International Journal of Advanced Technology and Engineering Exploration, Vol 11(121)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2024.111100094

Research Article

Quantum prioritized experience replay with MaDi-based priority and quantum circuit mechanisms for optimizing reinforcement learning

R. Palanivel* **and P. Muthulakshmi**
Department of Computer Science, Faculty of Science and Humanities, SRM Institute of Science and Technology, Kattankulathur, Chennai, Tamil Nadu, India

Received: 24-January-2024; Revised: 23-December-2024; Accepted: 25-December-2024

Abstract
Reinforcement learning (RL) encounters significant challenges related to scalability and computational efficiency, particularly in complex decision-making environments. Traditional RL algorithms often struggle with large-scale tasks, necessitating innovative approaches to enhance adaptability and performance. The quantum circuit-based priority replay (QCPR) algorithm was introduced in this study, leveraging quantum-inspired techniques to enhance learning efficiency and decision-making quality through quantum computing (QC). The QCPR algorithm implements two key mechanisms for prioritizing experiences: the magnitude and direction (MaDi) based priority, which assesses the significance and directionality of experiences, and QCPR-specific methods that ensure efficient experience replay. These mechanisms are seamlessly integrated into the Q-learning (QL) framework to optimize the RL process for superior performance. QCPR was evaluated against four well-established RL algorithms-QL, deep Q-networks (DQN), prioritized experience replay (PER), and dueling DQN (DDQN)-using standard decision-making benchmarks. The algorithm was further tested in various simulation and real-world environments, including Atari games, Qiskit integration, and Raspberry Pi hardware and software setups. These evaluations demonstrated QCPR's adaptability and robustness, showcasing its capability for dynamic, large-scale applications. The results revealed that QCPR significantly outperforms its counterparts across key performance metrics, including accuracy, precision, recall, F1-score, and cumulative rewards. Specifically, QCPR achieved a 15.29% improvement in accuracy over QL, a 9.98% increase over DQN, a 6.33% improvement over PER, and an 8.23% enhancement over DDQN. This study highlights the potential of quantum-inspired approaches to advance RL, offering scalable and efficient solutions for complex decision-making tasks.

Keywords
Quantum computing, MaDi priority, Quantum circuit-based priority replay, Quantum reinforcement learning, Quantum priority experience reply.

1.Introduction
Reinforcement learning (RL) is a crucial component of artificial intelligence (AI), enabling agents to learn optimal policies through trial and error in dynamic environments. This approach has been successfully applied to various domains, such as energy management in microgrids [1] and addressing real-world challenges [2]. Despite significant advancements, traditional RL algorithms face numerous challenges, including slow convergence, inefficiencies in managing experience replay, and limited scalability in complex environments [3–5].

In response to these limitations, quantum computing (QC) has emerged as a powerful tool to enhance RL's computational efficiency by leveraging quantum parallelism and superposition [6–8].

Recent research has shown that quantum-inspired approaches can improve the exploration and exploitation balance in RL, leading to faster learning and more efficient decision-making [9]. This paper introduces the quantum circuit-based priority replay (QCPR) algorithm, which integrates quantum circuits into prioritized experience replay (PER) mechanisms, addressing the limitations of traditional RL algorithms in terms of learning efficiency and scalability [10, 11]. The QCPR algorithm uses quantum principles to enhance computational power,

*Author for correspondence

optimize experience prioritization, and improve learning outcomes in complex environments [12–14].

Traditional RL algorithms encounter significant difficulties in managing prioritized experiences effectively, especially in large-scale and highly dynamic environments [15,16]. The PER mechanism addresses these challenges by assigning higher priority to more informative experiences [17]. However, PER's performance is still limited in environments where data is vast and rapidly changing [18]. Additionally, conventional RL models often struggle with slow convergence rates, which hinder their applicability in time-sensitive scenarios [19]. The QCPR algorithm leverages quantum circuits to implement an innovative method for improving the prioritization and processing of experiences, resulting in enhanced learning speed and accuracy. By utilizing quantum superposition and entanglement, quantum circuits offer a significant advantage over classical algorithms, enabling faster computations and more efficient exploration of the solution space [20].

The primary objective is to introduce the QCPR algorithm, which combines quantum circuits with prioritized replay mechanisms to improve the performance of RL in complex and dynamic environments. The algorithm's effectiveness is evaluated by comparing it with traditional RL methods, emphasizing metrics such as learning speed, decision-making accuracy, and computational efficiency. This research significantly advances RL through three key contributions: the QCPR algorithm, which integrates quantum circuits into the prioritized replay framework; the magnitude and direction (MaDi) priority equation, introducing a novel method for experience prioritization; and experimental validation, which demonstrates QCPR's superior learning efficiency and cumulative rewards compared to classical RL methods. Practical applicability is showcased through implementation on the Qiskit platform and Raspberry Pi in autonomous vehicle real-world scenarios.

Figure 1 illustrates the workflow of the QCPR algorithm, starting with the implementation of the quantum transfer-based fractal priority replay mechanism, followed by the execution of the QCPR algorithm, and concluding with its evaluation process.

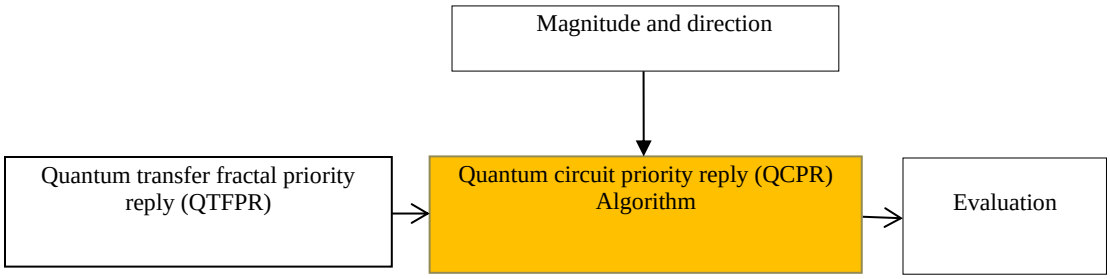

Figure 1 Workflow of QCPR algorithm and evaluation framework

The QTFPR algorithm is fundamental to this process and it combined with MaDi to define the behaviour of the proposed QCPR algorithm. The evaluation phase tests the QCPR algorithm with respect to convergence rate, rewards, experience and priority. The performance of the robotic device estimated based on the range of accuracy of decisions made in the complex environment.

This paper is organized as follows: Section 2 provides a review of related work on RL and quantum enhancements. Section 3 outlines the methods and describes the QCPR algorithm. Section 4 presents the experimental results and discusses the implications of the findings. Finally, Section 5 concludes the study.

2.Literature review

Quantum reinforcement learning (QRL) has rapidly evolved, with several recent studies exploring its various dimensions. Sannia et al. (2023) proposed a hybrid classical-quantum approach to speed up Q-learning (QL), effectively leveraging both classical and QC strengths to enhance learning efficiency [21]. Skolik et al. (2023) investigated the robustness of QRL under hardware errors, addressing a critical challenge for practical quantum implementations and ensuring reliability in real-world applications [22]. Chen (2023) explored asynchronous training of QRL, highlighting the benefits of parallel processing in quantum environments, which led to faster convergence and improved performance [23]. Li et al. (2023) introduced quantum attention pooling

(QAP), a quantum-inspired model for multimodal emotion recognition, showcasing the potential of quantum techniques in natural language processing and affective computing, thus broadening the applicability of QRL in understanding human emotions [24]. Tian et al. (2023) provided a comprehensive review of recent advances in quantum neural networks for generative learning, emphasizing their applications across various domains, such as image generation and data synthesis [25]. Chen (2023) presented quantum deep QL with distributed PER, aiming to enhance learning efficiency by prioritizing experiences based on their importance [26]. Dong et al. (2008) offered an overview of QRL, covering its theoretical foundations and potential applications, which contextualized advancements in the field [27]. Lamata (2023) discussed various quantum machine learning (QML) implementations, providing insights into practical realizations of quantum algorithms that solved complex problems [28]. Metz and Bukov (2023) utilized RL with tensor networks for self-correcting quantum many-body control, demonstrating how QML managed complex quantum systems [29]. Chen (2023) further investigated quantum deep recurrent RL, combining QC with recurrent neural networks for sequential decision-making tasks, which could lead to more effective algorithms in dynamic environments [30].

