properly in the landing pad region without crashing. This resulted in negative rewards, which is definitely not a good method for this environment. On the redefined environment, using the Hybrid A2C gave very good rewards, crossing even 200 and also consumed very few parameters compared to the Pure A2C. The majority contributor of this would be using a Classical Critic and due to the Quantum Actor using lesser parameters.

This study demonstrates the benefits of Variational Quantum Circuits (VQC) in Deep Q-Networks (DQN), particularly in reducing the number of trainable parameters. Future research could focus on integrating metaheuristic algorithms—such as Genetic Algorithms or Particle Swarm Optimization—to optimize quantum circuit parameters in Quantum Reinforcement Learning (QRL), potentially improving convergence and efficiency. Additionally, exploring the application of VQC with other reinforcement learning frameworks, like Asynchronous Advantage Actor-Critic (A3C) and Proximal Policy Optimization (PPO), may uncover new advantages and further reduce computational complexity. Another important direction is developing robust QRL models that can handle quantum hardware noise and errors, which is crucial as quantum computing systems advance. Implementing quantum error mitigation techniques could enhance the reliability of QRL.

The broader implications of these advancements extend complex real life environment fields such as finance, healthcare, and automation. In finance, QRL could optimize portfolio management, trading strategies, and risk assessment by efficiently processing large datasets and enabling quicker decision-making. In healthcare, quantum-enhanced simulations may accelerate drug discovery and personalized medicine through optimized molecular interactions. Similarly, automation industries, including robotics and autonomous systems, stand to benefit from faster and more accurate decision-making powered by QRL. Lastly, in optimization tasks like logistics and resource management, QRL could provide enhanced solutions for complex, large-scale problems.

VI. DECLARATION OF COMPETING INTERESTS

The authors confirm that there are no identifiable financial or interpersonal conflicts that could have impacted the research presented in this study.

VII. CODE AVAILABILITY

To explore the concepts of Reinforcement Learning and Quantum Computing further, along with our detailed literature review and the source code for our implementation, kindly refer the GitHub repository [31].

REFERENCES

[1] S. Chen, J. Cotler, H.-Y. Huang, and J. Li, "The complexity of NISQ," *Nature Commun.*, vol. 14, no. 1, p. 6001, Sep. 2023.

[2] A. Abbas et al., "Challenges and opportunities in quantum optimization," 2023, *arXiv:2312.02279*.

[3] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.

[4] A. Acuto, P. Barillà, L. Bozzolo, M. Conterno, M. Pavese, and A. Policicchio, "Variational quantum soft actor-critic for robotic arm control," 2022, *arXiv:2212.11681*.

[5] J.-Y. Hsiao, Y. Du, W.-Y. Chiang, M.-H. Hsieh, and H.-S. Goan, "Unentangled quantum reinforcement learning agents in the OpenAI gym," 2022, *arXiv:2203.14348*.

[6] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, no. 1, pp. 237–285, Jan. 1996.

[7] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[8] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien, "Quantum computers," *Nature*, vol. 464, no. 7285, pp. 45–53, Mar. 2010.

[9] S. Y.-C. Chen, C.-M. Huang, C.-W. Hsing, H.-S. Goan, and Y.-J. Kao, "Variational quantum reinforcement learning via evolutionary optimization," *Mach. Learn., Sci. Technol.*, vol. 3, no. 1, Feb. 2022, Art. no. 015025.

[10] Q. Liu, H. Wu, and A. Liu, "Modeling and interpreting real-world human risk decision making with inverse reinforcement learning," 2019, *arXiv:1906.05803*.

[11] Y. Kwak, W. J. Yun, S. Jung, J.-K. Kim, and J. Kim, "Introduction to quantum reinforcement learning: Theory and PennyLane-based implementation," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2021, pp. 416–420.

[12] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms," *New J. Phys.*, vol. 18, no. 2, Feb. 2016, Art. no. 023023.

[13] D. Wecker, M. B. Hastings, and M. Troyer, "Progress towards practical quantum variational algorithms," *Phys. Rev. A*, vol. 92, no. 4, Oct. 2015, Art. no. 042303.

