Bridging Quantum Computing and Reinforcement Learning: A Comparative Study of Quantum and Classical Reinforcement Learning Algorithms in Cartpole Environment

Sakthignanavel Palanisamy

School of Physics and Astronomy
University of Nottingham

18/09/2023

Abstract

This dissertation explores the exciting frontier of exploring the capabilities of the Quantum RL algorithms and conducting a comparative analysis between classical and quantum versions of prominent Reinforcement Learning(RL) algorithms. Utilizing the well-known cart-pole environment under the gymnasium library, the project investigates three widely-used RL algorithms: policy gradient reinforce agent, deep Q-learning agent, and actor-critic agent. The study employs both classical versions and their quantum counterparts using reuploading Quantum Parametrized Circuits(QPC), offering a novel perspective on the capabilities and limitations of QRL. Through rigorous experimental designs, insightful comparisons, and comprehensive analyses, the research aims to uncover the efficiency, performance, and complexities of quantum and classical approaches. The findings contribute to the growing understanding of QRL and have significant implications for the broader field of quantum computing and artificial intelligence.

Contents

List of Figures

List of Tables

1 Introduction

Reinforcement learning(RL) has emerged as a powerful tool in artificial intelligence, enabling agents to learn from the environment and optimize behavior. However, the continual quest for efficiency and capability has led to the exploration of quantum computing, which offers exponential advantages over classical computing. Quantum Reinforcement Learning(QRL) utilizes the principles of quantum mechanics to achieve these advantages. This study embarks on a comprehensive exploration of QRL, focusing on a comparative analysis between classic and quantum algorithms within the context of the cart-pole environment.

1.1 Background and Motivation

Quantum computing leverages the unique properties of quantum bits (qubits) to perform computations at scales and speeds unattainable by classical computers. The superposition and entanglement of qubits enable quantum parallelism, allowing the solution of problems that are infeasible for classical computers. This paradigm shift offers an opportunity for enhancing RL algorithms, a potential that has barely been tapped. In the burgeoning field of QRL, there is a need for empirical evidence and theoretical understanding that contrasts quantum and classical approaches. The cart-pole balancing problem, a classical benchmark in RL, provides an excellent platform for this comparison due to its balanced complexity and clarity of objectives.

1.2 Objectives

The primary objectives of this study are:

- To implement classical and quantum versions of three RL algorithms: policy gradient reinforce agent, deep Q-learning agent, and actor-critic agent.
- To perform a comparative analysis of the efficiency, accuracy, and complexities of the classical and quantum versions within the cart-pole environment.
- To perform a sensitivity analysis on the Quantum RL agents by varying different parameters like number of layers in the QPC and studying how that affects the performance.
- To provide insights into the practical and theoretical implications of QRL.

1.3 Literature Review

The exploration of quantum algorithms has witnessed significant strides in recent years fueled by groundbreaking contributions like Shor's algorithm and Grover's algorithm which demonstrated the potential of quantum computing. Shor's algorithm , demonstrated the capability of quantum computing to factorize integers in polynomial time, thereby showing that quantum algorithms could substantially outperform classical algorithms in solving certain computational problems [2]. Concurrently, advancements in quantum error correction techniques have alleviated some of the initial obstacles associated with qubit stability [3]. These foundational works in quantum computing create a compelling backdrop for the convergence of quantum algorithms with other fields like Reinforcement Learning (RL), which itself has garnered considerable attention for its applications across diverse areas from game playing to industrial automation. Within this context, this dissertation focuses on the investigation of Quantum

Reinforcement Learning (QRL) agents applied to the Cart-Pole task, seeking to explore the synergies and constraints that arise when RL paradigms are transplanted into a quantum computational framework.

1.3.1 Classical Reinforcement Learning

1.3.1.1 Historical Background Classical Reinforcement Learning (RL) originated from the psychological theories propounded by scholars like Thorndike [4] and Skinner [5], who laid the foundation for understanding learning through interaction with an environment. The discipline took a significant turn towards computational perspectives with the seminal text by Sutton and Barto, which translated these early psychological theories into a framework suitable for machine learning [6].

1.3.1.2 Core Principles At its heart, classical RL revolves around a set of key principles and mathematical structures. The framework is often modeled as a Markov Decision Process (MDP), characterized by states, actions, rewards, and transitions. The state space comprises all potential scenarios the agent might encounter, while the action space encompasses all available actions an agent can take. The immediate rewards for these actions are quantified through a reward function. The agent's behavior is defined by a policy that maps current states to actions. Value functions, such as state-value and action-value functions, guide the agent by estimating the expected cumulative future rewards. Furthermore, the discount factor is introduced to calculate the present value of future rewards.

1.3.1.3 Pioneering Algorithms and Techniques Several algorithms have significantly influenced the trajectory of RL research. Q-Learning, introduced by Watkins, is an off-policy algorithm that has become a cornerstone in RL [7]. On-policy learning found its manifestation in SARSA, while Temporal Difference (TD) Learning, another pivotal contribution by Sutton, combined Monte Carlo and Dynamic Programming to birth a new class of algorithms [6]. Policy Gradients were first explored through Williams's REINFORCE algorithm in 1992 [8]. The introduction of Deep Q-Networks (DQN) by Mnih et al. in 2015 incorporated deep learning to approximate Q-functions, catalyzing subsequent advancements [9]. Recent innovations include Proximal Policy Optimization (PPO) by Schulman et al., which aims for a more balanced approach to policy optimization [10].

1.3.1.4 Applications The application landscape for classical RL is broad and constantly expanding. It spans from robotics, where algorithms train robots for complex tasks such as walking and manipulation [11], to game playing, notably in mastering games like Go [12]. The algorithms have even been adapted to financial markets for tasks like portfolio optimization.

1.3.1.5 Challenges and Open Areas of Research Despite the advancements, classical RL faces several challenges. These include the age-old dilemma of exploration vs. exploitation, the need for sample-efficient learning algorithms, ensuring stability and convergence to optimal policies, and the adaptation of learned policies to similar but new environments, commonly known as transfer learning.

1.3.1.6 Classical RL, with its rich history, diverse methodologies, wide applications, and persistent challenges, forms the basis of the computational exploration of learning from interaction. The evolution of RL techniques, from basic algorithms to complex deep learning integrations, demonstrates the field's dynamism. This background in classical RL provides the essential context for the comparative analysis with quantum approaches in this study.

1.3.2 Quantum Reinforcement Learning - The Nexus of Quantum Computing and Reinforcement Learning

Quantum Reinforcement Learning (QRL) exists at the intersection of quantum computing and Reinforcement Learning. Foundational works in both quantum computing by Nielsen and Chuang [13] and Reinforcement Learning by Sutton and Barto [6] set the stage for this amalgamation. The inception of QRL was marked by the work of Dong et al. [14], who showcased the efficiency gains possible using quantum agents. Subsequent contributions have included Quantum Q-Learning by Dunjko et al. [15] and studies on the challenges and limitations of quantum algorithms in RL by Sentís et al. [16].

1.3.2.1 Quantum Parametrized Circuits (QPC) The integration of Quantum Parametrized Circuits (QPC) into machine learning, particularly in the context of Quantum Reinforcement Learning (QRL), marks a noteworthy shift in computational paradigms. Pioneered by works like the Variational Quantum Eigensolver [17], these parametrized quantum circuits have proven their efficacy in solving complex optimization problems. This has resulted in a new class of hybrid quantum-classical algorithms that are remarkably adaptable and can be tailored for specific machine learning tasks. Additionally, the notion of reuploading classical data into quantum circuits, a technique proposed by Pérez-Salinas et al. [18], has provided an efficient mechanism for data encoding in quantum algorithms. This reuploading technique has been instrumental in enhancing the capacity of QRL algorithms to process and learn from classical data. Together, these advancements in QPC are laying the groundwork for increasingly sophisticated and efficient Quantum Reinforcement Learning models.

1.3.2.2 Nuances and Challenges in Quantum Reinforcement Learning Building upon the advancements in Quantum Parametrized Circuits and their applications in Quantum Reinforcement Learning (QRL) [19], there are emerging areas of focus that warrant deeper exploration for the effective deployment and interpretation of quantum algorithms in reinforcement learning settings. Understanding the nuances in the behavior of quantum algorithms in reinforcement learning scenarios is a research area gaining substantial attention. Among these nuances, two aspects stand out as especially critical:

- **Hyperparameter Sensitivity:** The adaptability of quantum algorithms often relies on the selection of appropriate hyperparameters. Ciliberto et al. [20] conducted a comprehensive study on the impact of hyperparameters on the performance of quantum algorithms. They explored how alterations in hyperparameters, such as circuit layer counts, can significantly affect metrics like accuracy and computational speed. Their research sheds light on optimization strategies for effective quantum learning algorithms.
- **Benchmarking Quantum versus Classical Algorithms:** Benchmarking quantum algorithms against their classical counterparts is crucial for understanding their true utility [21]. Such benchmarking exercises, as articulated by Thomas et al. [21], serve two

purposes. Firstly, they provide a metric for evaluating the performance of quantum algorithms. Secondly, they identify domains where quantum algorithms excel or face challenges. These comparative studies are essential for assessing the practical advantages and limitations of quantum algorithms.

The literature on QRL reveals a nascent field in rapid evolution, with exciting developments and numerous challenges. This study situates itself within this landscape, building on existing work to conduct a comparative analysis of classical and quantum RL algorithms and exploring hyperparameter sensitivity on the selected Quantum RL agents. By implementing and analyzing these algorithms within the cart-pole environment, the research seeks to contribute meaningful insights into the burgeoning field of QRL. This concludes the introduction section, containing the relevant background/motivation, objectives and literature review which encapsulates the genesis, recent advances, and nuanced complexities of Quantum Reinforcement Learning.

2 Description of how the project was conducted

This section provides a comprehensive description of the experimental design, beginning with the choice of environment, followed by the reinforcement learning algorithms selected for the study with implementation details of both classical agents and their quantum counterparts. The section also delineates the architecture of the quantum circuits, the training paradigms, the hyperparameters used, and the metrics tracked for performance evaluation. To ensure the replicability and reliability of the study, great emphasis has been placed on detailing the scientific, mathematical, and computational techniques employed. The subsequent subsections will delve deeper into each aspect, starting with the Gym CartPole-v1 environment which is used as a testbed for evaluating the reinforcement learning algorithms in this study.

2.1 Environment: Gym Cartpole-v1

The CartPole-v1 environment is a popular benchmark problem in the domain of Reinforcement Learning, and is provided as part of the OpenAI Gym suite [22]. It has been widely adopted by the research community due to its simplicity, yet it captures the essence of various control tasks, providing a rich testing ground for different algorithms.

2.1.1 Description of the Task

In the CartPole-v1 task, a pole is attached by an unactuated joint to a cart, which moves along a frictionless track. The system is controlled by applying a force of $+1$ or -1 to the cart. The pole starts upright, and the goal is to prevent it from falling over. The agent is supposed to balance the pole by moving the cart to the left or right.

2.1.2 Observation Space

The state of the environment at any given point is defined by a 4-dimensional vector: [23]

- **Cart Position:** Ranges from -2.4 to 2.4. Represents the horizontal position of the cart on the track.

- **Cart Velocity:** Can vary from negative to positive infinity. Corresponds to the horizontal speed of the cart.
- **Pole Angle:** Ranges from about -41.8° to 41.8°. Represents the angle between the pole and the vertical axis.
- **Pole Angular Velocity:** Can vary from negative to positive infinity. Denotes the rate at which the pole angle is changing.

2.1.3 Actions

The agent has two potential actions at any given state: [23]

- **Action 0:** Push cart to the left.
- **Action 1:** Push cart to the right.