While not directly related to QML, Awad and Fraihat (2023) focused on feature selection methods for intrusion detection systems, using traditional machine learning techniques to enhance cybersecurity [31]. Similarly, Sharma et al. (2024) analyzed renewable energy systems using classical machine learning, providing insights into improving energy reliability and sensitivity [32]. Ghintab and Hassan (2023) explored deep learning networks for localization in self-driving vehicles, demonstrating the effectiveness of classical techniques in autonomous driving [33]. Li et al. (2024) introduced a PER method based on dynamics priority for RL, which indirectly benefited QRL by enhancing learning efficiency [34]. Wang and Xiang (2024) proposed a quantum optimization algorithm utilizing multistep quantum computation, demonstrating the potential of quantum techniques for solving optimization problems essential to many machine learning applications [35]. Roik et al. (2024) optimized routing in quantum communication networks using RL, enhancing quantum information transmission efficiency and reliability [36]. Vandelli et al. (2024) evaluated the practicality of quantum optimization algorithms in industrial applications, showcasing how QML addressed optimization challenges relevant to industry [37]. Patil et al. (2024) proposed noisy intermediate-scale quantum (NISQ)-friendly measurement-based quantum clustering algorithms, illustrating how quantum techniques were applied to data clustering tasks, making QML more accessible [38]. Zhao et al. (2022) focused on a multi-agent communication algorithm based on state control, which, while innovative, lay outside the direct realm of QML [39]. Palanivel and Muthulakshmi (2024) explored error mitigation with a quantum neural Q network (QNQN) for secure qutrit distribution, which was relevant to QC but did not directly address QML algorithms [40]. The reference by Mangla et al. (2024) focused on quantum key distribution (QKD), which utilized quantum mechanics for secure communication but was not directly related to QML [41]. Kundu et al. (2024) proposed RL techniques to enhance variational quantum state diagonalization, a crucial task in optimizing quantum algorithms [42]. Yun et al. (2023) developed quantum multi-agent actor-critic neural networks for coordinating internet-connected multirobots in smart factories, demonstrating the applicability of QML in industrial settings [43]. Finally, Peral-garcía et al. (2024) conducted a systematic literature review on QML and its applications, providing a comprehensive overview that served as a valuable resource for researchers in the field [44], while Hellstern et al. (2024) explored quantum computer-based feature selection, demonstrating the potential of quantum techniques to enhance feature engineering and improve model performance [45].

The research gaps identified in recent studies emphasize several critical areas for advancement. These include improving efficiency in RL, addressing optimization challenges in QML, and enhancing the reliability and performance of quantum communication networks. Practical applications of quantum optimization algorithms in industry and increasing the accessibility of quantum clustering techniques remain key priorities. Other gaps include extending QML to multi-agent systems, developing effective error mitigation strategies in QC, and refining variational quantum algorithms (VQA).

3.Methods

3.1Dataset and parameter description

The dataset captures the performance of the RL model utilizing the QCPR algorithm across 500 episodes, generated through Atari game simulations. These simulations were conducted in the Department of Computer Science, Faculty of Science and

Humanities, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu, India, in the free and open source software (FOSS) lab. The experiments were performed on a high-performance computing system with the following specifications: Intel Core i7 processor, 32GB RAM, NVIDIA RTX 3080 graphics processing unit (GPU), and 1TB SSD. Factors such as total reward, priority, convergence speed, experience, and learning rates were carefully controlled to ensure consistency. Additionally, some episodes were conducted in a custom-designed autonomous vehicle environment.

Each episode served as a trial in which the agent undertook tasks such as navigation, obstacle avoidance, and route planning to achieve specific goals. The model's performance was evaluated using key metrics, including total rewards and task completion rates. These metrics illustrate the impact of the QCPR algorithm on the agent's learning, particularly in real-world driving scenarios. Parameters related to task performance in the autonomous vehicle environment were calculated to provide further insights.

- Total rewards (range: 27–94): The cumulative rewards the agent earns during an episode, reflecting the agent's performance.
- Experience (range: 27–94): The total number of interactions or steps the agent has encountered during the learning process.
- Priority (range: 0.38–5.23): The weight or importance assigned to each experience for replay, influencing how it is sampled during training.
- Convergence speed (range: 10–26): The number of iterations required for the model to stabilize its learning process.
- Exploration rate (0.011): The likelihood of the agent selecting random actions to explore new strategies or states.
- Decay rate (range: 1–0.2): The rate at which the exploration probability decreases over time as the model becomes more confident.

The dataset simulated in this study is stored in a general repository, which can be accessed through the following link: https://github.com/PALANIVELAADHI/DATASET FOR_IJATEE.

3.2Conceptual architecture

The QCPR algorithm seamlessly integrates quantum circuits with priority replay techniques to enhance RL efficiency [46]. Utilizing the MaDi priority equation, the agent interacts with the environment, continuously updating a priority-based replay buffer [47, 48]. This process involves storing prioritized actions in replay memory, fine-tuning the quantum circuit based on these priorities, and optimizing decision-making through repeated interactions with the environment, which provides states and rewards (*Figure 2*).

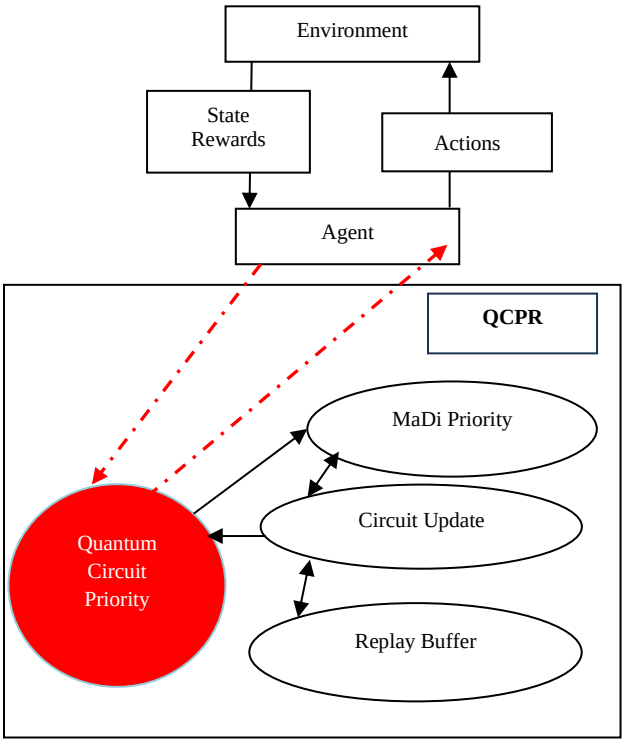

Figure 2 Conceptual architecture

3.3Working principle

Leveraging QC principles, this dual approach enhances adaptability and accelerates learning in reinforcement tasks [49]. The QCPR algorithm computes priority in two ways: 1) MaDi-based priority and 2) QCPR. MaDi-based priority evaluates the significance and directionality of experiences, while QCPR ensures efficient experience replay in RL.

3.3.1MaDi based priority

The PER method proposed by [50] for classical computers should be adapted and implemented on quantum computers. Accordingly, the PER equation is provided in Equation 1.

$$p_i = |\delta_i| + \epsilon \quad (1)$$

General description

In general, the absolute value of a number |x| is defined as shown in Equation 2:

$$|x| = \begin{cases} x, & x \geq 0 \\ -x, & x < 0 \end{cases} \quad (2)$$

Magnitude, denoted as |x|, measures the distance of a real number x from the origin (0) on the number line. For non-negative values ($x \geq 0$), $|x|$ equals x; for negative values ($x < 0$), it represents x transformed to its positive equivalent.

Equation 2 defines the sign function which is as under:

$$sign(x) = \begin{cases} 1, & x < 0 \\ 0, & x - 0 \\ -1, & x < 0 \end{cases} \quad (3)$$

The sign function indicates the direction of a real number x in Equation 3
For positive values ($x > 0$), it returns 1, for zero ($x = 0$), it returns 0, and for negative values ($x < 0$), it returns -1, signifying the respective directions.

Quantum description
In the context of QRL, the absolute value function |x| and the sign function sign(x) are utilized to handle the temporal difference (TD) error (δ) for prioritizing experiences within a replay buffer.

Calculating |δ| (Absolute value of TD error):
Compute the TD error:

$$\delta = R + \gamma \max_a Q(s', a) - Q(s, a) \quad (4)$$

Apply the absolute value function to δ:

$$|\delta| = \delta_s ign(\delta) \quad (5)$$

The resulting |δ| is used to prioritize experiences in the replay buffer, with larger |δ| values signifying greater importance for learning as shown in Equations 4 and 5 [51, 52].