[14] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, A. Kandala, A. Mezzacapo, P. Müller, W. Riess, G. Salis, J. Smolin, I. Tavernelli, and K. Temme, "Quantum optimization using variational algorithms on near-term quantum devices," *Quantum Sci. Technol.*, vol. 3, no. 3, Jul. 2018, Art. no. 030503.

[15] M. Moll and L. Kunczik, "Comparing quantum hybrid reinforcement learning to classical methods," *Hum.-Intell. Syst. Integr.*, vol. 3, no. 1, pp. 15–23, Mar. 2021.

[16] D. Dong, C. Chen, and Z. Chen, "Quantum reinforcement learning," in *Proc. 1st Int. Conf. Adv. Natural Comput. (ICNC)*, vol. 38, Changsha, China, Aug. 2005, pp. 686–689.

[17] J. D. Martín-Guerrero and L. Lamata, "Reinforcement learning and physics," *Appl. Sci.*, vol. 11, no. 18, Sep. 2021.

[18] I. Khalid, C. A. Weidner, E. A. Jonckheere, S. G. Schirmer, and F. C. Langbein, "Reinforcement learning vs. Gradient-based optimisation for robust energy landscape control of spin-1/2 quantum networks," in *Proc. 60th IEEE Conf. Decis. Control (CDC)*, Dec. 2021, pp. 4133–4139.

[19] V. V. Sivak, A. Eickbusch, H. Liu, B. Royer, I. Tsioutsios, and M. H. Devoret, "Model-free quantum control with reinforcement learning," *Phys. Rev. X*, vol. 12, no. 1, Mar. 2022, Art. no. 011059.

[20] T. Simonini and O. Sanseviero. (2023). *The Hugging Face Deep Reinforcement Learning Class*. [Online]. Available: https://github.com/huggingface/deep-rl-class

[21] N. Meyer, C. Ufrecht, M. Periyasamy, D. D. Scherer, A. Plinge, and C. Mutschler, "A survey on quantum reinforcement learning," 2022, *arXiv:2211.03464*.

[22] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, pp. 279–292, May 1992.

[23] R. Horodecki, P. Horodecki, M. Horodecki, and K. Horodecki, "Quantum entanglement," *Rev. Mod. Phys.*, vol. 81, p. 865, Jun. 2009.

[24] D. Voorhoede, "Quantum inspire," Qutech Delft Univ. Technol., Fac. Appl. Sci., Delft, The Netherlands, Tech. Rep., 2018.

[25] X. Yuan, S. Endo, Q. Zhao, Y. Li, and S. C. Benjamin, "Theory of variational quantum simulation," *Quantum*, vol. 3, p. 191, Oct. 2019.

[26] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, pp. 195–202, Sep. 2017.

[27] L. Lamata, "Quantum machine learning and quantum biomimetics: A perspective," *Mach. Learning: Sci. Technol.*, vol. 1, no. 3, Sep. 2020, Art. no. 033002.

[28] S. Y.-C. Chen, "Quantum deep recurrent reinforcement learning," 2022, *arXiv:2210.14876*.

[29] Y. Kwak, W. J. Yun, J. P. Kim, H. Cho, J. Park, M. Choi, S. Jung, and J. Kim, "Quantum distributed deep learning architectures: Models, discussions, and applications," *ICT Exp.*, vol. 9, no. 3, pp. 486–491, Jun. 2023.

[30] V. Dunjko, J. M. Taylor, and H. J. Briegel, "Advances in quantum reinforcement learning," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, pp. 282–287.

[31] N. R. Bhat, N. Nandakumar, S. Sagar, Sunil, R. Ravish, and P. B. Honnavalli. *Optimization of Reinforcement Learning Using Quantum Computation*. Accessed: Nov. 27, 2024. [Online]. Available: https://github.com/NischalRBhat/Optimization-of-Reinforcement-Learning-using-Quantum-Computation

[32] C. Mao, "Reinforcement learning with blackjack," PhD thesis, California State University, Northridge, Los Angeles, CA, USA, 2019.