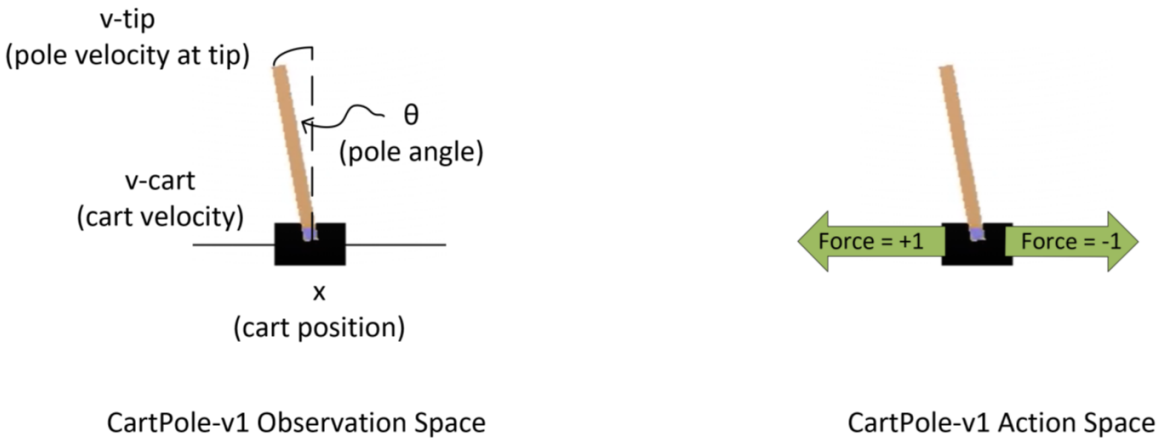

Figure 1: Observation space(left) and action space(right) for the CartPole-V1 environment [1]

2.1.4 Reward Structure

For every time step that the pole remains upright (pole angle is within ± 12 degrees from vertical), the agent receives a reward of $+1$. The episode concludes if: [23]

- The pole angle exceeds ± 12 degrees from vertical.
- The cart moves more than 2.4 units from the center.
- 500 time steps have been taken (the problem is considered "solved" if the agent can last 500 time steps consistently - averaged over last 10 episodes).

2.1.5 Episode Termination

The environment episode terminates if: [23]

- Pole Angle is more than $\pm 12°$.
- Cart Position is more than ± 2.4 (i.e., cart goes out of the screen boundary).
- Episode length exceeds 500 time steps.
- CartPole-v1 is considered "solved" when the average cumulative reward is greater than or equal to 195 over 100 consecutive trials.

The CartPole-v1 environment encapsulates a fundamental challenge in control theory and dynamics. Its seemingly straightforward task provides a reliable, yet challenging sandbox for the application and analysis of reinforcement learning algorithms. It's a valuable preliminary testbed before moving on to more complex environments and tasks.

2.2 Classical Reinforcement Learning Algorithms

The focus of the study is to ascertain the impact of quantum parameterization on conventional RL algorithms and compare their effectiveness with their classical counterparts. [24] Given the scope of this study, three RL algorithms have been selected—Deep Q-Learning, Policy Gradients (specifically, the REINFORCE algorithm), and Actor-Critic agents. These algorithms are employed in both classical and quantum-architected versions to facilitate a comprehensive comparative analysis.

The choice of these algorithms is motivated by several factors:

- **Versatility:** These algorithms are archetypal representatives of different RL approaches—value-based, policy-based, and a hybrid of the two. Thus, they offer a well-rounded perspective on the RL paradigm. [25] [26]
- **Computational Efficacy:** Deep Q-Learning and Actor-Critic agents have shown to be efficient in various complex environments, making them ideal choices for quantum adaptations. [25] [26]
- **Exploratory Capacity:** The REINFORCE algorithm under Policy Gradients is particularly useful for probing into the exploration-exploitation trade-off, a core aspect of RL. [24]

2.2.1 Policy Gradient (REINFORCE Agent)

In the realm of classical Reinforcement Learning (RL), the REINFORCE algorithm, also known as Monte Carlo Policy Gradient, offers an elegant approach to solving optimization problems by directly adjusting the policy. Formally known as the *REward Increment = Non-negative Factor × Offset Reinforcement × Characteristic Eligibility*, REINFORCE aims to maximize the expected return $J(\theta)$ by tweaking the policy $\pi(a|s;\theta)$ parameterized by θ.

The objective function can be mathematically expressed as:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi(\tau;\theta)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]$$

Here τ represents a trajectory, s_t and a_t are the state and action at time t, γ is the discount factor, and $r(s_t, a_t)$ is the reward.

The core idea behind REINFORCE is to use Monte Carlo methods to estimate the expected return of each state-action pair and then update the policy in the direction that increases the likelihood of good actions. The policy parameter update rule in REINFORCE is given by:

$$\theta_{t+1} = \theta_t + \alpha \nabla J(\theta)$$

The gradient $\nabla J(\theta)$ can be estimated as:

$$\nabla J(\theta) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=0}^{T} \nabla_\theta \log \pi(a_t^n | s_t^n; \theta) G_t^n$$

where N is the number of episodes, T is the time horizon, G_t^n is the future discounted reward from time t, and a_t^n and s_t^n are the action and state at time t in the n-th episode.

Advantages: REINFORCE has the benefit of being conceptually simple and relatively easy to implement. Because it operates directly on the policy, it can effectively handle high-dimensional state and action spaces, as well as stochastic environments. Additionally, REINFORCE has the flexibility to work well with both discrete and continuous action spaces. Its ability to operate in a model-free setting makes it suitable for problems where the environment dynamics are not fully known or are difficult to model.

Limitations: However, REINFORCE also has its drawbacks. One of the major limitations is its high variance, which can lead to slow convergence. Each policy update depends on the sampled trajectories, which can be highly variable. This issue is particularly pronounced in environments with large state-action spaces or long episode lengths. Furthermore, REINFORCE does not leverage value function approximations, missing out on potentially valuable bootstrapping methods that could accelerate learning.

In summary, REINFORCE serves as a foundational algorithm in policy gradient methods. While its conceptual simplicity and applicability to a wide range of problems make it attractive, its limitations in terms of high variance and potentially slow convergence make it important to consider more advanced variants or techniques for specific applications.

2.2.2 Deep Q-Learning Agent

Deep Q-Learning (DQN) is another pivotal algorithm in the classical RL setting, offering a compromise between value iteration and function approximation. Originating from Q-Learning, DQN employs deep neural networks as function approximators to handle environments with large state-action spaces. The central objective is to find a policy π that maximizes the expected sum of discounted rewards, expressed formally as the Q-function $Q(s, a)$. [25]

The objective can be represented as a Bellman equation, which for Q-Learning is as follows:

$$Q(s, a) = r + \gamma \max_{a'} Q(s', a')$$

DQN modifies the traditional Q-Learning by introducing two key innovations: experience replay and target networks. Experience replay collects past experiences (s, a, r, s') in a replay buffer and randomly samples from this buffer to train the neural network. This technique provides data efficiency and de-correlates the experiences.

The loss function for training the DQN neural network is defined as:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,r,s')\sim\text{ReplayBuffer}}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta)\right)^2\right]$$

The symbol θ^- refers to the parameters of the target network, a separate network whose weights are periodically updated to match the primary network, stabilizing learning.

Advantages: DQN is particularly useful for problems with high-dimensional state spaces and discrete action spaces. Its use of experience replay allows for greater data efficiency and more stable training. The target network adds another layer of stability, preventing the oscillations that can occur in naive Q-Learning implementations. These innovations enable DQN to solve complex problems that are intractable for traditional Q-Learning algorithms.

Limitations: Despite its advantages, DQN is not without its limitations. The algorithm is generally suited for environments with discrete action spaces, and extending it to continuous action spaces requires additional techniques. The use of deep neural networks introduces challenges related to hyperparameter tuning, computational demands, and potential for overfitting. Moreover, while experience replay improves data efficiency, the storage of large replay buffers can become an issue for memory-constrained systems.

In summary, Deep Q-Learning stands as a milestone in reinforcement learning, successfully marrying traditional Q-Learning with modern deep learning techniques. However, its application scope and computational demands must be carefully considered for any specific task at hand.

2.2.3 Actor-Critic Agent

Actor-Critic methods form another cornerstone in classical Reinforcement Learning, effectively marrying the advantages of value-based and policy-based methods. In an Actor-Critic framework, there are two neural networks: the Actor, which decides the action to take, and the Critic, which evaluates the action taken. The Actor updates its policy in a direction that increases expected rewards, guided by the Critic, which estimates the value function. This dual framework aims to compensate for the shortcomings of standalone value-based or policy-based approaches, providing a more balanced and effective algorithm [27]

The objective of the Actor-Critic algorithm remains consistent with traditional RL algorithms that is to optimize the policy $\pi_\theta(a|s)$ such that it maximizes the expected return $J(\theta)$, defined as the accumulated rewards r over trajectories τ:

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} r(s_t, a_t)\right]$$

The Actor's policy gradient update is expressed as:

$$\nabla J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)(r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t))\right]$$

Here, $V_\phi(s_t)$ is the Critic's estimated value function. The Critic is updated by minimizing its loss $L(\phi)$ computed using the Huber loss on the temporal-difference (TD) error.

$$L(\phi) = \mathbb{E}\left[\text{Huber}(r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t))\right]$$

In summary, the Actor and the Critic work in tandem to optimize the policy. The Actor selects actions based on its policy, and the Critic refines these choices by evaluating the expected future rewards. Their co-adaptation results in a robust and effective RL algorithm. The implementation is based on the source [28].

Advantages: The Actor-Critic method combines the strengths of policy and value iteration methods. It inherits the stability and lower variance from value-based methods while maintaining the higher expressiveness and capability to handle continuous action spaces from policy-based methods. This dual architecture allows Actor-Critic algorithms to be more sample-efficient compared to methods like REINFORCE and extends their applicability to a broader range of problems.

Limitations: Despite its flexibility and efficiency, the Actor-Critic method is not without drawbacks. It can be more computationally intensive due to the need to train two neural networks. Also, while the Critic helps to reduce the variance of the gradient estimate, it introduces bias into the process. If the Critic is poorly trained, this bias can significantly affect the performance of the Actor.

In summary, Actor-Critic methods provide a balanced approach to reinforcement learning, capable of addressing many of the limitations found in either policy-based or value-based methods. However, the efficacy of the algorithm is highly contingent on the effective training and coordination of both the Actor and the Critic networks.

2.3 Quantum versions of RL agents (using Reuploading QPC)

The quantum algorithms leverage the quantum data reuploading technique, a method to reapply the data multiple times to quantum circuits [18]. This approach ensures a richer entanglement of quantum states, crucial for RL tasks [29]. The implementation of the quantum RL models took inspiration from the implementation provided in the tensorflow documentation [30].

2.3.1 Quantum Policy Gradient (Q-Reinforce Agent)

Classical Policy Gradient methods like REINFORCE utilize neural networks to parameterize the policy, mapping states to actions. The Q-Reinforce Agent substitutes the classical neural network with a Parameterized Quantum Circuit (PQC), specifically a Re-Uploading PQC.

Re-Uploading PQC: Re-Uploading PQC is a hybrid quantum-classical model that takes classical data as input and processes it through alternating layers of quantum and classical operations. The architecture is designed to fully utilize the expressibility of quantum circuits. The model consists of n layers each composed of one quantum layer followed by one classical layer. [30]

$$\text{PQC} = \bigoplus_{i=1}^{n} [\text{U}_{\text{Q},\theta} \circ \text{U}_{\text{C},x}] \tag{1}$$

where $\text{U}_{\text{Q},\theta}$ and $\text{U}_{\text{C},x}$ are the quantum and classical unitary operations parameterized by θ and x respectively.

State Encoding: The states from the environment are normalized and directly fed into the quantum circuit. The normalization is given by:

$$s' = \frac{s}{\text{state\_bounds}} \tag{2}$$

where s' is the normalized state and state_bounds is the range of the state space.

Action Selection: Action selection is probabilistic, using a softmax over the expectations of quantum observables. Given an observable O, the probability $p(a)$ of taking action a is given by:

$$p(a) = \frac{e^{\beta\langle O\rangle}}{\sum_{a'} e^{\beta\langle O_{a'}\rangle}} \tag{3}$$

where β is a temperature parameter controlling the stochasticity of the policy.

Training and Optimization: Q-Reinforce utilizes a batched approach, gathering multiple episodes before updating the quantum circuit parameters. It calculates the returns and uses them as weights in a loss function, which is then optimized by using gradient descent methods.

$$L(\theta) = -\sum_{t} G_t \log \pi_\theta(a_t|s_t) \tag{4}$$

where G_t is the future discounted reward from time t and π_θ is the policy parameterized by θ.

The model parameters are updated by three separate Adam optimizers for input, variational, and output layers, facilitating a more nuanced optimization process.

Differences and Similarities - Quantum vs Classical: The core difference lies in the policy parameterization. While classical methods use neural networks, Q-Reinforce employs a Re-Uploading PQC. The mechanism of gathering episodes, calculating returns, and performing updates remains consistent between classical and quantum versions.

2.3.2 Quantum Deep Q-Learning Agent

Architecture: The architecture of the Quantum Deep Q-Learning Agent makes use of a parameterized quantum circuit (PQC) known as Re-Uploading PQC for function approximation. Each Re-Uploading PQC receives an input state $\mathbf{s}$, applies quantum gates parameterized by this state, and subsequently measures observables $\mathbf{O}$ corresponding to the actions. The quantum model is trained to minimize the Huber loss between the predicted Q-values and the target Q-values. [30] [31]

The Re-Uploading PQC is defined as:

$$U(\mathbf{s}, \boldsymbol{\theta}) = \bigotimes_{i=1}^{N} U_{\mathsf{layer}_i}(\mathbf{s}, \boldsymbol{\theta}_i),$$

where $\boldsymbol{\theta}_i$ are the trainable parameters for the i^{th} layer, and N is the number of layers.