Determining sign (δ) (Sign of TD error):

Determining sign (δ) is crucial for distinguishing reward overestimation and underestimation by the agent [52],
If δ > 0, sign (δ) = 1 (underestimated the reward)
If δ = 0, sign (δ) = 0 (matches the actual reward)
If δ < 0, sign (δ) = -1 (overestimated the reward)

Sign (δ) is then applied in Equation 9 (MaDi equation) to adjust experience priorities based on the direction of the TD error [53]. From the Equation 2, the quantum system absolute value calculated, that is |δ|, is the absolute value of the TD error. This term ensures that experiences with a large TD error are given a high priority [54].

Substituting it into Equation 1 results in Equation 6:

$$P = \delta \cdot \text{sign}(\delta) + \epsilon_1 \cdot \left(1 + \text{sign}(\delta) \cdot \delta_{\text{weight}}\right)\epsilon_2 \quad (6)$$

Distributing ε1 and ε2 for control factors as per Equation 7:

$$P = \delta \cdot \text{sign}(\delta) + \epsilon_1 \cdot \epsilon_2 + \epsilon_1 \cdot \text{sign}(\delta) \cdot \delta_{\text{weight}} \cdot \epsilon_2 \quad (7)$$

Factoring out ε2 from the last two terms as per Equation 8:

$$p = |\delta| + \epsilon_1\left(1 + \text{sign}(\delta) \cdot \delta_{\text{weight}}\right) \cdot \epsilon_2 \quad (8)$$

Therefore, the equation for MaDi is presented in Equation 9:

$$p = |\delta| + \epsilon_1\left(1 + \text{sign}(\delta) \cdot \delta_{\text{weight}}\right) \cdot \epsilon_2 \quad (9)$$

In Equation 9, δ denotes the TD error [55]. The parameter ϵ_1 regulates its significance, while δ_{weight} controls the importance of the TD error's sign. Furthermore, ϵ_2 manages the balance between exploration and exploitation, allowing the agent to consider less prioritized experiences alongside those with higher priority.

3.3.2QCPR based priority

The QCPR algorithm enhances RL by incorporating quantum circuits into the MaDi priority equation. This integration optimizes experience prioritization by efficiently managing quantum states that encode actions, transitions, and TD errors, thereby significantly improving the efficiency and effectiveness of the RL framework.

Quantum circuit
The QC Q(s) computes the probability distribution over actions, represented as Equation 10[56].

$$Q(s) = |\psi(s)|^2 \quad (10)$$

Here, $|\psi(s)|^2$ is the squared norm of the quantum state ψ(s), indicating the likelihood of selecting each action in state "s" in Equation 10. The quantum circuit computes action probabilities by squaring the norm of the quantum state, a critical operation in quantum computations and algorithms [56, 57]. *Figure 3* is a detailed description of the foundational quantum circuit designed for three qubits and illustrates a quantum circuit for a QCPR agent, featuring state and action registers where Q-values are encoded in superposition amplitudes. Quantum gates, including Hadamard and Controlled-NOT gates, are used to represent actions and transitions,

iteratively updating Q-values and evolving transition states [58, 59].

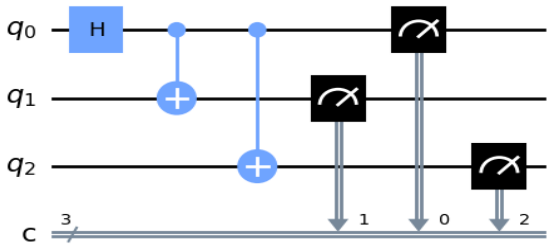

Figure 3 Basic circuit set up without MaDi update

Priority queue

The priority queue assigns priorities to experiences using the squared norm of the TD error, denoted as Equation 11.

$$p(e) = |\delta(e)|^2 \quad (11)$$

This prioritization strategy accentuates experiences with greater priority, thereby enhancing the learning process [60, 61].

Replay memory

The replay memory M stores past experiences (e, priority (e)), enabling the RL agent to sample and learn from prioritized experiences.

$$M = \{(e_1, p(e_1)), (e_2, p(e_2)), \ldots\} \quad (12)$$

Where, M signifies the replay memory, e_i denotes the i^{th} experience, and p (e_i) represents its priority in Equation 12[60].

Action selection

The quantum circuit computes the probability P(a|s) for action selection based on $|Q(s)\_a|^2$, the squared norm of the corresponding amplitude ψ(s) for action as shown in Equation 13.

$$P(a|s) = |Q(s)|^2 \quad (13)$$

In this context, P(a|s) signifies the probability of choosing action "a" in state "s", with $|Q(s)|^2$ representing the squared norm of ψ(s) amplitude for action "a" in the current state [62, 63].

Priority replay

The priority of an experience is updated using the MaDi equation (Equation 9) and is subsequently formulated into Equation 14.

$$Pri(e) = abs(\delta(e)) + \varepsilon 1 (1 + sign(\delta(e)) \delta\_weight)\ \varepsilon 2 + \gamma p(e) \quad (14)$$

This update considers the absolute value of the TD error δ(e) and additional weighting factors ε1, ε2, and δ_{weight}, along with a discount factor γ [64–66].

Update process

The Q-values, Q(s,a), are updated following equation, with circuit reduction described in Equation 15.

$$Q(s, a) = Q(s, a) + \alpha (reward + \gamma\ {}_{max}Q(s', a') - Q(s, a)) \quad (15)$$

Here, α represents the learning rate, reward indicates the received reward, γ denotes the discount factor, s' represents the next state, and a' represents the best action in state 's'.

Figure 4 illustrates a fundamental quantum circuit employed by a QCPR agent prior to incorporating the MaDi update. The circuit utilizes Hadamard and rotation Z gates and includes state and action registers for storing the current environmental state and possible actions. Q-values are encoded in superposition, allowing simultaneous consideration of multiple actions, while quantum gates represent actions and state transitions. This design leverages quantum principles to enable more efficient learning and decision-making compared to traditional approaches.

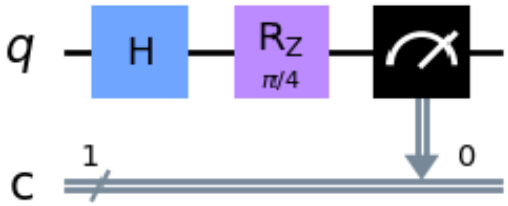

Figure 4 Updated circuit using the MaDi equation

3.4Algorithm design

The QCPR algorithm enhances QL by integrating PER. This algorithm iteratively interacts with the environment, continuously refining quantum circuits to improve efficiency. It presents an innovative framework for RL, addressing existing challenges and advancing the development of more effective learning algorithms.

***Algorithm*: Quantum circuit-based priority replay (QCPR)**

Inputs:

- State s, Action a, Reward R, Next State s'
- Q-values $Q(s, a)$, Learning Rate α, Discount Factor γ
- Error Parameters: $\epsilon_1, \epsilon_2, \delta_{\text{weight}}$

Outputs:

• Updated Q-values $Q(s,a)$

• Prioritized Experience Replay Buffer with Priority $p(e)$

Steps:

1. Temporal Difference (TD) Error:

$$\delta = R + \gamma \cdot \max_a Q(s',a) - Q(s,a)$$

2. Magnitude Calculation:

$$|\delta| = \delta \cdot \text{sign}(\delta)$$

3. Direction(sign) Function:

$$\text{sign}(\delta) = \begin{cases} 1, & \delta > 0 \\ 0, & \delta = 0 \\ -1, & \delta < 0 \end{cases}$$

4. Priority Update :

$$p(e) = |\delta| + \epsilon_1 \cdot \left(1 + \text{sign}(\delta) \cdot \delta_{\text{weight}}\right) \cdot \epsilon_2$$ // **MaDi Priority1**

5. Quantum Circuit for Action Selection:

$$P(a|s) = |Q(s)|^2$$

6. Q-Value Update:

$$Q(s,a) = Q(s,a) + \alpha \cdot \left(\text{reward} + \gamma \cdot \max_a Q(s',a) - Q(s,a)\right)$$

7. Priority Replay Update:

$$\text{Pri}(e) = |\delta(e)| + \epsilon_1 \cdot \left(1 + \text{sign}(\delta(e)) \cdot \delta_{\text{weight}}\right) \cdot \epsilon_2 + \gamma \cdot p(e)$$ //**QCPR Priority2**

3.5Environment setup

This section outlines the simulation and autonomous vehicle environments, including Atari games, Qiskit integration, and Raspberry Pi hardware and software setup (*Table 1*).

The real-world performance analysis of the QCPR algorithm was conducted in a simulated real-world environment, with the environment shown in *Figure 5* by the autonomous vehicle.