[33] P. Gawłowicz and A. Zubow, "Ns-3 meets OpenAI gym: The playground for machine learning in networking research," in *Proc. 22nd Int. ACM Conf. Model., Anal. Simul. Wireless Mobile Syst.*, Nov. 2019, pp. 113–120.

[34] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vols. SMC–13, no. 5, pp. 834–846, Sep. 1983.

[35] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI gym," 2016, *arXiv:1606.01540*.

[36] T.-A. Drăgan, M. Monnet, C. B. Mendl, and J. Miriam Lorenz, "Quantum reinforcement learning for solving a stochastic frozen lake environment and the impact of quantum architecture choices," 2022, *arXiv:2212.07932*.

[37] M. Chevalier-Boisvert, B. Dai, M. Towers, R. de Lazcano, L. Willems, S. Lahlou, S. Pal, P. Samuel Castro, and J. Terry, "Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks," 2023, *arXiv:2306.13831*.

[38] Z. Cheng, K. Zhang, L. Shen, and D. Tao, "Offline quantum reinforcement learning in a conservative manner," in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, 2023, pp. 7148–7156.

[39] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, "Variational quantum algorithms," *Nature Rev. Phys.*, vol. 3, no. 9, pp. 625–644, 2021.

[40] M. Ostaszewski, L. M. Trenkwalder, W. Masarczyk, E. Scerri, and V. Dunjko, "Reinforcement learning for optimization of variational quantum circuit architectures," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 18182–18194.

[41] S. Y. Chen, C. H. Yang, J. Qi, P.-Y. Chen, X. Ma, and H.-S. Goan, "Variational quantum circuits for deep reinforcement learning," *IEEE Access*, vol. 8, pp. 141007–141024, 2020.

[42] Z. Wang, "Quantum control based on deep reinforcement learning," 2022, *arXiv:2212.07385*.

[43] O. Lockwood and M. Si, "Reinforcement learning with quantum variational circuits," 2020, *arXiv:2008.07524*.

[44] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, and P. Balaprakash, "Reinforcement-learning-based variational quantum circuits optimization for combinatorial problems," 2019, *arXiv:1911.04574*.

[45] Y. J. Patel, S. Jerbi, T. Bäck, and V. Dunjko, "Reinforcement learning assisted recursive QAOA," 2022, *arXiv:2207.06294*.

[46] T. Fösel, M. Y. Niu, F. Marquardt, and L. Li, "Quantum circuit optimization with deep reinforcement learning," 2021, *arXiv:2103.07585*.

[47] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices," *Phys. Rev. X*, vol. 10, no. 2, Jun. 2020, Art. no. 021067.

[48] E. Andrés, M. P. Cuéllar, and G. Navarro, "On the use of quantum reinforcement learning in energy-efficiency scenarios," *Energies*, vol. 15, no. 16, p. 6034, Aug. 2022.

[49] A. Kumari and S. Tanwar, "AI-based peak load reduction approach for residential buildings using reinforcement learning," in *Proc. Int. Conf. Comput., Commun., Intell. Syst. (ICCCIS)*, Feb. 2021, pp. 972–977.

[50] A. Kumari and S. Tanwar, "Reinforcement learning for multiagent-based residential energy management system," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2021, pp. 1–6.

[51] A. Kumari, R. Kakkar, S. Tanwar, D. Garg, Z. Polkowski, F. Alqahtani, and A. Tolba, "Multi-agent-based decentralized residential energy management using deep reinforcement learning," *J. Building Eng.*, vol. 87, Jun. 2024, Art. no. 109031.

[52] A. Kumari and S. Tanwar, "A reinforcement-learning-based secure demand response scheme for smart grid system," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2180–2191, Feb. 2022.

[53] A. Kumari, M. Trivedi, S. Tanwar, G. Sharma, and R. Sharma, "SV2G-ET: A secure vehicle-to-grid energy trading scheme using deep reinforcement learning," *Int. Trans. Electr. Energy Syst.*, vol. 2022, pp. 1–11, Apr. 2022.

[54] E. Andrés, M. P. Cuéllar, and G. Navarro, "Efficient dimensionality reduction strategies for quantum reinforcement learning," *IEEE Access*, vol. 11, pp. 104534–104553, 2023.