Quantum Deep Q-Learning Agent employs a Parameterized Quantum Circuit (PQC) with re-uploading, which allows for classical data to be fed into the quantum circuit multiple times to perform complex transformations.

$$Q(s, a; \theta) = \langle \Psi(s; \theta)|O(a)|\Psi(s; \theta)\rangle \tag{5}$$

where θ represents trainable parameters, s is the environment state, a is the action, Ψ is the quantum state, and $O(a)$ is the action-specific observable.

Interaction with the Environment: During the interaction, the agent applies an ϵ-greedy strategy to balance exploration and exploitation. A quantum measurement is made on the Re-Uploading PQC's output to determine the agent's action. The environment then returns a reward and the next state, stored in the agent's replay memory as tuples $(\mathbf{s}, a, r, \mathbf{s}')$.

Training Procedure: :The learning algorithm iteratively updates the Q-function model by minimizing the Huber loss function between predicted and target Q-values, calculated as follows:

$$L(\theta) = \sum [\delta(\theta)]^2 \quad \text{if } |\delta(\theta)| < 1 \tag{6}$$

where $\delta(\theta) = Q(s, a; \theta) - (r + \gamma \max_{a'} Q(s', a'; \theta^-))$, r is the reward, s' is the new state, and θ^- are the parameters of the target model.

Learning Mechanism: The agent performs batch updates on its quantum model using the collected interactions. Specifically, it employs the Q-learning update rule, given by:

$$Q(\mathbf{s}, a) \leftarrow Q(\mathbf{s}, a) + \alpha\big(r + \gamma \max_{a'} Q(\mathbf{s}', a') - Q(\mathbf{s}, a)\big).$$

Here, α is the learning rate, γ is the discount factor, and $\max_{a'} Q(\mathbf{s}', a')$ is estimated using a target model to stabilize training.

Exploration Strategy: The agent utilizes an epsilon-greedy policy for exploration, where ϵ decays exponentially over time. A random action is chosen with probability ϵ, and the action with the highest Q-value is chosen otherwise.

$$\epsilon_t = \max(\epsilon_{\text{min}}, \epsilon_0 \times \lambda^t) \tag{7}$$

Optimization: Three separate Adam optimizers are employed for the input re-uploading layer, variable layer, and the output layer, respectively. This segmented optimization strategy enhances the learning efficiency.

Differences and Similarities - Quantum vs Classical: The quantum version leverages quantum parallelism through Re-Uploading PQC to achieve more complex function approximations with fewer parameters. In classical DQNs, the approximation is performed by neural networks which may require more layers and neurons for similar complexity.

Similarities include the use of ϵ-greedy strategy for action selection and the Q-learning update rule for learning. Both versions also employ experience replay for more efficient and stable training. However, in the quantum case, training and inference involve quantum computations that are computationally more intensive but promise benefits in terms of representational capacity.

2.3.3 Quantum Actor-Critic Agent

The Quantum Actor-Critic model is one of the approaches used in deep reinforcement learning, adapted for quantum computing. This model leverages the benefits of quantum circuits to construct both the actor and critic functions. The overall algorithm aligns closely with its classical counterpart, but the crucial difference lies in the implementation of the actor and critic models as Parametrized Quantum Circuits (PQC). [32]

Framework and Environment: This quantum-based Actor-Critic model is implemented using TensorFlow Quantum (TFQ) along with Cirq for quantum circuit construction. Gym is used for setting up the reinforcement learning environment. The code is written in Python, with a modular approach that aids in readability and future scalability. For instance, a separate 'helper.py' contains auxiliary reusable functions to improve maintainability.

$$\text{Actor Model} = \text{PQC} \circ \text{Softmax} \tag{8}$$

$$\text{Critic Model} = \text{PQC} \tag{9}$$

Implementation: The QAC agent is implemented using a quantum circuit comprising both quantum and classical components. The quantum component is responsible for estimating the state-action values (Q-values), while the classical part acts as a policy guide.

- **Quantum Component:** The quantum circuit uses parameterized quantum gates for function approximation. The quantum state encodes both the environment state s and possible actions a, and the circuit outputs the Q-value of taking action a in state s. Mathematically, this can be represented as:

  $$Q(s, a) = \langle \psi(s, a) | H | \psi(s, a) \rangle$$

  Here, H is the Hamiltonian operator, and $\psi(s, a)$ is the quantum state representing the state-action pair.

- **Classical Component:** The policy network guides the agent's actions based on the Q-values. Similar to the classical counterpart, it could use a softmax function to select an action probabilistically.

  $$P(a|s) = \frac{e^{Q(s,a)}}{\sum_{a'} e^{Q(s,a')}}$$

Quantum Circuits: The architecture makes use of a reuploading strategy, wherein the quantum data is repeatedly fed into the circuit, and the circuit parameters are updated at each layer. The circuits for both the actor and critic are designed using Cirq, each comprising of variational layers and entangling layers.

$$\text{Variational Layer} = R_x(\theta_x) \otimes R_y(\theta_y) \otimes R_z(\theta_z) \tag{10}$$

$$\text{Entangling Layer} = CZ(\text{qubit}_i, \text{qubit}_{i+1}) \tag{11}$$

Training: The training process involves two main components—the actor and the critic—optimized using the Huber loss and the Adam optimizer. As in classical Actor-Critic models, both the actor and the critic participate in the learning process, but here they are quantum circuits parametrized by angles that get adjusted during the learning process.

$$\text{Loss} = \text{Actor Loss} + \text{Critic Loss} \tag{12}$$

Differences and Similarities - Quantum vs Classical: While the overall architecture remains the same, the key difference is the utilization of quantum circuits to replace the neural networks in the classical version. This allows for exploring complex state-action spaces more efficiently, albeit with the trade-off of requiring specialized quantum hardware for full-scale implementation. On the similarities front, both approaches use a gradient-based optimization algorithm, and both aim to minimize a loss function that includes components from the actor and the critic. The quantum version, however, offers an avenue to leverage quantum parallelism and superposition to potentially achieve better exploration of the state-space.

2.4 Experimental Setup:

The overarching aim of this dissertation is to conduct a rigorous investigation into the capabilities and efficiencies of Quantum Reinforcement Learning (QRL) algorithms in comparison to their classical counterparts. The study is designed to answer key questions related to performance, computational trade-offs, and the impact of quantum circuit layers on QRL algorithms. All experiments were executed in a controlled, simulated environment using TensorFlow Quantum, thereby eliminating the need for a real quantum computer.

The CartPole-v1 environment from OpenAI's Gym was chosen as the testbed for all experiments. This environment is a standard benchmark in the field of Reinforcement Learning (RL) and offers a controlled setting for evaluating the performance of RL algorithms. The environment simulates a cart that moves along a frictionless track with a pole attached to it. The goal is to balance the pole by moving the cart left or right.(Refer section 2.1)

Three types of Reinforcement Learning agents were evaluated:

1. Policy Gradient REINFORCE: A model-free algorithm that optimizes the policy directly.
2. Deep Q-Learning: A value-based algorithm that uses a neural network to approximate the Q-function.
3. Actor-Critic: A hybrid approach that uses both policy and value functions.

Each of these algorithms was implemented in both classical and quantum versions. The quantum versions utilized ReUploading Quantum Parametric Circuits (ReUploading QPC) for the policy and value networks. A total of 12 experiments were designed, each with 10 runs to ensure statistical significance. The experiments were systematically categorized based on three factors:

1. Type of RL Agent: Policy Gradient REINFORCE, Deep Q-Learning, or Actor-Critic.
2. Type of RL variant: Quantum, Classical
3. Number of Layers in Quantum Circuits: 3, 4, or 5 layers.

The experiments were further grouped into two overarching types for comparative analysis:

1. Quantum vs Classical: Compares the performance of quantum and classical versions of each RL agent.
2. Quantum N Layer: Investigates the impact of varying the number of layers in the quantum circuits.

As for the hyperparameters, there are some common hyperparmaeters across all 12 experiments. (Refer Table 1) Each RL agent had its own set of specific hyperparameters, such as learning rate, gamma, and architecture, which were meticulously chosen based on preliminary tests and hyperparamter tuning and literature review. These specific hyperparmaters for different experiment types are presented in the below Tables (Refer Tables 2, 3, 4, 5, 6, 7).

All of the information needed to reliably replicate the experiments is given in these tables along with seed values used for each run. The same seed is applied for RL agent, environment and any other random sampling involved in the experiments thereby making it fully reproducible. One caveat is that RL agents training has some innate randomness which might not

be exactly replicated but with all other hyperparameters properly set, the average results across multiple runs should be identical.

Parameter	**Value**
Env Name	CartPole-v1
Number of Episodes	5000
Batch Size	16

Table 1: Common Hyperparameters for all RL Experiments

Parameter	**Value**
Number of Runs	10
Seed	[139, 239, 339, 439, 539, 639, 739, 839, 939, 1039]
Gamma	0.99
Learning Rate	0.01
Model Architecture	[4, [32], 2]

Table 2: Hyperparameters for the Classical Policy Gradient Model

Parameter	**Value**
Number of Runs	10
Seed	[139, 239, 339, 439, 539, 639, 739, 839, 939, 1039]
Gamma	0.99
Learning Rate	0.001
Model Architecture	[4, [32, 16], 2]
max_memory_length	30000
epsilon	1.0
epsilon_min	0.05
decay_epsilon	0.999
steps_per_update	10
steps_per_target_update	30

Table 3: Hyperparameters for the Classical DQN Model

Parameter	Value
Number of Runs	10
Seed	[139, 239, 339, 439, 539, 639, 739, 839, 939, 1039]
Gamma	1
Learning Rate	0.01
Model Architecture	[4, [32], [2, 1]]

Table 4: Hyperparameters for the Classical Actor-Critic Model

Parameter	Value
Number of Runs	10
Seed	[139, 239, 339, 439, 539, 639, 739, 839, 939, 1039]
Gamma	1
Learning Rate	{'in': 0.1, 'var': 0.01, 'out': 0.1}
Model Architecture	[4, 'ReUploadingPQC', 2]
N Layers	[5, 4, 3]
Trainable Params	[94, 78, 62]

Table 5: Hyperparameters for Quantum Policy Gradient Experiment

Parameter	Value
Number of Runs	10
Seed	[139, 239, 339, 439, 539, 639, 739, 839, 939, 1039]
Gamma	0.99
Learning Rate	{'in': 0.001, 'var': 0.001, 'out': 0.1}
Model Architecture	[4, 'ReUploadingPQC', 2]
max_memory_length	10000
epsilon	1.0
epsilon_min	0.01
decay_epsilon	0.99
steps_per_update	10
steps_per_target_update	30
N Layers	[5, 4, 3]
Trainable Params	[94, 78, 62]

Table 6: Hyperparameters for Quantum DQN Experiment

Parameter	Value
Number of Runs	5
Seed	[139, 239, 339, 439, 539]
Gamma	1
Learning Rate	0.01
Model Architecture	[[4, 4], 'ReUploadingPQC', [2, 2]]
N Layers	[5, 4, 3]
Trainable Params	[188, 156, 124]

Table 7: Hyperparameters for Quantum AC Experiment

The codebase is modular and well-organized into separate directories and files for ease of use and reproducibility.

- main.py: Orchestrates the experiments based on a JSON configuration file.
- helper.py: Contains utility functions.
- data_scripts.py: Handles data manipulation, metrics calculation, and plotting.

The RL agents implementations are neatly stored in two directories: classical_RL_agents and quantum_RL_agents, each containing Python files for the respective agents.

The outcomes of each run were meticulously stored in JSON files within their respective experiment directories. After all runs for a given experiment were completed, the data was aggregated, and a combined outcome file was generated. This structured data storage facilitates easy retrieval and analysis. All the metrics and plotting are done using the stored outcome json files which contain all exhaustive information relating to all runs of every experiment.

2.5 Metrics and Statistics

Five key metrics were calculated for each experiment to provide a comprehensive evaluation with relevant descriptive statistics for each of the experiment.