Table 1 Environment setup tools

Component	Description
Atari game environment	
Atari Game Environments	Diverse Atari games from OpenAI Gym (Pac-Man, Pong, Space Invaders).
Qiskit Integration	Quantum circuits simulated and optimized with Qiskit.
Raspberry Pi	Headless operation with remote access via secure Shell (SSH) and virtual network computing (VNC) tool.
Autonomous vehicle environment	
Model	Raspberry Pi 4 Model B
Processor	I5, 1.5GHz
RAM	32GB
Storage	1 TB SSD
GPU	VideoCore VI
Power Supply	5V via USB-C
OS	Windows, Raspberry Pi OS
GPS Sensor	UBlox NEO-6M
Programming	Python
Tools	OpenAI Gym, NumPy, Matplotlib, Qiskit, Pennylane, Cirq

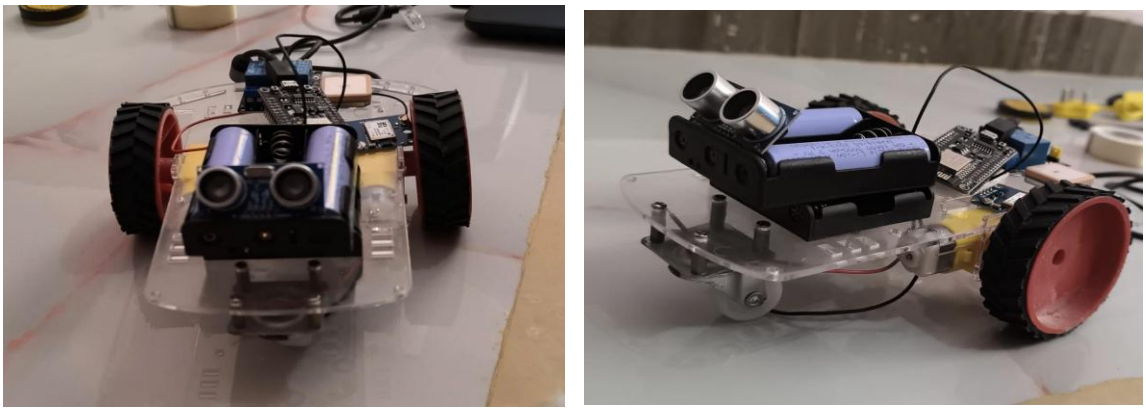

Figure 5 Autonomous vehicle environment with different motions

Figure 6 demonstrates the effectiveness of QCPR's workflow in real-time and Atari games, leveraging Qiskit and Raspberry Pi to enable quantum-enhanced RL.

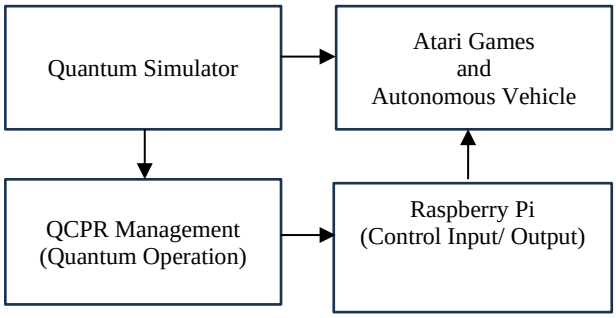

Figure 6 Process work flow of QRL framework

4.Results and discussion

This section evaluates the performance of the QCPR algorithm by analyzing its convergence speed and rewards, focusing on its learning efficiency, decision-making effectiveness, and real-world applicability across various evaluation metrics.

4.1Model evaluation

The evaluation compares the QCPR algorithm with QL and deep Q network (DQN) algorithms by analyzing their loss and accuracy curves. Furthermore, cross-validation is employed to assess robustness, offering insights into each algorithm's performance across various datasets and scenarios.

4.1.1Accuracy and loss

QCPR Accuracy and loss

Figure 7(a) shows how the QCPR algorithm steadily improves both training and validation accuracy. The training accuracy increases from about 0.88 to 0.96, while validation accuracy goes from around 0.87 to 0.95. This upward trend reflects effective learning, better generalization, and overall improved performance, proving the algorithm's ability to make reliable predictions with extended training.

Figure 7(b) highlights the QCPR algorithm's sharp reduction in both training and validation loss over the epochs. The training loss drops from 0.297 to 0.101, and the validation loss decreases from 0.34 to 0.14. These improvements demonstrate the algorithm's strong learning capabilities, better model performance, and enhanced optimization.

QL accuracy and loss

Figure 8(a) illustrates the strong performance of the QL model, with training accuracy peaking at approximately 0.927 and validation accuracy reaching 0.901, reflecting a statistically significant improvement of 0.014. The consistent upward trends in both accuracies emphasize the model's effectiveness in generalizing to unseen data, indicating robust quantum-based RL capabilities.

Figure 8(b) shows that the QL model's training loss decreases from 0.327 to 0.186 by epoch 400. However, the observed fluctuations indicate the possibility of overfitting. Validation loss consistently declines from 0.386 to 0.246, indicating effective learning and generalization. These trends reflect improved data fitting, performance refinement, and adaptability over 500 epochs of training.

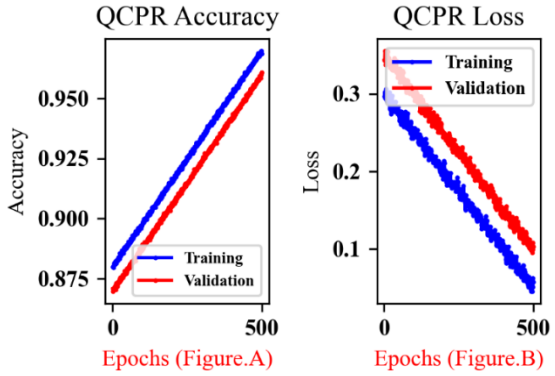

Figure 7 (a) QCPR accuracy and (b) QCPR loss

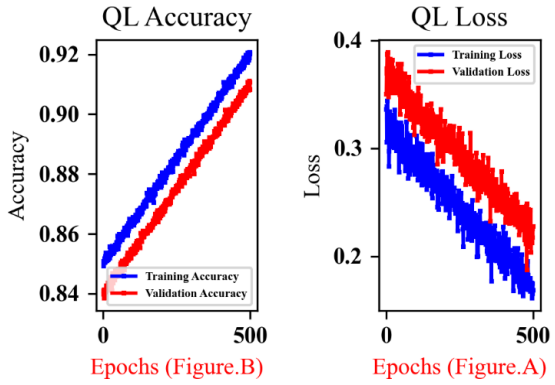

Figure 8 (a) QL accuracy and (b) QL loss

DQN accuracy and loss

Figure 9(a) illustrates a steady increase in training accuracy, peaking at 0.869, which reflects the model's ability to effectively capture data patterns. Initial fluctuations indicate adaptation to dataset complexity. Validation accuracy follows suit, highlighting generalization ability. As both metrics stabilize, they demonstrate the model's increasing

proficiency and reliability in providing accurate predictions.

Figure 9(b) illustrates a downward trend in training loss, stabilizing at approximately 0.26, indicating improved model performance. In contrast, the validation loss fluctuates, peaking at 0.37, suggesting potential overfitting. The Nemenyi test confirms the statistical significance of these differences, highlighting the need for further regularization to enhance generalization.

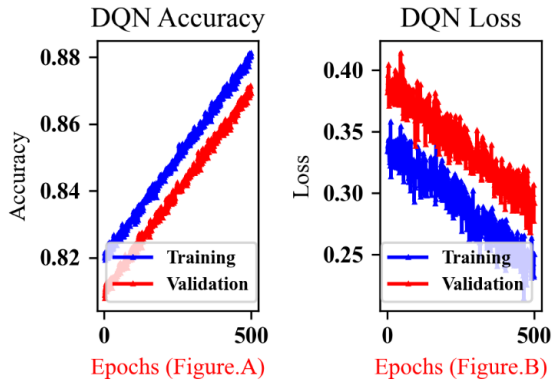

Figure 9 (a) DQN accuracy and (b) DQN loss

4.2K-fold cross-validation

In k-fold cross-validation for QCPR, the dataset is divided into k equal-sized folds (5 folds). Each fold is alternately used for validation while the remaining k−1 folds are used for training. Average metrics, such as accuracy, precision, recall, F1-score, and cumulative reward, ensure a robust evaluation of QCPR's effectiveness.

4.2.1Validation results

The cross-validation results in *Table 2* demonstrate the QCPR algorithm's strong performance, with accuracy between 85% and 89%, precision from 82% to 87%, and recall values of 80% to 88%. Cumulative rewards range from 910 to 970, highlighting its effectiveness in maximizing rewards and generalization.