[55] A. Skolik, S. Mangini, T. Bäck, C. Macchiavello, and V. Dunjko, "Robustness of quantum reinforcement learning under hardware errors," *EPJ Quantum Technol.*, vol. 10, no. 1, pp. 1–43, Dec. 2023.

[56] A. Dua, T. Jochym-O'Connor, and G. Zhu, "Quantum error correction with fractal topological codes," *Quantum*, vol. 7, p. 1122, Sep. 2023.

[57] S. Jerbi, C. Gyurik, S. C. Marshall, H. J. Briegel, and V. Dunjko, "Parametrized quantum policies for reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 1–14.

[58] V. Bergholm et al., "PennyLane: Automatic differentiation of hybrid quantum-classical computations," 2018, *arXiv:1811.04968*.

[59] H. Hohenfeld, D. Heimann, F. Wiebe, and F. Kirchner, "Quantum deep reinforcement learning for robot navigation tasks," 2022, *arXiv:2202.12180*.

[60] D. Singh, E. Trivedi, Y. Sharma, and V. Niranjan, "TurtleBot: Design and hardware component selection," in *Proc. Int. Conf. Comput., Power Commun. Technol. (GUCON)*, Sep. 2018, pp. 805–809.

[61] S. Y. Chen, S. Yoo, and Y. L. Fang, "Quantum long short-term memory," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 8622–8626.

[62] C. Jiang, Y. Pan, Z.-G. Wu, Q. Gao, and D. Dong, "Robust optimization for quantum reinforcement learning control using partial observations," *Phys. Rev. A, Gen. Phys.*, vol. 105, no. 6, pp. 1–14, Jun. 2022.

[63] S. Wu, S. Jin, D. Wen, D. Han, and X. Wang, "Quantum reinforcement learning in continuous action space," 2020, *arXiv:2012.10711*.

[64] H. van Hasselt and M. A. Wiering, "Reinforcement learning in continuous action spaces," in *Proc. IEEE Int. Symp. Approx. Dyn. Program. Reinforcement Learn.*, Apr. 2007, pp. 272–279.

[65] G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, T. Mann, T. Weber, T. Degris, and B. Coppin, "Deep reinforcement learning in large discrete action spaces," 2015, *arXiv:1512.07679*.

[66] Z. T. Wang, Y. Ashida, and M. Ueda, "Deep reinforcement learning control of quantum cartpoles," *Phys. Rev. Lett.*, vol. 125, no. 10, pp. 1–10, Sep. 2020.

[67] F. A. Cárdenas-López, L. Lamata, J. C. Retamal, and E. Solano, "Multiqubit and multilevel quantum reinforcement learning with quantum technologies," *PLoS ONE*, vol. 13, no. 7, Jul. 2018, Art. no. e0200455.

[68] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, "Reinforcement learning with neural networks for quantum feedback," *Phys. Rev. X*, vol. 8, no. 3, pp. 1–15, Sep. 2018.

[69] S. J. Devitt, W. J. Munro, and K. Nemoto, "Quantum error correction for beginners," *Rep. Prog. Phys.*, vol. 76, no. 7, Jul. 2013, Art. no. 076001.

[70] V. Saggio, B. E. Asenbeck, A. Hamann, T. Strömberg, P. Schiansky, V. Dunjko, N. Friis, N. C. Harris, M. Hochberg, D. Englund, S. Wölk, H. J. Briegel, and P. Walther, "Experimental quantum speed-up in reinforcement learning agents," *Nature*, vol. 591, no. 7849, pp. 229–233, Mar. 2021.

[71] E. A. Cherrat, I. Kerenidis, and A. Prakash, "Quantum reinforcement learning via policy iteration," *Quantum Mach. Intell.*, vol. 5, no. 2, pp. 1–30, Dec. 2023.

[72] P. Peng, X. Huang, C. Yin, L. Joseph, C. Ramanathan, and P. Cappellaro, "Deep reinforcement learning for quantum Hamiltonian engineering," *Phys. Rev. Appl.*, vol. 18, no. 2, pp. 1–17, Aug. 2022.