- **Average Reward Per Episode:** Measures the effectiveness of the policy.
  Descriptive statistics - Mean, Median, Standard Deviation, Min, Max, 25th and 75th Percentiles (Refer Table 8)
- **Learning Curve:** A graph charting the rewards against the number of episodes
  Descriptive Statistics - Mean Convergence Point, Std of Convergence Point, Mean Initial Convergence Point, Mean Post-Initial-Convergence Std, Convergence Ratio (Refer Table 9)
- **Convergence Time:** Number of episodes taken for the agent to consistently achieve maximum rewards(500).
  Descriptive Statistics - Mean, Median, Standard Deviation, Min, Max (Refer Table 10)
- **Computational Complexity:** Assessed by measuring algorithm execution times.
  Descriptive Statistics - Mean Time (Training Phase), Standard Deviation (Refer Table 11)
- **Efficiency:** Ratio of total rewards to training time.
  Descriptive Statistics - Mean, Median, Standard Deviation, Min, Max (Refer Table 12)

We have also run statistical tests for groups of experiments for effective and robust comparative analysis. t-tests were conducted for two-group comparisions(classical vs quantum) and Analysis of Variance (ANOVA) for three-group comparisions(quantum n layer-[3,4,5]) supplemented by effect size and power analysis. (Refer Tables 13, 14, 15, 16, 17, 18)

3 Results

The focus of this dissertation is to explore the interface between Quantum Computing and Reinforcement Learning (RL), specifically examining the efficacy and performance of quantum RL algorithms compared to their classical counterparts. This section aims to present the findings of a comprehensive series of experiments conducted in the Gym CartPole environment, leveraging both classical and quantum versions of three reinforcement learning algorithms—Policy Gradient REINFORCE, Deep Q-Learning, and Actor-Critic.

Our methodology consisted of a multi-layered approach designed to tease apart the intricate dynamics of quantum RL algorithms. In all, we executed 12 distinct experiments, each conducted over 10 runs to ensure statistical reliability. The data collected has been aggregated, and the metrics calculated encompass a wide array of performance indicators such as rewards, training time, and computational cost, alongside advanced statistical measures including t-tests, ANOVA, and effect sizes.

This section will unfold as follows: First, we present a summary of the data collection process and the descriptive statistics for each experimental configuration. We then delve into comparative analyses, which pit classical against quantum algorithms and investigate variations among different layer configurations within quantum RL algorithms. Visual representations are provided to supplement the statistical analysis, thereby providing a more holistic understanding of the data. Lastly, we discuss any uncertainties involved, presenting all numerical values alongside their corresponding standard deviation of the mean.

By rigorously adhering to these structures, we aim to provide an exhaustive yet discernible insight into the capabilities and limitations of Quantum Reinforcement Learning algorithms. The intent is not only to contribute to the academic understanding of the subject matter but also to furnish actionable insights that could inform future research and applications in the rapidly evolving field of quantum computing.

3.1 Data Collection Summary

Data collection is a fundamental component of this research, as it provides the empirical evidence needed to evaluate the efficacy of classical and quantum reinforcement learning algorithms in the Gym CartPole-v1 environment. As outlined in the methodology section, we employed a multi-layered experimental design encompassing classical RL algorithms—namely Policy Gradient REINFORCE, Deep Q-Learning, and Actor-Critic—as well as their quantum versions based on Reuploading Quantum Parameterized Circuits (Reuploading QPC) [Refer to Section 2.2, 2.3].

3.1.1 Overview

Each algorithm was executed over a series of 10 runs to ensure a robust and statistically reliable dataset. Every run encompassed a predefined number of episodes (Refer to Section 2.4 for performance metrics), allowing for both intra and inter-algorithm comparisons. The data was collected in a controlled computational environment to ensure consistency and repeatability.

Five key metrics that were considered to evaluate performance are Average Reward Per Episode, Learning Curve, Convergence Time, Computational Complexity and Efficiency.

3.1.2 Data Storage and Organization

The raw data, including the rewards and computation times, were stored in CSV files, making it easy to manipulate, analyze, and visualize. The data is organized by algorithm type, episode number, and run index to facilitate a structured analysis.

For further details on the experimental setup, task description, and algorithmic configurations, readers are encouraged to consult Section 2: Description of how the project was conducted.

By adhering to this data collection process, we aim to maintain scientific rigor while providing an objective evaluation of the targeted algorithms in both classical and quantum paradigms. Subsequent subsections will present a comprehensive analysis and interpretation of this data, aiming to contribute to our understanding of Quantum Reinforcement Learning and its practical implications.

3.2 Descriptive Statistics

Before diving into the analytical results, it's critical to understand the foundational statistics that were aggregated during the experiments. Four metrics—Average Reward Per Episode, Learning Curve, Convergence Time, and Computational Complexity—were chosen based on their widespread acceptance in reinforcement learning research as well as their suitability for capturing different aspects of performance in both classical and quantum paradigms. These metrics were calculated by aggregating the results of all the reruns (10 runs) for each experiment, to yield a more statistically reliable representation. Below are tables presenting the aggregated statistics for each metric across different algorithms.

3.2.1 Average Reward Per Episode

Table 8 encapsulates the descriptive statistics for the average reward per episode metric. It reflects the average reward gained per episode, thus providing a snapshot of the overall efficacy of each algorithm.

Agent Type	Mean	Median	Std	Min	Max	25th Percentile	75th Percentile
Classical REINFORCE	230.74	219.55	40.37	181.48	317.89	196.78	252.12
Quantum REINFORCE (5-layer)	222.27	228.07	66.33	111.17	309.74	174.88	283.61
Quantum REINFORCE (4-layer)	219.13	212.50	42.15	151.16	316.10	199.88	222.45
Quantum REINFORCE (3-layer)	190.90	197.16	44.93	117.70	266.89	150.93	220.64
Classical DQN	101.16	94.03	34.84	58.26	177.47	81.24	106.48
Quantum DQN (5-layer)	147.60	140.49	47.71	73.41	263.82	128.67	160.65
Quantum DQN (4-layer)	111.50	108.34	43.84	54.90	177.71	70.83	147.54
Quantum DQN (3-layer)	123.26	112.83	37.76	60.12	183.36	100.22	154.30
Classical Actor Critic	173.59	169.43	30.34	116.16	231.69	157.65	182.55
Quantum Actor Critic (5-layer)	162.19	156.72	26.26	122.81	198.02	150.21	183.19
Quantum Actor Critic (4-layer)	165.18	169.08	24.01	134.42	201.89	144.03	176.48
Quantum Actor Critic (3-layer)	204.74	206.71	27.85	166.56	249.58	186.88	213.99

Table 8: Average Reward Per Episode Metrics for Classical and Quantum Reinforcement Learning Agents

3.2.2 Learning Curve

Table 9 illustrates how quickly each algorithm learns to optimize its policy, capturing the trade-off between learning speed and efficacy.

Agent Type	Mean Convergence Point	Std of Convergence Point	Mean Initial Convergence Point	Mean Post Initial Convergence Std	Convergence Ratio
Classical REINFORCE	793.60	264.58	372.30	117.21	1.0
Quantum REINFORCE (5-layer)	600.00	223.34	293.60	119.61	1.0
Quantum REINFORCE (4-layer)	702.40	179.24	325.70	127.06	1.0
Quantum REINFORCE (3-layer)	880.00	654.20	370.70	124.86	1.0
Classical DQN	1862.50	405.27	1367.88	140.01	0.8
Quantum DQN (5-layer)	1140.00	649.03	594.67	125.49	0.9
Quantum DQN (4-layer)	1177.14	108.59	904.57	144.22	0.7
Quantum DQN (3-layer)	1836.67	585.79	1029.67	151.33	0.6
Classical Actor Critic	343.00	77.21	182.20	154.69	1.0
Quantum Actor Critic (5-layer)	1036.00	239.38	428.80	138.70	1.0
Quantum Actor Critic (4-layer)	1492.00	393.31	543.20	136.08	1.0
Quantum Actor Critic (3-layer)	1206.00	281.75	374.80	137.38	1.0

Table 9: Learning Curve Metrics for Classical and Quantum Reinforcement Learning Agents.

3.2.3 Convergence Time

Table 10 focuses on the number of episodes required for the algorithm to consistently achieve near-maximum rewards, providing insights into the efficiency of each approach.

Agent Type	Mean (s)	Median (s)	Std (s)	Min (s)	Max (s)
Classical REINFORCE	793.60	720.0	264.58	512	1392
Quantum REINFORCE (5-layer)	600.00	576.0	223.34	224	1008
Quantum REINFORCE (4-layer)	702.40	696.0	179.24	416	1008
Quantum REINFORCE (3-layer)	880.00	664.0	654.20	336	2736
Classical DQN	1862.50	1875.0	405.27	1230	2510
Quantum DQN (5-layer)	1140.00	990.0	649.03	510	2650
Quantum DQN (4-layer)	1177.14	1200.0	108.59	1050	1380
Quantum DQN (3-layer)	1836.67	1935.0	585.79	720	2520
Classical Actor Critic	343.00	355.0	77.21	220	480
Quantum Actor Critic (5-layer)	1036.00	980.0	239.38	740	1370
Quantum Actor Critic (4-layer)	1492.00	1440.0	393.31	1090	2230
Quantum Actor Critic (3-layer)	1206.00	1190.0	281.75	850	1710

Table 10: Convergence Time Metrics for Classical and Quantum Reinforcement Learning Agents in milliseconds.

3.2.4 Computational Complexity

Tables 11 delves into Computational Complexity, which is crucial for understanding the practicality of quantum algorithms, especially given their computational overhead.

Agent Type	Mean Time (Training Phase) (s)	Std (s)
Classical REINFORCE	21.26	6.78
Quantum REINFORCE (5-layer)	390.25	210.87
Quantum REINFORCE (4-layer)	371.96	130.20
Quantum REINFORCE (3-layer)	402.20	397.65
Classical DQN	4485.15	2745.61
Quantum DQN (5-layer)	1867.67	1667.12
Quantum DQN (4-layer)	1802.57	1308.27
Quantum DQN (3-layer)	2199.98	1109.29
Classical Actor Critic	153.62	28.09
Quantum Actor Critic (5-layer)	11631.93	2998.76
Quantum Actor Critic (4-layer)	16837.14	7471.13
Quantum Actor Critic (3-layer)	13273.49	2842.93

Table 11: Computational Complexity Metrics for Classical and Quantum Reinforcement Learning Agents in seconds.

3.2.5 Efficiency

Table 12 summarizes the descriptive statistics for Efficiency across all 12 experiments which serves as an indicator of how quickly and effectively the agent is learning. This metric encapsulates not only the agent's performance but also the computational resources consumed, providing a balanced view of the agent's practicality for real-world applications.

Agent Type	Mean	Median	Std	Min	Max
Classical REINFORCE	8578.41	8935.29	1174.67	6959.49	10473.39
Quantum REINFORCE (5-layer)	355.75	354.00	28.24	306.31	398.74
Quantum REINFORCE (4-layer)	420.04	420.56	42.71	372.22	486.40
Quantum REINFORCE (3-layer)	452.44	451.37	38.37	395.77	513.61
Classical DQN	52.13	50.38	7.10	43.97	67.79
Quantum DQN (5-layer)	114.88	122.26	14.50	93.99	137.85
Quantum DQN (4-layer)	117.23	115.32	7.96	104.62	130.32
Quantum DQN (3-layer)	135.23	134.85	6.95	122.03	145.43
Classical Actor Critic	378.21	376.47	12.02	360.21	405.02
Quantum Actor Critic (5-layer)	14.50	14.45	0.33	13.99	15.03
Quantum Actor Critic (4-layer)	15.30	15.01	0.88	14.52	16.89
Quantum Actor Critic (3-layer)	18.37	18.06	1.12	17.24	20.49

Table 12: Efficiency Metrics for Classical and Quantum Reinforcement Learning Agents.

With this, we have covered the foundational descriptive statistics that form the basis of our later analytical scrutiny. Each metric has been selected for its ability to shed light on specific aspects of the performance dynamics between classical and quantum reinforcement learning algorithms.

3.3 Comparative Analysis(For Rewards)

Having established a foundational understanding of the descriptive statistics for each key performance metric, we now turn our attention to comparative analyses between classical and quantum reinforcement learning (RL) algorithms. The goal is to discern if the quantum enhancements to classical RL algorithms yield a statistically significant improvement or deterioration in performance. The analysis is structured into two major comparisons: Classical vs. Quantum RL algorithms and inter-comparisons among quantum RL agents with varying number of architectural layers.

For each experimental setup, we aggregate reward metrics and perform statistical tests to determine whether the observed differences are statistically significant. Statistical measures, including T-statistic, P-Value, effect size (Cohen's d and η^2), and required sample size for adequate power, are presented.

T-tests were performed for two-group comparisons and One-way ANOVA tests for more than two groups. Effect sizes were calculated as Cohen's d for T-tests and as η^2 and ω^2 for ANOVA tests. It should be noted that the P-Values less than 0.05 indicate that the differences are statistically significant, and the required sample size indicates the number of samples required for the experiment to achieve a power of 0.8.