Table 2 Validation results

Fold	Accuracy (%)	Precision (%)	Recall (%)	F1-score(%)	Cumulative Reward (%)
1	0.85	0.82	0.80	0.81	910
2	0.88	0.86	0.85	0.85	940
3	0.87	0.85	0.86	0.85	970
4	0.86	0.83	0.84	0.83	950
5	0.89	0.87	0.88	0.87	930

Validation metrics calculations for 5 sample data shown from Equation 16 to 20.

Let's calculate the average for each metric.

- Average Accuracy

$$\text{Average Accuracy} = \frac{1}{5}\sum_{i=1}^{5} \text{Accuracy}_i \quad (16)$$

$$\text{Average Accuracy} = \frac{1}{5}(0.85 + 0.88 + 0.87 + 0.86 + 0.89)$$

$$\text{Average Accuracy} = \frac{1}{5} \times 4.35$$

$$\text{Average Accuracy} = 0.87$$

- Average Precision

$$Precisioni\text{Average Precision} = \frac{1}{5}\sum_{i=1}^{5} \text{Precision}_i \quad (17)$$

$$\text{Average Precision} = \frac{1}{5}(0.82 + 0.86 + 0.85 + 0.83 + 0.87)$$

$$\text{Average Precision} = \frac{1}{5} \times 4.23A$$

$$\text{Average Precision} = 0.846$$

- Average Recall

$$\text{Average Recall} = \frac{1}{5}\sum_{i=1}^{5} \text{Recall}_i \quad (18)$$

$$\text{Average Recall} = \frac{1}{5}(0.80 + 0.85 + 0.86 + 0.84 + 0.88)$$

$$\text{Average Recall} = \frac{1}{5} \times 4.23$$

$$\text{Average Recall} = 0.846$$

- Average F1-score

$$\text{Average F1-score} = \frac{1}{5}\sum_{i=1}^{5} \text{F1 Score}_i \quad (19)$$

$$\text{Average F1-score} = \frac{1}{5}(0.81 + 0.85 + 0.85 + 0.83 + 0.87)$$

$$\text{Average F1-score} = \frac{1}{5} \times 4.21$$

$$\text{Average F1} - \text{score} = 0.842$$

- Average cumulative reward

$$\text{Average Cumulative Reward} = \frac{1}{5}\sum_{i=1}^{5} \text{Cumulative Reward}_i \quad (20)$$

$$\text{Average Cumulative Reward} = \frac{1}{5}(910 + 940 + 970 + 950 + 930)$$

$$\text{Average Cumulative Reward} = \frac{1}{5} \times 4700$$

$$\text{Average Cumulative Reward} = 940$$

Based on the sample trials in *Table 3*, the QCPR algorithm achieved an accuracy of 87%, with

precision and recall both at 84.6%, and an F1-score of 84.2%, indicating balanced performance. A reward of 940 indicates its effectiveness, while k-fold cross-validation improves reliability by reducing biases.

Table 3 Summary of average performance

Accuracy	Precision	Recall	F1-score	Reward
0.87	0.846	0.846	0.842	940

4.2.2Receiver operating characteristic (ROC) curve

The QCPR model's performance, evaluated through five-fold cross-validation (*Figure 10*), averaged 87% accuracy, 84.6% precision, and 84.2% F1-score, demonstrating robustness. The ROC curve indicated an area under curve (AUC) of 0.75, reflecting moderate discrimination capability. The results affirm the model's effectiveness and generalization in classification tasks.

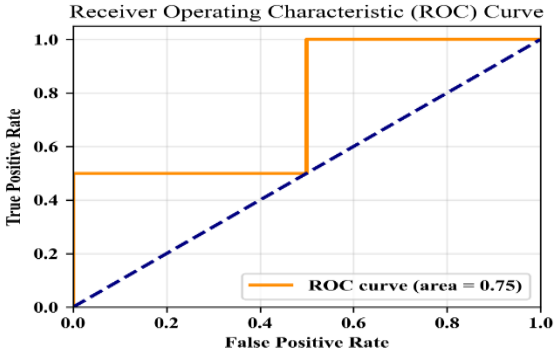

Figure 10 ROC curve for classification

4.2.3Model cross validation comparison: QCPR vs. Baseline algorithms

Table 4 summarizes that QCPR consistently outperforms baseline algorithms, with average improvements of 7% in accuracy, precision, recall, and F1-score, along with significantly higher cumulative rewards across all metrics.

Table 4 Cross validation with different algorithms

Metric	QL	DQN	PER	DDQN	QCPR
Average Accuracy (%)	78.5	82.3	85.1	83.6	90.5
Average Precision (%)	80.2	84.0	86.7	85.2	91.8
Average Recall (%)	77.9	81.5	84.4	82.9	89.9
Average F1-score (%)	79.0	82.7	85.5	84.0	90.8
Average Reward	850	870	890	880	950

The QCPR algorithm demonstrates significant implications for QC and RL. It advances QC by showcasing practical applications of quantum mechanics in optimization, addressing real-world challenges. For RL, QCPR enhances learning with the MaDi priority equation and quantum circuits, enabling improved decision-making in areas like robotics and healthcare.

4.3Hyperparameter tuning

The influence of key hyperparameters—learning rate (α), discount factor (γ), exploration rate (ε), and batch size (M)—on the performance of QL and QCPR was thoroughly examined. *Table 5* shows that QCPR consistently outperformed QL, achieving higher average rewards across trials, highlighting its robustness and superior adaptability in reward maximization.

Table 5 Hyper parameters for QL vs. QCPR

Algorithm	Trails	α	γ	ε	M(Bits)	Episode	Avg. Rewards
QL	1	0.003	0.9	0.2	32	1000	750.34
	2	0.002	0.8	0.2	32	1000	801.23
	3	0.001	0.7	0.2	32	1000	820.34
QCPR	1	0.003	0.9	0.2	32	1000	1024.23
	2	0.002	0.8	0.2	32	1000	1102.45
	3	0.001	0.7	0.2	32	1000	1204.56

4.3.1Sensitivity analysis on hyperparameters

Table 6 summarizes the impact of key hyperparameters on QCPR's performance. The learning rate regulates stability and convergence speed, the discount factor guides reward prioritization, the exploration rate balances exploration vs. exploitation, and batch size affects learning efficiency. Optimal tuning of these parameters is critical for maximizing performance and adaptability across environments.

4.4QCPR and other algorithms with different Atari games

This performance analysis (*Table 7*) highlights QCPR's average improvements over QL and other algorithms in various Atari games. By incorporating

quantum-inspired mechanisms like priority replay, QCPR consistently outperforms traditional models, achieving a 97.5% improvement over QL in Space Invaders and a 96.3% improvement over DQN in Galaga. The consistent superiority demonstrated in *Table 7* against various algorithms highlights the strategic advantage of integrating quantum mechanics into RL. This approach enhances computational efficiency, adaptability, and learning acceleration in complex environments like Atari games. QCPR shows improvements of 12.6% over QL, 13.0% over DQN, 11.3% over asynchronous advantage actor-critic (A3C), and 7.0% over proximal policy optimization (PPO), underscoring its significant enhancements compared to conventional methods.

Table 6 Sensitivity analysis

Hyperparameter	Impact on QCPR performance	Optimal range
Learning Rate (α)	Balances convergence speed and stability	Moderate, empirically tuned
Discount Factor (γ)	Influences long-term vs. short-term reward prioritization	High for sparse rewards
Exploration Rate (ε)	Manages exploration-exploitation balance, annealed over time	High to low (dynamic)
Batch Size (M)	Affects stability vs. computational demands	Larger sizes preferred

Table 7 Atari games with different algorithms

Atari game	QCPR (%)	DQN (%)	A3C (%)	PPO (%)
Pac-Man	89.2	86.5	85.3	87.1
Galaga	97.0	96.3	95.2	91.0
Pong	88.9	86.1	85.0	86.8
Frogger	96.7	96.0	92.9	93.7
Space Invaders	97.5	95.2	94.8	96.0

4.5Conceptual analysis

This section provides a comparative analysis of the QCPR algorithm against various state-of-the-art RL algorithms, highlighting its strengths and weaknesses (*Table 8*). The insights gathered from these comparisons illustrate QCPR's advantages in addressing the limitations present in traditional methods. The QCPR algorithm significantly enhances RL by integrating quantum circuits and prioritized replay, improving learning efficiency, convergence speed, and generalization across diverse environments.