[73] W. Shi, N. K. Kundu, and R. Malaney, "Error-mitigated multi-layer quantum routing," 2024, *arXiv:2409.14632*.

[74] M. U. Khan, M. A. Kamran, W. R. Khan, M. M. Ibrahim, M. U. Ali, and S. W. Lee, "Error mitigation in the NISQ era: Applying measurement error mitigation techniques to enhance quantum circuit performance," *Mathematics*, vol. 12, no. 14, p. 2235, Jul. 2024.

[75] *Differences in Error Suppression, Mitigation, and Correction | IBM Quantum Computing Blog — Ibm.com*. Accessed: Sep. 29, 2024. [Online]. Available: https://www.ibm.com/quantum/blog/quantum-error-suppression-mitigation-correction

[76] *Exploring Error Mitigation With the Mthree Qiskit Extension | IBM Quantum Computing Blog — Ibm.com*. Accessed: Sep. 29, 2024. [Online]. Available: https://www.ibm.com/quantum/blog/mthree-qiskit-extension

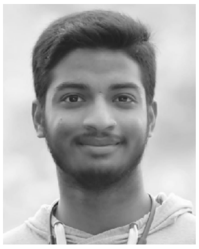
[77] *Landmark IBM Error Correction Paper on Nature Cover | IBM Quantum Computing Blog — Ibm.com*. Accessed: Sep. 29, 2024. [Online]. Available: https://www.ibm.com/quantum/blog/nature-qldpc-error-correction

ROOPA RAVISH received the bachelor's degree in instrumentation technology from Kuvempu University and the master's and Ph.D. degrees in computer science and engineering from Visvesvaraya Technological University, India.

She is currently an Associate Professor with the Department of Computer Science and Engineering, PES University, Bengaluru, India. Her research interests include intelligent transport systems, reinforcement learning, machine learning, the IoT, and automotive cybersecurity.

NISCHAL R. BHAT is currently pursuing the B.Tech. degree in computer science with PES University, Bengaluru, India.

In Summer 2023, he interned with the Center for Data Sciences and Applied Machine Learning (CDSAML), PES University. He has participated in IBM Quantum Challenges, in 2022 and 2023, and the IBM Quantum Summer School, in 2023, where he was awarded with certification. His research interests include reinforcement learning, quantum computing, graph machine learning, and cloud computing.



N. NANDAKUMAR is currently pursuing the B.Tech. degree in computer science with PES University, Bengaluru, India.

He recently demonstrated his abilities with the IBM Quantum-Hosted Qiskit Global Summer School, in 2023, where he was awarded with a Certificate and Badge. His research interests include reinforcement learning, quantum computing, and also in augmented and virtual reality (ARVR).

S. SAGAR is currently pursuing the B.Tech. degree in computer science with PES University, Bengaluru, excelled at the Qiskit Global Summer School 2023 by IBM Quantum.

He earned a Certificate and the prestigious Badge of Quantum Excellence. His research interests include quantum computing and reinforcement learning, reflecting his passion for cutting-edge technologies.

SUNIL is currently pursuing the B.Tech. degree in computer science and engineering with PES University, Bengaluru, India.

He recently showcased his talent with the Qiskit Global Summer School 2023, organized by IBM Quantum, earning both a Certificate and the prestigious Badge of Quantum Excellence. His research interests include quantum computing and reinforcement learning, reflecting his passion for exploring cutting-edge technologies.

PRASAD B. HONNAVALLI (Member, IEEE) received the B.E. degree from UVCE, Bangalore University, and the M.B.A. degree from The University of Melbourne, Australia.

He is currently a Professor with PES University, Bengaluru, India. He is also the Director of the PESU Centre for Information Security, Forensics and Cyber Resilience (C-ISFCR) and the PESU Centre for Internet of Things, with a focus on security (C-IoT). He leads the teaching, research, executive education, and industry partnership; and a consultancy in the areas of focus. He has authored or co-authored a number of research papers in leading journals and conferences. His research interests include information security, such as network and cloud security, software and web security, mobile application security, cryptography and blockchain, the IoT, SCADA, and industry 4.0.