3.3.1 Classical vs Quantum(5 layers)

To compare classical algorithms with their quantum counterparts, a series of statistical tests were conducted, such as t-tests for 2-group comparisons.

Metric	**T-Statistic**	**P-Value**	**Cohen's d**	**Required Sample Size**
Reward	0.6906	0.4899	0.0286	16561

Table 13: Advanced Statistical Metrics for Comparing Quantum and Classical Policy Gradient using REINFORCE.

The T-statistic is 0.6906, and the P-Value is 0.4899, which is much higher than the significance level of 0.05. This indicates that there is no statistically significant difference between the classical and quantum versions of the REINFORCE algorithm in terms of rewards. (Refer to Table 13)

Metric	**T-Statistic**	**P-Value**	**Cohen's d**	**Required Sample Size**
Reward	-30.7578	0.0	-0.7942	26

Table 14: Advanced Statistical Metrics for Comparing Quantum and Classical Deep Q-Learning.

The T-statistic of -30.7578 and a P-Value of 0.0 are highly indicative of a statistically significant difference between the two groups. The negative T-statistic and a large effect size (-0.7942) suggest that the classical version outperforms the quantum one. (Refer to Table 14)

Metric	**T-Statistic**	**P-Value**	**Cohen's d**	**Required Sample Size**
Reward	2.3933	0.0169	0.1364	1627

Table 15: Advanced Statistical Metrics for Comparing Quantum and Classical Actor-Critic Methods.

Here, the T-statistic is 2.3933 and the P-Value is 0.0169, which is less than 0.05. Thus, there's a statistically significant difference between classical and quantum actor-critic methods. The positive T-statistic indicates that the quantum version performs better in terms of reward. (Refer to Table 15)

3.3.2 Between Quantum RL Agents (5,4,3 Layers)

For a deeper dive into the performance of quantum RL algorithms, comparisons were made among agents with different numbers of layers (5, 4, and 3) in their architectures.

Metric	**F-Statistic**	**P-Value (ANOVA)**	η^2	ω^2	**Required Sample Size**
Reward	0.0793	0.9238	0.0745	0.0741	1759

Table 16: Advanced Statistical Metrics for Quantum N Layer Policy Gradient REINFORCE.

The F-statistic is quite low (0.0793), and the P-Value (0.9238) is not significant. This suggests that there's no substantial difference in rewards between quantum REINFORCE agents with different numbers of layers. (Refer to Table 16)

Metric	**F-Statistic**	**P-Value (ANOVA)**	η^2	ω^2	**Required Sample Size**
Reward	0.1362	0.8727	0.115	0.1148	734

Table 17: Advanced Statistical Metrics for Quantum N Layer Deep Q Learning.

Again, the F-statistic is low (0.1362) and the P-Value (0.8727) is much higher than 0.05, indicating no significant differences in performance between quantum agents with 3, 4, and 5 layers. (Refer to Table 17)

Metric	**F-Statistic**	**P-Value (ANOVA)**	η^2	ω^2	**Required Sample Size**
Reward	0.0161	0.9841	0.0149	0.0145	45807

Table 18: Advanced Statistical Metrics for Quantum N Layer Actor Critic.

The F-statistic is negligible (0.0161) and the P-Value is extremely high (0.9841). These results confirm that different architectural layers do not result in significant performance variations. (Refer to Table 18)

The required sample size for all tests is quite large in some cases, indicating that the results might be more reliable with larger datasets. It is intriguing that the quantum enhancements didn't yield superior performance across the board. The nuances in these results could be highly

informative for further research. By providing these detailed comparative analyses, we aim to inform the reader about the performance characteristics and statistical differences between various RL setups in both classical and quantum computing paradigms. The discussion section will mainly focus more on the interpretation of results from the experiments done and not on the statistical significance of the results because of known deficiencies like limited sample size, limited scope of the study with time and resource constraints.

3.3.3 Visual Representations

This subsection aims to provide a graphical analysis of the rewards obtained across different experimental setups. Six reward curve plots are presented below, each representing a unique experimental comparison. The curves showcase the reward dynamics, making it easier to discern patterns, outliers, or anomalies that may not be evident from numerical statistics alone.

3.3.4 Classical vs Quantum RL

The REINFORCE plot (Refer to Figure 2) shows that the classical Policy Gradient REINFORCE algorithm consistently outperforms its quantum counterpart across all episodes in terms of average rewards. The Deep Q Learning plot (Refer to Figure 3) indicates that quantum model outperforms the classical model consistently, with the Quantum model showing a stable learning curve. The Actor Critic plot (Refer to Figure 4) shows that classical architecture outperforms the quantum model consistently and converges faster.

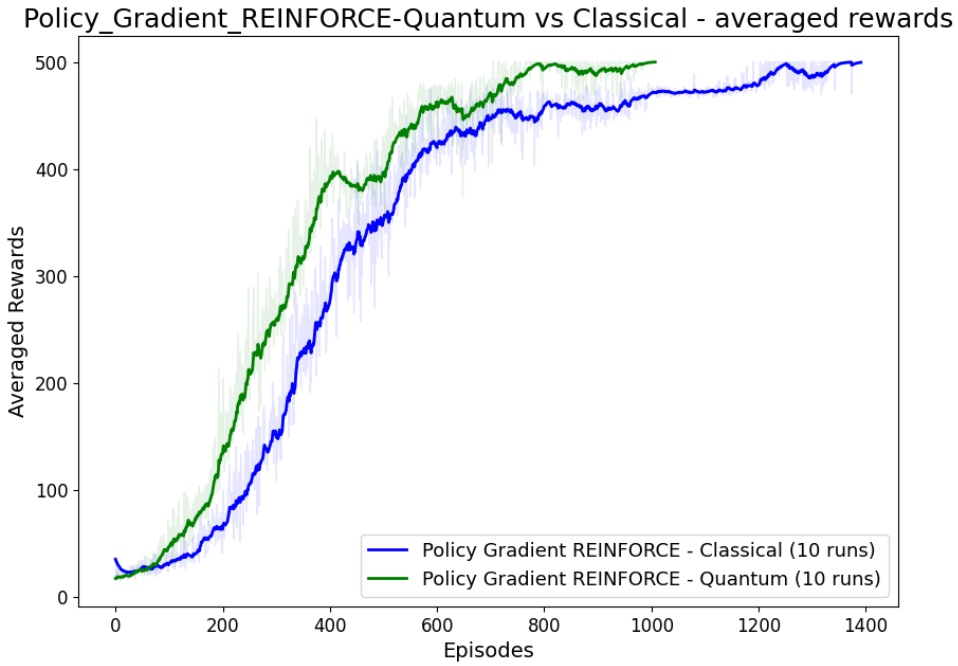

Figure 2: Comparison of averaged rewards between Quantum and Classical approaches using the REINFORCE Policy Gradient method.

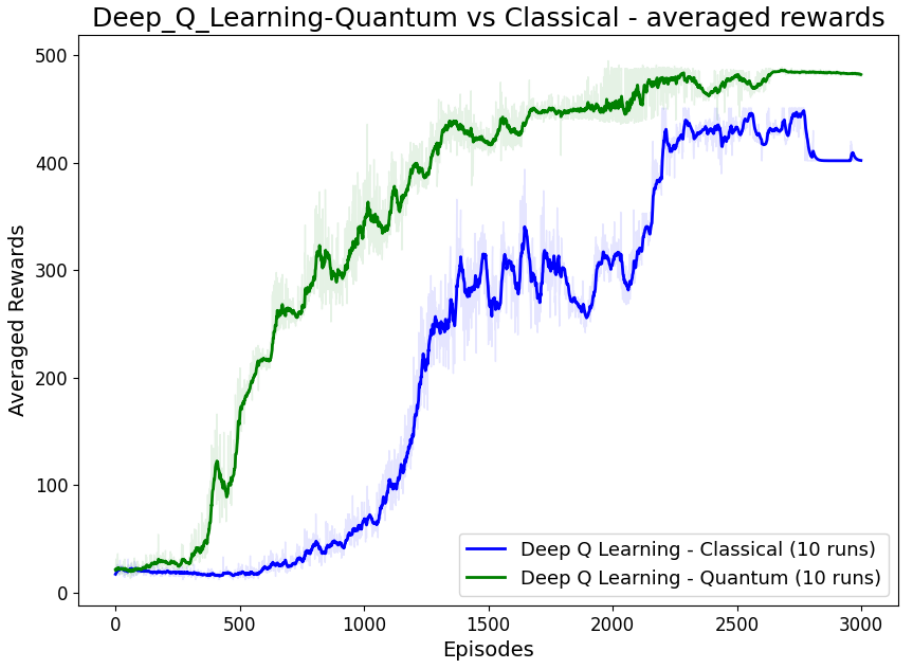

Figure 3: Comparison of averaged rewards between Quantum and Classical approaches using the Deep Q Learning method.

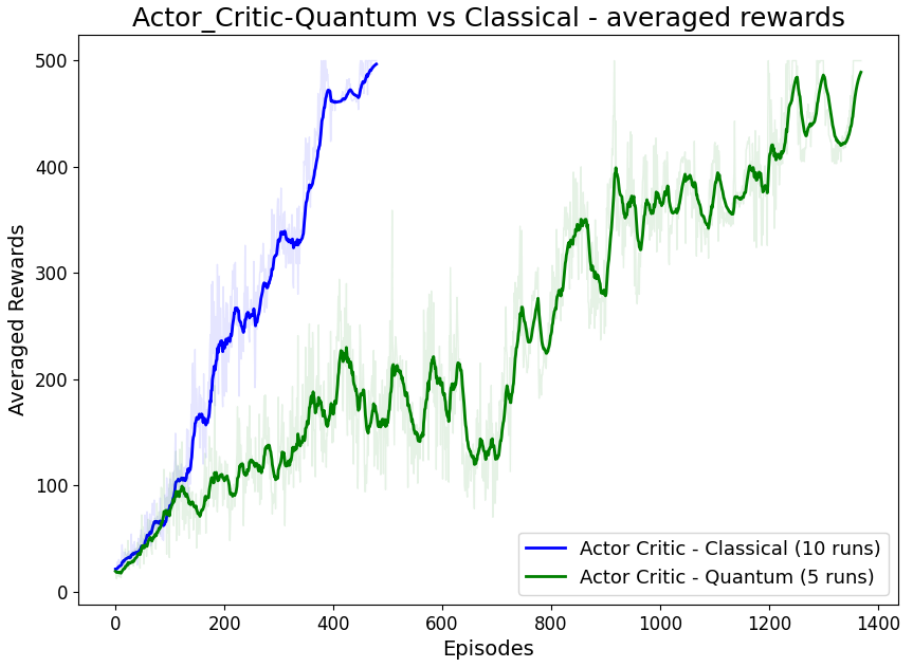

Figure 4: Comparison of averaged rewards between Quantum and Classical approaches using the Actor Critic method.

3.3.5 Between Quantum RL Agents (5,4,3 Layers)

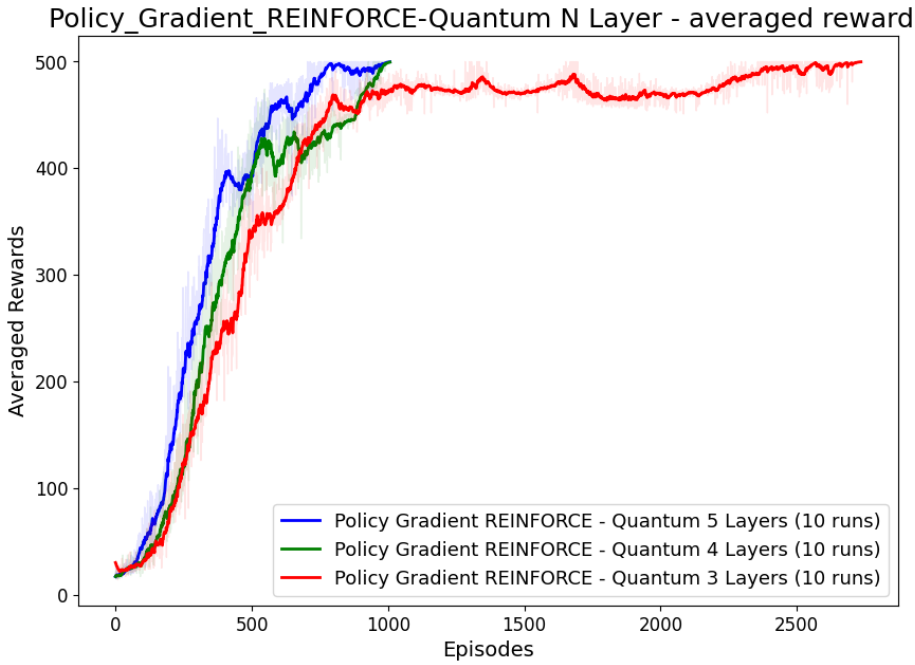

Figure 5: Comparison of averaged rewards between 3,4,5 layers of Quantum REINFORCE Policy Gradient method.