Table 8 Conceptual analysis

Algorithm	Strengths	Limitations	QCPR Advantage
QL	Simple, easy to implement	Slow convergence, overestimation bias	Enhances efficiency with quantum circuits and prioritized replay
DQN	Q-value approximation with deep neural nets	Instability, requires careful tuning	Improves stability and convergence using quantum circuits
PER	Accelerates learning using significant experiences	Relies on suboptimal heuristics	Adaptive prioritization with MaDi priority equation and quantum circuits
DDQN	Effective in sparse reward environments	Struggles in complex environments, needs tuning	Robustness and generalization through quantum-inspired replay
QCPR	Integrates quantum circuits and prioritized replay	Need to test a real-time scenario.	Enhances learning efficiency, convergence speed, and generalization

4.6RL matrices

Four key metrics were evaluated over 200 episodes to assess the RL algorithms: convergence speed, rewards, experience, and priority. Convergence speed measures how quickly the algorithm reaches an optimal solution, reflecting its learning efficiency. Rewards reflect performance based on training outcomes, indicating the effectiveness of decision-making. Experience indicates the algorithm's overall learning capacity, while priority assigns importance to different experiences, enhancing both learning and decision-making. This analysis offers a comprehensive evaluation of the QCPR algorithm's effectiveness and efficiency across various trials in RL.

4.6.1Convergence speed

This section analyses the speed at which a system, algorithm, or process stabilizes or reaches its goal, referred to as 'convergence speed.' By analyzing the provided sample data, insights are gained into how the speed varies across different episodes and its correlation with the frequency of specific events or

cycles. This analysis provides valuable insights into the dynamic learning patterns and adaptive behaviours of the system.

Convergence speed over episodes

Figure 11 (a) provides a graphical representation of these fluctuations in convergence speed across episodes, helping to visualize how the algorithm's behaviour evolves over time. The observed variations reflect the system is dynamic learning patterns and its ability to adapt to different conditions.

Convergence speed frequency over episodes

This section discusses the relationship between convergence speed and frequency, as illustrated in *Figure 11(b).* Convergence speed indicates how quickly a robotic learning algorithm stabilizes and reaches its objective, with higher speeds reflecting greater efficiency. Frequency, on the other hand, refers to how often learning events occur, such as model updates or training iterations. *Figure 11(b)* visually demonstrates this relationship, offering insights into the efficiency and adaptability of the robotic learning algorithm as it optimizes performance over time.

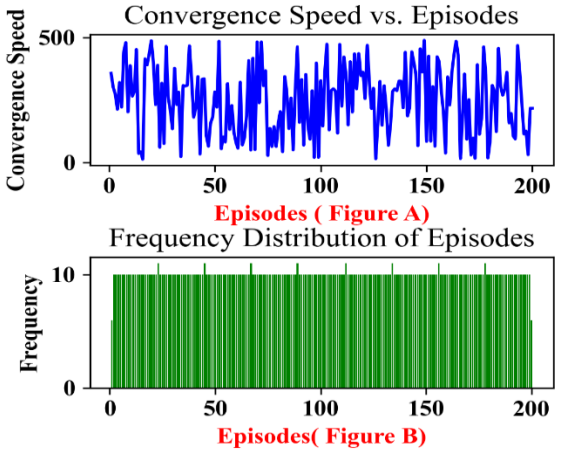

Figure 11 (a) Convergence speed vs. episodes (b) Convergence speed vs. frequency

4.6.2Rewards

The QCPR algorithm outperforms others by more effectively balancing exploration and exploitation during the learning process. It shows greater reward consistency, making it a more efficient approach for RL tasks, especially in complex environments.

Rewards over episodes

The sample data shows the rewards that an agent receives over several episodes while training with an RL algorithm. Let us take a closer look at the first three episodes,

Episode 1: Reward of 501.64

The agent received a high reward, indicating it performed well and likely took effective actions during this episode.

Episode 2: Reward of 481.86

Similarly, the agent maintained a high reward, showing consistent good performance early in its learning process.

Episode 3: Reward of 35.20

The reward dropped significantly in this episode, indicating that the agent may have encountered challenges or made suboptimal decisions, which is typical as it explores various actions and refines its strategies. *Figure 12(a)* illustrates how the agent's rewards fluctuate over time. In early episodes, these variations reflect the agent's extensive exploration of the environment, leading to inconsistent rewards. A noticeable drop in rewards suggests the agent is adjusting its actions, transitioning from exploration to exploiting the best strategies for the task.

Rewards frequency over episodes

The relationship between rewards and episode frequency is depicted in *Figure 12(b),* showing a weak correlation (0.0097) that indicates episode frequency has minimal impact on the rewards received by the agent. This suggests that variations in episode occurrence do not significantly influence reward outcomes. *Figure 12(a)* highlights the dynamic nature of the reward system during early training, while *Figure 12(b)* reveals that reward outcomes are largely independent of episode frequency, emphasizing the complexity of the learning process in RL, where exploration and strategy adjustments interact intricately.

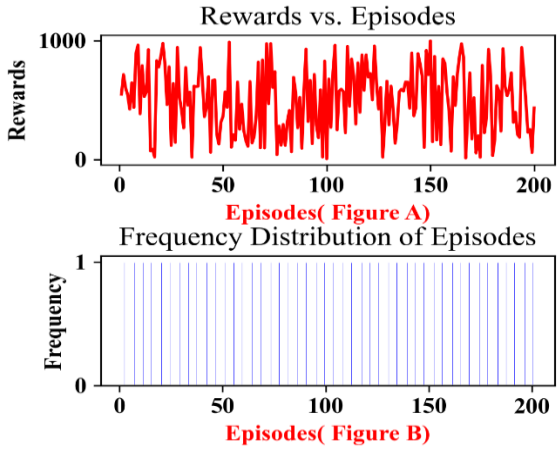

Figure 12 (a) Rewards vs. episodes, (b) Rewards vs. frequency

4.6.3Experience

The QCPR algorithm performs better by leveraging experiences more effectively, resulting in superior performance in RL tasks. The strong correlation between experience data and the number of episodes shows how agents gradually build their skills over time. These observations highlight the effectiveness of their learning and adaptation processes, as more interactions help them gain experience and improve performance.

Experience over episodes

Figure 13(a) illustrates a strong positive correlation of 0.9998 between experience data and the number of episodes, indicating a near-perfect relationship. As episodes increase, the agent's experience grows consistently, reflecting effective learning and skill improvement, ultimately enhancing the agent's performance and adaptability over time.

Experience frequency over episodes

This section examines the relationship between frequency data and the number of episodes (*Figure 13(b)*). There is a strong positive correlation of 0.9998, indicating that as the number of episodes increases, the frequency of experience rises proportionally. This suggests that frequent interactions enhance the agent's learning and skill development. *Figures 13(a)* and *13(b)* highlight the positive relationship between experience and episode frequency, emphasizing the importance of repeated episodes in improving the agent's performance over time.

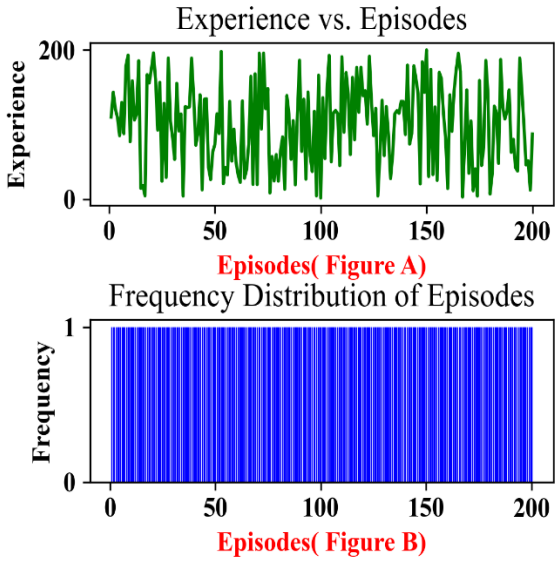

Figure 13(a) Experience vs Episodes, (b) Frequency vs. Episodes

4.6.4Priority

In this section, analyzing the correlation between priority and episodes provides insights into task evolution, facilitating efficient resource allocation. Similarly, prioritizing based on frequency optimizes workflow, thereby enhancing overall efficiency.

Priority over episodes

This section briefs the behaviour of priority changes with the number of episodes (*Figure 14(a)*). Understanding this relationship is important for managing tasks and allocating resources efficiently. Through the analysis of the evolution of priority over episodes, we can make better decisions and ensure that important tasks receive the attention they need throughout the process.

Priority frequency over episodes

This section highlights the relationship between priority and task frequency (*Figure 14(b)*). A positive correlation indicates that higher-priority tasks occur more frequently, which is crucial for optimizing task management and resource allocation. *Figures 14(a)* and *14(b)* illustrate how priority managed across episodes, offering insights into task evolution and enhancing overall efficiency in learning and operational contexts.

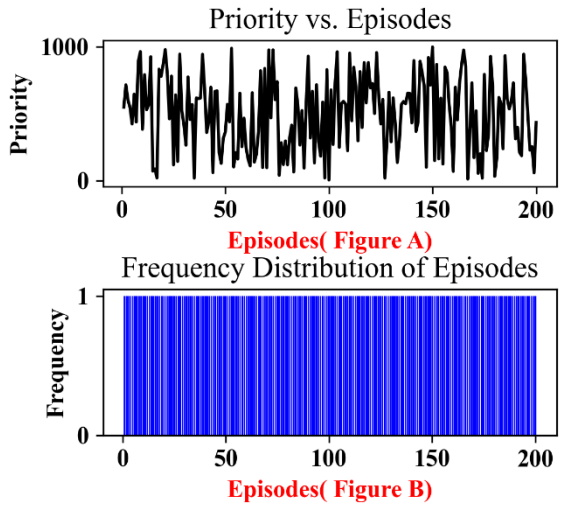

Figure 14 (a)Priority vs Episodes, (b) Priority vs Frequency

4.6.5Limitations

The QCPR algorithm demonstrates impressive performance gains; however, there are several challenges worth noting. First, the quantum-inspired mechanisms, such as MaDi-based prioritization, introduce additional computational complexity,

which can be difficult to manage on resource-constrained hardware like Raspberry Pi. Second, the majority of our evaluations were conducted in simulations and controlled benchmarks, which may not fully reflect the unpredictable nature of real-world scenarios. Third, scaling the algorithm using frameworks like Qiskit could present challenges as the problem size increases. Lastly, energy consumption and resource efficiency, which are critical factors for practical and sustainable applications, were not explored. The full list of abbreviations is provided in *Appendices I and II.*

5.Conclusion and future work

The QCPR algorithm exhibits superior performance relative to classical RL models, such as QL and DQN. Specifically, QCPR has achieved an average classification accuracy of 90.5%, outperforming QL (78.5%) and DQN (82.3%). In terms of precision and recall, QCPR attained 91.8% and 89.9%, respectively, while QL and DQN recorded 86.7% and 84.4%, respectively. Furthermore, QCPR demonstrated an F1-score of 90.8%, highlighting its ability to effectively balance precision and recall, thus outperforming both QL and DQN in this critical metric.

When evaluated for cumulative reward maximization, QCPR demonstrated excellence with an average reward of 950, significantly higher than QL's 750 and DQN's 820. This result highlights QCPR's superior capability in making long-term optimal decisions. Cross-validation experiments conducted across multiple datasets reinforced the robustness of QCPR, with an average accuracy of 87% and an F1-score of 84.2%. Sensitivity analysis revealed that QCPR's performance was most sensitive to hyperparameters such as the learning rate and exploration rate, with a notable 4% improvement in accuracy when these parameters were optimized.

In real-world applications, QCPR can significantly enhance decision-making processes, particularly in the domain of autonomous vehicles. By optimizing navigation strategies, QCPR can effectively balance exploration and exploitation, ensuring safety, efficiency, and adaptability in dynamic environments.

The future work directions are as follows:

1. Investigating the performance of QCPR in domains such as healthcare, autonomous vehicles, and supply chain management to assess its flexibility and real-world applicability.
2. Optimizing the algorithm to reduce energy consumption, making it suitable for resource-constrained devices and promoting environmentally friendly applications.
3. Implementing QCPR on actual quantum machines to evaluate its practical potential and uncover areas for hardware-specific improvements.
4. Enhancing QCPR to handle noisy, unpredictable, and rapidly changing environments, making it more robust and effective for real-world challenges.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

Data availability

The data used in this article is accessible at the following URL:
https://github.com/PALANIVELAADHI/DATASET_FOR_IJATEE.git

Author's contribution statement

R. Palanivel: Conceptualization, investigation, writing – original draft, writing – review and editing. **P. Muthulakshmi:** Writing – original draft, review and editing, analysis, and interpretation of result.

References

[1] Nakabi TA, Toivanen P. Deep reinforcement learning for energy management in a microgrid with flexible demand. Sustainable Energy, Grids and Networks. 2021; 25:100413.

[2] Dulac-arnold G, Levine N, Mankowitz DJ, Li J, Paduraru C, Gowal S, et al. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. Machine Learning. 2021; 110(9):2419-68.

[3] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S. Quantum machine learning. Nature. 2017; 549(7671):195-202.

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

[5] Nielsen MA, Chuang IL. Quantum computation and quantum information. Cambridge University Press; 2010.

[6] Padakandla S. A survey of reinforcement learning algorithms for dynamically varying environments. ACM Computing Surveys. 2021; 54(6):1-25.

[7] Xiao H, Chen X, Xu J. Using a deep quantum neural network to enhance the fidelity of quantum convolutional codes. Applied Sciences. 2022; 12(11):1-11.

[8] Saggio V, Asenbeck BE, Hamann A, Strömberg T, Schiansky P, Dunjko V, et al. Experimental quantum speed-up in reinforcement learning agents. Nature. 2021; 591(7849):229-33.

[9] Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, et al. Human-level control through deep reinforcement learning. Nature. 2015; 518(7540):529-33.
[10] Preskill J. Quantum computing in the NISQ era and beyond. Quantum. 2018; 2:79.
[11] Schuld M, Petruccione F, Schuld M, Petruccione F. Quantum models as kernel methods. Machine Learning with Quantum Computers. 2021:217-45.
[12] Wei Q, Ma H, Chen C, Dong D. Deep reinforcement learning with quantum-inspired experience replay. IEEE Transactions on Cybernetics. 2021; 52(9):9326-38.
[13] Gonçalves CP. Quantum robotics, neural networks and the quantum force interpretation. Neuro Quantology. 2019; 17(2):33-55.
[14] Jiang H, Gui R, Chen Z, Wu L, Dang J, Zhou J. An improved sarsa ($\lambda $) reinforcement learning algorithm for wireless communication systems. IEEE Access. 2019; 7:115418-27.
[15] Baritompa WP, Bulger DW, Wood GR. Grover's quantum algorithm applied to global optimization. SIAM Journal on Optimization. 2005; 15(4):1170-84.
[16] Andrés E, Cuéllar MP, Navarro G. On the use of quantum reinforcement learning in energy-efficiency scenarios. Energies. 2022; 15(16):1-24.
[17] Fährmann D, Jorek N, Damer N, Kirchbuchner F, Kuijper A. Double deep q-learning with prioritized experience replay for anomaly detection in smart environments. IEEE Access. 2022; 10:60836-48.
[18] Miyajima H, Shigei N, Makino S, Miyajima H, Miyanishi Y, Kitagami S, et al. A proposal of privacy preserving reinforcement learning for secure multiparty computation. Artificial Intelligence Research. 2017; 6(2):57-68.
[19] Lin LJ. Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine Learning. 1992; 8:293-321.
[20] Schaul T, Quan J, Antonoglou I, Silver D. Prioritized experience replay. 4th International Conference on Learning Representations 2016(pp. 1-21). ICLR.
[21] Sannia A, Giordano A, Gullo NL, Mastroianni C, Plastina F. A hybrid classical-quantum approach to speed-up Q-learning. Scientific Reports. 2023; 13(1):1-10.
[22] Skolik A, Mangini S, Bäck T, Macchiavello C, Dunjko V. Robustness of quantum reinforcement learning under hardware errors. EPJ Quantum Technology. 2023; 10(1):1-43.
[23] Chen SY. Asynchronous training of quantum reinforcement learning. Procedia Computer Science. 2023; 222:321-30.
[24] Li Z, Zhou Y, Liu Y, Zhu F, Yang C, Hu S. QAP: a quantum-inspired adaptive-priority-learning model for multimodal emotion recognition. In findings of the association for computational linguistics: ACL 2023 (pp. 12191-204). Association for Computational Linguistics.
[25] Tian J, Sun X, Du Y, Zhao S, Liu Q, Zhang K, et al. Recent advances for quantum neural networks in generative learning. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023; 45(10):12321-40.
[26] Chen SY. Quantum deep Q-learning with distributed prioritized experience replay. In international conference on quantum computing and engineering (QCE) 2023 (pp. 31-5). IEEE.
[27] Dong D, Chen C, Li H, Tarn TJ. Quantum reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2008; 38(5):1207-20.
[28] Lamata L. Quantum machine learning implementations: proposals and experiments. Advanced Quantum Technologies. 2023; 6(7):1-7.
[29] Metz F, Bukov M. Self-correcting quantum many-body control using reinforcement learning with tensor networks. Nature Machine Intelligence. 2023; 5(7):780-91.
[30] Chen SY. Quantum deep recurrent reinforcement learning. In international conference on acoustics, speech and signal processing (ICASSP) 2023 (pp. 1-5). IEEE.
[31] Awad M, Fraihat S. Recursive feature elimination with cross-validation with decision tree: feature selection method for machine learning-based intrusion detection systems. Journal of Sensor and Actuator Networks. 2023; 12(5):1-23.
[32] Sharma NK, Kumar S, Yadav PK. Enhancing infrastructure sustainability: reliability and sensitivity analysis of localized integrated renewable energy systems using feed forward backpropagation neural network. International Journal of Advanced Technology and Engineering Exploration. 2024; 11(110):58-75.
[33] Ghintab SS, Hassan MY. Localization for self-driving vehicles based on deep learning networks and RGB cameras. International Journal of Advanced Technology and Engineering Exploration. 2023; 10(105):1016-36.
[34] Li H, Qian X, Song W. Prioritized experience replay based on dynamics priority. Scientific Reports. 2024; 14(1):1-9.
[35] Wang H, Xiang H. Quantum optimization algorithm based on multistep quantum computation. New Journal of Physics. 2024; 26(7):1-16.
[36] Roik J, Bartkiewicz K, Černoch A, Lemr K. Routing in quantum communication networks using reinforcement machine learning. Quantum Information Processing. 2024; 23(3):89.
[37] Vandelli M, Lignarolo A, Cavazzoni C, Dragoni D. Evaluating the practicality of quantum optimization algorithms for prototypical industrial applications. Quantum Information Processing. 2024; 23(10):1-14.
[38] Patil S, Banerjee S, Panigrahi PK. NISQ-friendly measurement-based quantum clustering algorithms. Quantum Information Processing. 2024; 23(10):341.
[39] Zhao LY, Chang TQ, Zhang L, Zhang J, Chu KX, Kong DP. Targeted multi-agent communication algorithm based on state control. Defence Technology. 2022; 31: 544-56.