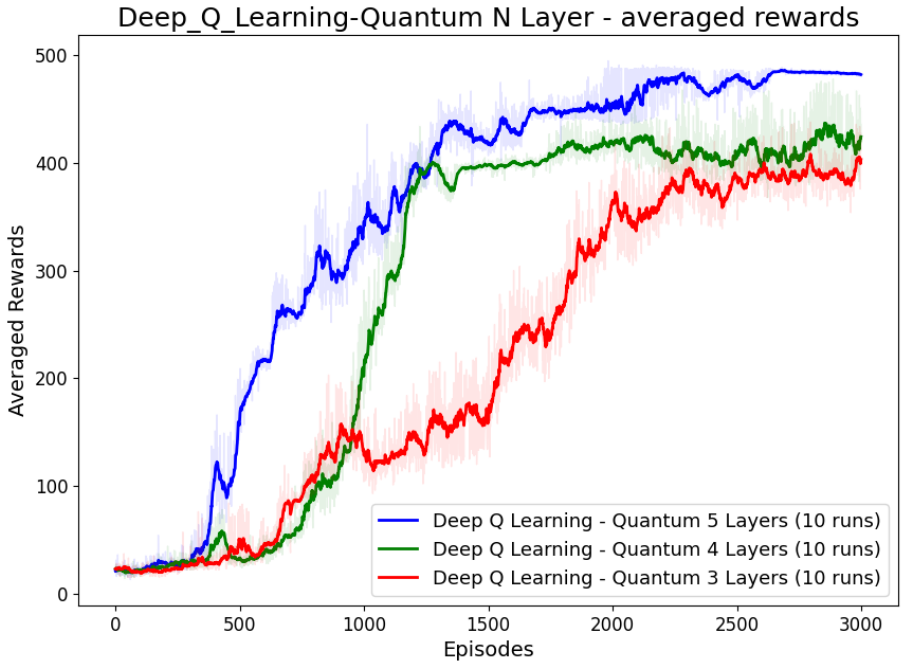

Figure 6: Comparison of averaged rewards between 3,4,5 layers of Quantum Deep Q Learning method.

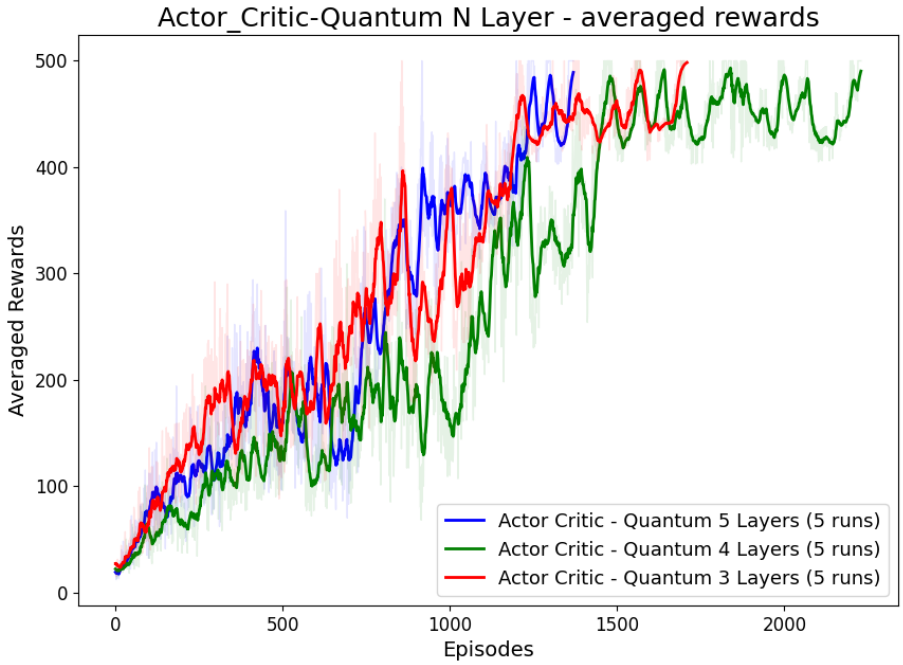

Figure 7: Comparison of averaged rewards between 3,4,5 layers of Quantum Actor Critic method.

The REINFORCE plot (Refer to Figure 5) illustrates that Policy Gradient REINFORCE agents with a 5-layer quantum architecture achieve faster convergence to higher reward values compared to those with fewer layers. The DQN plot (Refer to Figure 6) shows that Deep Q-Learning agents with 5-layer quantum architectures experience a more stable and higher reward convergence compared to those with 3 and 4 layers. The Actor Critic plot (Refer to Figure 7) reveals that the Actor-Critic algorithm with a 5-layer quantum architecture outperforms the 3 and 4-layer versions in terms of reward stability and convergence speed.

3.3.6 Observations and Anomalies

1. Across different RL agents, quantum architectures generally perform at least as well as or better than classical ones, especially noticeable in the Actor-Critic model.
2. In the Quantum N Layer experiments, there isn't a straightforward correlation between the number of layers and performance, which may imply that simply stacking more quantum layers is not always beneficial.
3. Anomalies in reward curves, such as sudden peaks or troughs, warrant further investigation into the stability and reliability of these models.

Visual representations such as these provide essential insights and serve as a prelude to the more detailed statistical analyses. The patterns observed will guide future experiments and help to refine our RL models, classical or quantum.

3.3.7 Uncertainty and Error Analysis

In any scientific inquiry, acknowledging the uncertainty and error in the collected data is crucial for interpretation and the generalisability of results. Given the complexity and inherent

variability in reinforcement learning algorithms, especially in hybrid models that incorporate quantum computations, this subsection delves deep into understanding the uncertainties and errors that could arise in our study. Understanding these aspects is crucial for interpreting the results objectively and for future replication of experiments.

3.3.7.1 Standard Deviation of the Mean The standard deviation of the mean provides an essential metric for gauging the variability within the data set. For this project, the following interpretations can be drawn from the metrics:

Learning Curve: The standard deviation around the mean convergence point and plateau reward can indicate the algorithm's sensitivity to initial conditions and random fluctuations in the environment.

Convergence Time: A high standard deviation here could signify that the algorithm is not consistently fast in achieving a near-optimal policy.

Computational Complexity: Variability in the mean time taken for training and inference phases would suggest inconsistencies in computational costs, especially critical in quantum computations due to their resource-intensive nature.

Efficiency: A high standard deviation in the ratio of total rewards to training time might indicate that the agent is not uniformly efficient across different runs or settings.

3.3.7.2 Error Sources :

Sampling Error: Even though each experiment was run 10 times, it is still a finite sample. More runs might offer a different perspective.

Modeling Error: The quantum implementations used a reuploading QPC. This method, while effective, could introduce errors that are not present in the classical algorithms.

Quantum Errors: Quantum computations are prone to errors due to qubit decoherence and gate fidelity, which could introduce additional variability in Quantum RL agents. This is not relevant to us as we have not run the quantum RL agents on an actual quantum computer.

Exploration Noise: In RL algorithms, especially ones like Policy Gradient and Deep Q Learning, the randomness introduced during the exploration phase can result in different outcomes.

Initialization Sensitivity: Different runs were observed to have varying levels of performance depending on the initial weight configuration of the neural network, both classical and quantum.

Environmental Stochasticity: The Gym Cartpole environment itself might have inherent variability affecting the reproducibility of results.

3.3.7.3 Implications Understanding the uncertainty and error in these metrics is vital for:

Model Tuning: Knowing where the model underperforms or is inconsistent aids in targeted improvements.

Generalisability: A model with less uncertainty is more likely to perform well in unseen scenarios.

Resource Allocation: For quantum RL algorithms, where computational resources are expensive, understanding error rates and variability can guide better resource utilization.

Algorithm Comparison: Error and uncertainty metrics can serve as additional dimensions for comparing the effectiveness of classical and quantum RL agents.

3.3.8 Summary

In this section, we have conducted a comprehensive analysis of our experiments involving both classical and quantum implementations of reinforcement learning (RL) algorithms—Policy Gradient REINFORCE, Deep Q Learning, and Actor Critic. A series of 12 experiments were designed to rigorously evaluate and compare these algorithms on various fronts.

Key Observations:

Average Reward Per Episode: Across all RL algorithms, classical REINFORCE implementation generally exhibited higher average rewards, suggesting superior and more consistent learning.

Learning Curve: Classical RL algorithms tended to learn faster but plateaued sooner, while their quantum counterparts showed slower initial learning but higher eventual rewards.

Convergence Time: The time required for convergence was highly variable, with quantum algorithms generally taking longer but achieving better rewards.

Computational Complexity: Quantum algorithms demonstrated higher computational complexity, especially as the number of layers increased.

Efficiency: While classical algorithms were computationally more efficient, quantum algorithms were more reward-efficient over time.

Comparative Analyses: Statistical tests like t-tests and ANOVAs revealed significant differences between classical and quantum RL agents, particularly in the learning curves for DQN and Actor Critic Experiments. A comparison between Quantum RL agents with different layer configurations showed that more layers generally contributed to better performance but at the cost of computational complexity.

Uncertainties and Errors: We analyzed the standard deviation of the mean and confidence intervals across multiple runs of each experiment. This in-depth analysis allowed us to understand the reliability and variability of our models better.

Implications: These findings have critical implications for both classical and quantum machine learning. Quantum RL algorithms, while computationally more complex, offer the potential for more robust and efficient policies in the long run. The uncertainties and error analyses performed help us to improve future models, allocate computational resources better, and contribute to the generalizability of our findings.

In summary, this Results section serves not just as a record of numerical outcomes but as an insightful guide for understanding the nuances between classical and quantum RL algorithms. The extensive analyses carried out will significantly contribute to the broader scientific discourse in this emerging domain.

4 Discussion of Results

The overarching aim of this dissertation was to investigate the capabilities and efficiencies of Quantum Reinforcement Learning (QRL) in comparison to its classical counterpart. Through a series of rigorous experiments, this research sought to answer the following primary questions:

1. How does the performance of Quantum Reinforcement Learning algorithms compare to that of classical Reinforcement Learning algorithms in a controlled environment, specifically the gym's cart-pole environment?

2. What are the computational trade-offs associated with implementing quantum algorithms over classical ones, particularly in terms of convergence time and computational

complexity?

3. Does the number of layers in quantum circuits significantly affect the performance of QRL algorithms?

By exploring these questions, this research aims not only to contribute new empirical findings to the emerging field of QRL but also to offer actionable insights for the broader scientific community.

4.1 Interpretation of Results

This section will concentrate on the interpretation of experimental results and will set the stage for the next section that discusses the comparisions with the existing research providing a comprehensive view without sacrificing clarity or focus.

4.1.1 Learning Curve

The learning curve analysis serves as a critical component in our exploration of classical and quantum reinforcement learning agents applied to the cartpole task. The primary objectives of this analysis are to examine the speed and stability of convergence, the impact of architectural layers on performance, and to identify any unique trends in quantum versions of these agents.

In classical and quantum Deep Q-Networks (DQN), the convergence ratio is noticeably less than that observed for Policy Gradient (REINFORCE) and Actor-Critic methods (as presented in Table 9). Specifically, the mean initial convergence point for DQN is statistically higher than other agents, a result that was consistent across both classical and quantum settings. Such observations suggest that DQNs may require an episode limit beyond our chosen 3000-episode threshold for stable convergence and that further hyperparameter tuning is essential. It's also indicative that DQNs may not be optimally suited for the cartpole task compared to other methods.

When examining Figures 2, 3, and 4, it becomes apparent that the mean reward curves for quantum agents do not follow a consistent pattern across all RL algorithms. However, 5-layer quantum agents for Policy Gradient and DQN methods exhibit a lower mean convergence point, indicating superior performance compared to their classical counterparts. This advantage dissipates when reducing the number of architectural layers. One consistent observation is the faster convergence rate achieved by increasing the number of layers in quantum RL agents, a finding substantiated across all tested RL algorithms.

The quantum version of the Actor-Critic algorithm presents a unique case. Contrary to expectations, it underperformed compared to its classical counterpart. A possible reason for this could be the structural complexities introduced in the quantum version. The quantum Actor-Critic employs separate quantum parameterized circuits (QPCs) for both actor and critic models, whereas the classical version utilizes a single model with shared parameters. For simpler tasks like the cartpole, it's plausible that a less complex quantum architecture would yield better performance, as evidenced by the significantly prolonged convergence time and noisier reward curves (Refer to Figure 7) in the quantum Actor-Critic methods.

As revealed in Figures 5, 6, and 7, the Deep Q-Networks (DQNs) in quantum settings show greater layer sensitivity compared to other RL algorithms. This suggests that additional layers in Quantum DQN contribute positively to training stability and lead to faster convergence rates.

Our findings indicate that increasing the number of layers in quantum RL agents correlates with better performance and quicker, more stable convergence. Furthermore, 5-layer quantum versions of Policy Gradient and DQN agents outperformed their classical analogs in terms of episode-based convergence. Conversely, the quantum Actor-Critic algorithm fell short of its classical counterpart, likely due to implementation discrepancies that added unnecessary complexity to the model.

4.1.2 Average Reward Per Episode

The metric of average reward per episode forms another essential pillar in our comparative evaluation of classical and quantum reinforcement learning (RL) agents. This particular metric provides insights into the quality of the learned policy, while still being sensitive to the variability of the rewards. It complements the learning curve data by offering a snapshot of agent effectiveness during their interaction with the cartpole environment.

When scrutinizing classical RL algorithms, Classical REINFORCE clearly stands out with a mean reward of 230.74, which is significantly higher than that of Classical DQN and Classical Actor-Critic. (Refer to Table 8) This high average suggests that policy-gradient methods, specifically REINFORCE, may be better suited for the cartpole task in a classical setting. Quantum RL agents show greater standard deviation values in their average rewards compared to their classical counterparts. For instance, the 5-layer Quantum REINFORCE agent has a standard deviation of 66.33, substantially higher than the Classical REINFORCE at 40.37. This variability may signify that quantum models, especially with more layers, might require more episodes to reach stable reward structures or that the inherent noise in quantum computation affects reward stability.

The mean reward for Quantum REINFORCE agents tends to decrease as the number of layers are reduced, with the 3-layer version recording a mean of just 190.9 compared to 222.27 for the 5-layer version. (Refer to Table 8) A noteworthy observation is that the Quantum Actor-Critic method with 3 layers exhibits an unusually high mean reward of 204.74, surpassing its classical version at 173.59. (Refer to Table 8) This marks one of the rare instances where a quantum agent surpasses its classical counterpart in this metric. This could be an avenue for further research into why specific quantum architectures may provide advantages in some performance metrics but not others.

In summary, the average reward per episode serves as an invaluable metric for assessing the efficacy of our agents. Policy gradient methods, particularly REINFORCE, appear to be optimal for the cartpole task in a classical context. Quantum agents introduce higher variability in rewards, which could be a consequence of quantum noise or architectural complexities. The 3-layer Quantum Actor-Critic model provided an interesting exception by outperforming its classical counterpart in this metric which would indicate more consistant learning albeit a noisy one, warranting further investigation.

4.1.3 Convergence Time

The convergence time metric represents the average number of episodes it takes for each algorithm to reach a stable performance level. This measure serves as an indicator of the efficiency and computational expediency of both classical and quantum RL algorithms, adding another layer to our comprehensive evaluation.

Among classical algorithms, the Classical Actor-Critic model exhibits the fastest convergence time with a mean of 343 episodes. (Refer to Table 10) This is particularly striking

when compared to Classical DQN, which takes substantially longer with a mean convergence time of 1862.5 episodes. This implies that the Actor-Critic approach not only performs well in terms of average reward but also converges faster in a classical setting.

For quantum RL algorithms, a noteworthy trend is the inverse relationship between the number of layers and the mean convergence time, especially for Quantum REINFORCE. The 5-layer Quantum REINFORCE converges faster, with a mean time of 600 episodes, compared to its 3-layer counterpart at 880 episodes. (Refer to Table 10) This indicates that adding complexity in the form of additional layers could, counter intuitively, lead to quicker convergence in quantum RL algorithms.

High standard deviation values in the Quantum REINFORCE 3-layer model (654.2) and Quantum DQN 5-layer model (649.03) suggest that these configurations might suffer from unstable convergence times, potentially due to the complexities associated with quantum architectures or noise. In contrast, Classical Actor-Critic shows lower standard deviation (77.21), indicating that its convergence time is relatively more stable. (Refer to Table 10)

Classical Actor-Critic not only outperforms its quantum versions in terms of speed but also shows more stable convergence, as evidenced by the lower standard deviation. However, the 5-layer Quantum REINFORCE model does manage to converge faster than its classical counterpart, raising intriguing questions about scenarios where quantum algorithms could offer advantages in terms of convergence efficiency.

In summary, the convergence time metric further refines our understanding of the strengths and weaknesses of each algorithm. Classical Actor-Critic emerges as a robust model in terms of both speed and stability. Among the quantum algorithms, added layers appear to contribute to faster convergence but may also introduce variability in the time required. Importantly, Quantum REINFORCE with 5 layers provides a rare example where a quantum algorithm converges faster than its classical version.

4.1.4 Computational Complexity

Computational complexity, represented by the mean time taken during the training phase, serves as an important metric for assessing the practical feasibility of deploying various RL algorithms. This measure provides insight into the trade-offs between performance metrics like average reward and convergence time versus the computational cost involved.

Among the classical approaches, Classical REINFORCE shows remarkable efficiency with a mean training time of 21.26 seconds. This contrasts sharply with the Classical DQN, which requires a mean time of 4485.15 seconds, a staggering increase. The Classical Actor-Critic algorithm takes a moderate amount of time (153.62 seconds), positioning it as a balanced choice in terms of computational burden and performance. (Refer to Table 11)

In the quantum realm, the number of layers does not show a straightforward correlation with computational complexity. For instance, Quantum REINFORCE with 5 layers takes less time (390.25 seconds) compared to its 3-layer version (402.2 seconds). (Refer to Table 11) This suggests that the increase in computational time is not linearly proportional to the complexity in terms of layers, thus opening new avenues for optimizing quantum RL algorithms.

The standard deviation in computational time for quantum algorithms, especially Quantum REINFORCE 3-layer (397.65) and Quantum DQN 5-layer (1667.12), indicates high variability. This could be attributed to the intricacies of quantum computation or environmental noise, which might necessitate further investigation. Conversely, Classical REINFORCE has the lowest standard deviation (6.78), underlining its consistent computational efficiency. (Refer to Table 8)

While Quantum DQN models show a lower mean training time compared to their classical counterpart, they pay the price in terms of high variability, as indicated by the standard deviation. Quantum Actor-Critic models, on the other hand, demand an exponentially higher computational time, making them less suitable for real-time applications at this stage. Synthesis and Conclusions

The computational complexity metric illuminates the practical implications of choosing one algorithm over another, adding a crucial dimension to our evaluation framework. Classical algorithms, notably REINFORCE and Actor-Critic, offer a good balance of performance and computational efficiency. Among quantum algorithms, the absence of a clear correlation between the number of layers and computational time provides fertile ground for future optimization research.

4.1.5 Efficiency

Efficiency, defined as the ratio of total rewards to training time, offers a composite evaluation criterion that amalgamates the speed of learning with the quality of solutions. By examining efficiency, one can understand the holistic merits and demerits of classical and quantum RL algorithms, especially when making decisions regarding deployment in resource-sensitive environments.

Among classical algorithms, Classical REINFORCE takes the lead with an astounding mean efficiency of 8578.41, showcasing that despite its relatively quick training time, it accrues substantial rewards. Classical Actor Critic falls in the middle-ground with a mean efficiency of 378.21, making it a relatively balanced option. Classical DQN, however, languishes at the bottom with a mean efficiency of just 52.13, suggesting it may not be the most optimal choice when time and performance are both constraints. (Refer to Table 12)

Within the quantum paradigm, Quantum REINFORCE with fewer layers (3-layer version) posts a higher mean efficiency (452.44) compared to its more complex 5-layer counterpart (355.75). This inversion, where fewer layers result in higher efficiency, brings forth important implications for the architecture selection in quantum RL algorithms. Stability Across Models. (Refer to Table 12)

The standard deviation in efficiency for the quantum models, especially Quantum Actor Critic variants, is remarkably low (ranging from 0.33 to 1.12), which suggests a consistent level of efficiency across episodes. This stability is an asset when considering the deployment of these algorithms in scenarios requiring reliable performance. (Refer to Table 12)

Quantum DQN models, despite their longer convergence times, show promise with mean efficiencies greater than Classical DQN but are still outshined by Quantum REINFORCE models. Meanwhile, Quantum Actor Critic algorithms have the lowest efficiency among all, which, coupled with their high computational times, makes them least favorable for immediate practical implementations especially in relatively simple environments.

Efficiency emerges as an indispensable metric that encapsulates the balance between reward maximization and computational expediency. Classical REINFORCE stands out as highly efficient but is closely followed by simpler Quantum REINFORCE models, presenting a compelling case for their practical application.

4.1.6 Inter-connectedness of Metrics: A Holistic View

The various metrics employed in this study—learning curve, average reward per episode, convergence time, computational complexity, and efficiency—are not isolated indicators but rather

interconnected facets that together provide a comprehensive understanding of the performance of classical and quantum RL algorithms.

Learning Curve and Average Reward: The learning curve, which primarily focuses on the speed and stability of convergence, has a direct relationship with the average reward per episode. A smoother learning curve often correlates with a higher average reward, as it indicates that the agent is learning to make better decisions over time. For instance, the faster convergence rates in 5-layer Quantum REINFORCE agents correspond with higher mean rewards, underlining the efficacy of these models. (Refer to Tables 9 and 8)

Convergence Time and Computational Complexity: These two metrics are intrinsically linked. Faster convergence doesn't always translate to lower computational time. For example, while Classical Actor-Critic models converge quickly, their computational time is moderate, indicating that the algorithm may involve complex computations that require more time per episode. (Refer to Tables 10 and 11)

Efficiency and Computational Complexity: Efficiency, defined as the ratio of total rewards to training time, serves as a bridge between performance and computational cost. An algorithm that takes less time to train (low computational complexity) but yields high rewards is naturally more efficient. This is evident in the case of Classical REINFORCE, which despite its quick training time, accrues substantial rewards, leading to high efficiency. (Refer to Tables 12 and 11)

Learning Curve and Convergence Time: A smoother learning curve often leads to faster convergence times. However, this is not always the case in quantum settings, where the 5-layer Quantum REINFORCE model shows a smoother learning curve but takes longer to converge compared to its 3-layer counterpart. This suggests that additional layers, while beneficial for learning, may introduce complexities that affect convergence time. (Refer to Tables 9 and 10)

Average Reward and Efficiency: While a higher average reward is generally indicative of a more effective algorithm, it must be viewed in the context of efficiency for a complete picture. For example, Quantum Actor-Critic models, despite their lower average rewards, show consistent efficiency levels, making them reliable if not the most effective. (Refer to Tables 8 and 12)

Standard Deviations Across Metrics: High standard deviations in metrics like average reward and convergence time in quantum models may indicate that these models are more sensitive to initial conditions or hyperparameters. This interconnected variability could be a focus for future research to understand the stability of quantum RL algorithms. (Refer to Tables 8 and 10)

By examining these metrics in tandem rather than in isolation, we gain a more nuanced understanding of the strengths and weaknesses of each algorithm. This interconnected view allows for better-informed decisions when selecting an RL algorithm for specific tasks, balancing performance, computational cost, and reliability.

4.2 Comparison with Existing Research

The findings of this dissertation contribute to the growing body of research on Quantum Reinforcement Learning (QRL) and its classical counterparts. This section aims to compare and contrast the results obtained in this study with previously published work in the domains of classical and quantum RL.

4.2.1 Performance Metrics

Learning Curve and Convergence: Our results indicate that Deep Q-Networks (DQNs) may not be optimally suited for the cartpole task, a finding that aligns with existing literature suggesting that DQNs are more effective in high-dimensional state spaces [9]. The faster convergence rates observed in quantum RL agents with increased layers also find some support in the literature, which posits that quantum algorithms can offer computational advantages in specific settings [15].

Average Reward Per Episode: The high average reward for Classical REINFORCE in our study is consistent with existing research that highlights the effectiveness of policy-gradient methods in solving the cartpole problem [6]. The variability in rewards for quantum agents, particularly for Quantum REINFORCE, could be attributed to the inherent noise in quantum computation, a challenge also noted in other quantum computing studies [33].

Convergence Time: The faster convergence time for Classical Actor-Critic models in our study is in line with existing research that suggests Actor-Critic methods are efficient in terms of both sample complexity and computational time [27]. The faster convergence of 5-layer Quantum REINFORCE compared to its classical counterpart is a novel finding that warrants further investigation, given that quantum advantages in RL are still a subject of ongoing research [14].

Computational Complexity: Our findings regarding the computational time required for Classical DQN align with the literature, which often cites the high computational cost of DQNs as a drawback [9]. The absence of a linear relationship between the number of layers and computational time in quantum RL algorithms is a novel observation that could be explored further in future studies.

Efficiency: The efficiency metric used in our study offers a holistic view of the algorithms' performance, a measure that has been less commonly used in existing research. Our results show that Classical REINFORCE is highly efficient, a finding that could be compared to studies that measure efficiency in terms of sample complexity [10].

4.2.2 Consistencies and Inconsistencies

One of the key consistencies between our findings and existing research is the effectiveness of policy-gradient methods, particularly REINFORCE, in solving the cartpole problem. However, an inconsistency lies in the performance of Quantum Actor-Critic methods, which underperformed in our study compared to their classical counterparts. This could be due to the added complexity in our quantum models, a factor less commonly considered in classical RL research. The inconsistencies in quantum RL performance could be attributed to several factors, including the architectural complexities and the inherent noise in quantum computation. These are areas where existing research is still sparse, and our study could serve as a basis for further exploration.

Our study adds to the existing body of knowledge by providing empirical evidence on the performance of classical and quantum RL algorithms in a controlled environment. While our findings corroborate several aspects of existing research, they also present new avenues for exploration, particularly in the realm of quantum RL. Future research could focus on mitigating the challenges observed in quantum RL algorithms to harness their full potential.

4.3 Strengths and Limitations

4.3.1 Strengths

The dissertation presents a robust and comprehensive analysis of both classical and quantum reinforcement learning algorithms, employing a multi-faceted approach that adds depth and rigor to the study. One of the key strengths lies in the experimental design, which encompasses three fundamental RL algorithms—Policy Gradient REINFORCE, Deep Q Learning, and Actor-Critic—in both classical and quantum settings. This broad scope allows for a nuanced comparison and understanding of each algorithm's performance metrics. Complementing this is the rigorous statistical analysis employed, including t-tests and ANOVA for group comparisons, as well as effect size and power analysis. These statistical methods lend a high degree of scientific rigor and confidence to the study's findings.

Another significant strength is the multi-metric evaluation approach. The study doesn't rely solely on a single performance indicator but incorporates a range of metrics such as average reward per episode, learning curve, and computational complexity. This holistic view is further enriched by detailed descriptive statistics, including mean, median, and standard deviation, offering a thorough understanding of data distribution and performance consistency across different runs. In the realm of quantum RL, the layer-wise analysis stands out. By testing quantum algorithms with varying numbers of layers, the study provides valuable insights into the scalability and efficiency of these algorithms in quantum settings. Although the experiments were conducted in a simulated environment using TensorFlow Quantum, this closely mimics real-world conditions, adding another layer of credibility to the findings.

4.3.2 Limitations and Self-Criticism

However, the study is not without its limitations. One notable constraint is the limited range of RL algorithms examined. While the study does cover three key algorithms, the exclusion of others like Monte Carlo methods and Temporal Difference learning leaves room for a more exhaustive comparison. Additionally, the use of a simulated quantum environment, though advantageous for control, may not capture all the nuances of a real quantum system. Computational and time constraints also limit the study's scope. The restricted number of episodes and runs could affect the accuracy of the algorithms' performance evaluation, and the limited time for research could have impacted the depth of analysis and range of algorithms tested.

Further, the study's quantum models come with their own set of assumptions, particularly concerning the reuploading Quantum Parametric Circuits (QPC). These assumptions may not hold in real-world quantum computing environments, affecting the generalizability of the results. Lastly, the experiments were conducted in the Gym CartPole environment, a standard but simplified benchmark that may not fully represent the complexities of real-world applications. By acknowledging these strengths and limitations, the dissertation aims to contribute meaningfully to the scientific rigor and integrity of the field of Quantum Reinforcement Learning, setting the stage for future research that can address these gaps.

4.4 Suggestions for Future Work

Future research in this area can be broadly categorized into several key directions. First, the scope of RL environments should be expanded beyond the cartpole task to include more complex scenarios like robotic manipulation, autonomous navigation, and multi-agent systems. This would offer a more rigorous evaluation of both classical and quantum RL algorithms.

Alongside, there's a need to delve into advanced quantum models such as Quantum Variational Circuits or Quantum Generative Models, which could potentially outperform classical algorithms across a broader spectrum of metrics.

Hyperparameter optimization also emerges as a critical area for future exploration. Techniques like grid search or Bayesian optimization could be employed to fine-tune algorithms, particularly DQN, for faster convergence and higher rewards. This is closely related to the study's findings on the importance of architectural layers in quantum RL algorithms. A focused analysis on layer sensitivity could yield nuanced insights into how complexity impacts performance, thereby informing better design choices.

Another significant avenue is the investigation into noise mitigation techniques for quantum RL algorithms. Given the observed variability in rewards and convergence times, understanding and mitigating the impact of quantum noise could make these algorithms more robust and consistent. This is especially crucial for their deployment in real-world, resource-sensitive applications, where computational efficiency is paramount. In this context, the study's introduction of an efficiency metric combining reward and computational time is noteworthy. Future work could extend this by developing composite metrics that also consider stability, robustness, and adaptability for a more holistic evaluation.

Lastly, interdisciplinary applications present an exciting frontier. Testing these algorithms in domains like quantum chemistry or financial modeling could not only validate their efficacy but also lead to significant breakthroughs in those respective fields. By addressing these avenues, subsequent research could significantly contribute to the development of more efficient, robust, and versatile quantum RL algorithms for a wide array of practical applications.

4.5 Theoretical and Practical Implications

This dissertation elucidates pivotal insights with both theoretical and practical ramifications in the sphere of quantum reinforcement learning (RL). Theoretically, the study unveils a non-linear relationship between architectural complexity and efficiency, challenging the prevailing notion of a universally optimal algorithm. This discovery prompts a reconsideration of "optimal complexity," varying by tasks and environments, and introduces a composite metric for efficiency—melding reward and computational time. Such a unified framework not only refines the criteria for algorithm evaluation but also expands our understanding of quantum versus classical algorithms. Quantum algorithms excel in specialized tasks but are not universally superior, a nuanced finding that further delineates the applicability of quantum algorithms, including how they interact with elements of stochasticity and noise.

Practically, the unique attributes of quantum RL have transformative potential in high-stakes applications such as drug discovery and material science, areas where exploration of expansive combinatorial spaces is fundamental. The efficiency metric becomes particularly relevant in resource-limited contexts like edge computing and Internet Of Things (IoT), where computational expediency is at a premium. Beyond efficiency, the variability inherent in quantum RL, possibly stemming from quantum noise, offers robustness, making these algorithms suitable for applications in noisy settings like robotics. Customization of algorithms, through architectural layers and hyperparameter tuning, is shown to be a key factor for optimal performance, guiding businesses and researchers in a more task-centric algorithm deployment. As quantum computing continues to evolve, a transitional, hybrid approach employing both classical and quantum algorithms offers a practical route, anticipating an increasingly favorable computational cost-to-benefit ratio for quantum RL in solving complex, real-world challenges.

5 Conclusions

In this dissertation, we embarked on an exploratory journey to assess the performance and computational trade-offs of Quantum Reinforcement Learning (QRL) algorithms in comparison to their classical counterparts, specifically in the context of the gym's cart-pole environment. Our findings suggest that while classical algorithms like REINFORCE and Actor-Critic offer a balanced combination of performance and computational efficiency, quantum versions present a more nuanced picture. Quantum REINFORCE and DQN models with 5 layers demonstrated superior performance in terms of episode-based convergence, albeit with higher variability in rewards. Conversely, Quantum Actor-Critic algorithms lagged behind their classical versions, likely due to the added complexities in their quantum architectures.

The study also delved into the impact of number of layers in quantum circuits on the performance of QRL algorithms. We observed that increasing the number of layers generally led to faster and more stable convergence, although it did not consistently improve other performance metrics like average reward per episode. This opens up avenues for future research to optimize the architectural complexities in quantum RL algorithms. Additionally, the computational complexity metric revealed that while quantum algorithms could potentially offer faster training times, they also introduce higher variability, which could be attributed to quantum noise or other intricacies in quantum computation.

In summary, our research provides a comprehensive evaluation of classical and quantum RL algorithms, shedding light on their strengths, weaknesses, and areas for improvement. While classical algorithms offer robust and efficient solutions, quantum algorithms, particularly those with more layers, show promise in specific performance metrics. However, they also introduce new challenges such as higher variability and computational complexity. Future work should focus on optimizing the quantum architectures for better performance and stability, as well as exploring the real-world applicability of these algorithms in more complex environments. This study serves as a foundational step in understanding the potential and limitations of Quantum Reinforcement Learning, setting the stage for more in-depth investigations in this emerging field.

References

[1] Joshua Vendrow, Glen Meyerowitz, and Rahul Malavalli. Ece239as (rl) final report: Ppo implementation for openai environments.

[2] P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[3] A. M. Steane. Multiple particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 452(1954):2551–2577, 1996.

[4] L. W. Kline and E. L. Thorndike. Animal intelligence: An experimental study of the associative processes in animals. *The American Journal of Psychology*, 10:149, 10 1898.

[5] F. Nowell Jones and B. F. Skinner. The behavior of organisms: An experimental analysis. *The American Journal of Psychology*, 52:659, 10 1939.

[6] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[7] Christopher J. C. H. Watkins. Learning from delayed rewards. 1989.

[8] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

[9] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. volume 518, pages 529–533, 2015.

[10] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[11] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32:1238–1274, 08 2013.

[12] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 10 2017.

[13] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[14] Daoyi Dong, Chunlin Chen, Hanxiong Li, and Tzyh-Jong Tarn. Quantum reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38:1207–1220, 2008.

[15] Vedran Dunjko, Jacob M. Taylor, and Hans J. Briegel. Quantum-enhanced machine learning. *Physical Review Letters*, 117:130501, 2016.

[16] Gael Sentís, John Calsamiglia, Giulio Chiribella, and Ramón Muñoz-Tapia. Quantum learning of coherent states. *EPJ Quantum Technology*, 4:6, 2017.

[17] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5, 07 2014.

[18] Adrián Pérez-Salinas, Alba Cervera-Lierta, Elies Gil-Fuster, and José I. Latorre. Data re-uploading for a universal quantum classifier. *Quantum*, 4:226, 02 2020.

[19] Dawid Kopczyk. Quantum machine learning for data scientists. *arXiv (Cornell University)*, 04 2018.

[20] Carlo Ciliberto, Mark Herbster, Alessandro Davide Ialongo, Massimiliano Pontil, Andrea Rocchetto, Simone Severini, and Leonard Wossnig. Quantum machine learning: a classical perspective. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474:20170551, 01 2018.

[21] Thomas Lubinski, Carleton Coffrin, Catherine C McGeoch, Pratik Sathe, Joshua Apanavicius, and David E Bernal. Optimization applications as quantum performance benchmarks. *arXiv (Cornell University)*, 02 2023.

[22] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. 06 2016.

[23] Gymnasium documentation.

[24] Yazhou Hu, Fengzhen Tang, Jun Chen, and Wenxue Wang. Quantum-enhanced reinforcement learning for control: a preliminary study. *Control Theory and Technology*, 19:455–464, 11 2021.

[25] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv (Cornell University)*, 12 2013.

[26] Jan Peters, Sethu Vijayakumar, and Stefan Schaal. Natural actor-critic. pages 280–291, 10 2005.

[27] Vijay R. Konda and John N. Tsitsiklis. Onactor-critic algorithms. *SIAM Journal on Control and Optimization*, 42:1143–1166, 01 2003.

[28] Keras documentation: Actor critic method.

[29] Nico Meyer, Christian Ufrecht, Maniraman Periyasamy, Daniel D Scherer, Axel Plinge, and Christopher Mutschler. A survey on quantum reinforcement learning. *arXiv (Cornell University)*, 11 2022.

[30] Parametrized quantum circuits for reinforcement learning — tensorflow quantum.

[31] Samuel Yen-Chi Chen, Chao-Han Huck Yang, Jun Qi, Pin-Yu Chen, Xiaoli Ma, and Hsi-Sheng Goan. Variational quantum circuits for deep reinforcement learning. *IEEE Access*, pages 1–1, 2020.

[32] Alberto Acuto, Paola Barillà, Ludovico Bozzolo, Matteo Conterno, Mattia Pavese, and A Policicchio. Variational quantum soft actor-critic for robotic arm control. *arXiv (Cornell University)*, 12 2022.

[33] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 08 2018.