[40] Palanivel R, Muthulakshmi P. Error mitigation using quantum neural Q network in secure qutrit distribution on cleve's protocol on quantum computing. Quantum Information Processing. 2024; 23(4):1-30.
[41] Mangla C, Rani S, Abdelsalam A. QLSN: quantum key distribution for large scale networks. Information and Software Technology. 2024; 165:107349.
[42] Kundu A, Bedełek P, Ostaszewski M, Danaci O, Patel YJ, Dunjko V, et al. Enhancing variational quantum state diagonalization using reinforcement learning techniques. New Journal of Physics. 2024; 26(1):1-18.
[43] Yun WJ, Kim JP, Jung S, Kim JH, Kim J. Quantum multiagent actor–critic neural networks for internet-connected multirobot coordination in smart factory management. IEEE Internet of Things Journal. 2023; 10(11):9942-52.
[44] Peral-garcía D, Cruz-benito J, García-peñalvo FJ. Systematic literature review: quantum machine learning and its applications. Computer Science Review. 2024; 51:1-20.
[45] Hellstern G, Dehn V, Zaefferer M. Quantum computer based feature selection in machine learning. IET Quantum Communication. 2024; 5(3):232-52.
[46] Palanivel R, Muthulakshmi P. Design and analysis of quantum transfer fractal priority replay and mirdad priority loss algorithms for quantum reinforcement learning. In international conference on data management, analytics & innovation 2024 (pp. 409-24). Singapore: Springer Nature Singapore.
[47] Palanivel R, Muthulakshmi P. Design and analysis of parallel quantum transfer fractal priority replay with dynamic memory algorithm in quantum reinforcement learning for robotics. IET Quantum Communication. 2024:1-24.
[48] Van SH, Van HH, Whiteson S, Wiering M. A theoretical and empirical analysis of expected Sarsa. In symposium on adaptive dynamic programming and reinforcement learning 2009 (pp. 177-84). IEEE.
[49] Dong D, Chen C, Li H, Tarn TJ. Quantum reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2008; 38(5):1207-20.
[50] Barnard E. Temporal-difference methods and Markov models. IEEE Transactions on Systems, Man, and Cybernetics. 1993; 23(2):357-65.
[51] Yao J, Bukov M, Lin L. Policy gradient based quantum approximate optimization algorithm. In mathematical and scientific machine learning 2020 (pp. 605-34). PMLR.
[52] Semola R, Moro L, Bacciu D, Prati E. Deep reinforcement learning quantum control on IBMQ platforms and Qiskit pulse. In international conference on quantum computing and engineering (QCE) 2022 (pp. 759-62). IEEE.
[53] Dalla PN, Buffoni L, Martina S, Caruso F. Quantum reinforcement learning: the maze problem. Quantum Machine Intelligence. 2022; 4(1):1-10.
[54] Schuld M, Killoran N. Quantum machine learning in feature Hilbert spaces. Physical Review Letters. 2019; 122(4):040504.
[55] Mitarai K, Negoro M, Kitagawa M, Fujii K. Quantum circuit learning. Physical Review A. 2018; 98(3):1-6.
[56] Chintala P, Dornberger R, Hanne T. Robotic path planning by Q learning and a performance comparison with classical path finding algorithms. International Journal of Mechanical Engineering and Robotics Research. 2022; 11(6):373-8.
[57] Zhou X, Feng Y, Li S. A monte carlo tree search framework for quantum circuit transformation. In proceedings of the 39th international conference on computer-aided design 2020 (pp. 1-7). ACM.
[58] Zhao C, Gao XS. QDNN: deep neural networks with quantum layers. Quantum Machine Intelligence. 2021; 3(1):1-9.
[59] Payares ED, Martínez-santos JC. Parallel quantum computation approach for quantum deep learning and classical-quantum models. In journal of physics: conference series 2021 (pp. 1-11). IOP Publishing.
[60] Hu W, Hu J. Q learning with quantum neural networks. Natural Science. 2019; 11(1):31-9.
[61] Fang P, Zhang C, Situ H. Quantum state clustering algorithm based on variational quantum circuit. Quantum Information Processing. 2024; 23(4):125.
[62] Uehara GS, Spanias A, Clark W. Quantum information processing algorithms with emphasis on machine learning. In 12th international conference on information, intelligence, systems & applications (IISA) 2021 (pp. 1-11). IEEE.
[63] Yan R, Wang Y, Xu Y, Dai J. A multiagent quantum deep reinforcement learning method for distributed frequency control of islanded microgrids. IEEE Transactions on Control of Network Systems. 2022; 9(4):1622-32.
[64] Cleve R. An introduction to quantum complexity theory. Collected Papers on Quantum Computation and Quantum Information Theory. 2000:103-27.
[65] Peng X, Chen R, Zhang J, Chen B, Tseng HW, Wu TL, et al. Enhanced autonomous navigation of robots by deep reinforcement learning algorithm with multistep method. Sensors & Materials. 2021; 33(2):825-42.
[66] Dong D, Chen C, Chu J, Tarn TJ. Robust quantum-inspired reinforcement learning for robot navigation. IEEE/ASME Transactions on Mechatronics. 2010; 17(1):86-97.

R. Palanivel is a researcher, industry professional, and educator from Tamil Nadu, India. He is currently conducting advanced research on Quantum Computing, AI, Robotics, and Edge Computing at SRM Institute of Science and Technology. His work focuses on applying quantum computing to machine learning, optimization, cryptography, medical image processing, and generative AI. Alongside his research, he is dedicated to advancing education in these fields, fostering innovative solutions, and pushing the boundaries of technology and its applications.
Email: jta.palanivel@gmail.com

Dr. P. Muthulakshmi is a Professor and Head of the Department of computer science at SRM Institute of Science and Technology, from Tamil Nadu, India. Her research interests encompass Smart Systems, Parallel and Distributed Computing, Quantum Computing, Edge Computing, and Algorithms. She is dedicated to advancing the field through innovative research and teaching, contributing to the development of efficient computing solutions. With a strong focus on the intersection of technology and applications, her aims to foster collaboration and promote advancements in smart system design and computational methodologies.
Email: muthulap@srmist.edu.in

Appendix I

S. No.	Abbreviation	Description
1	A3C	Asynchronous Advantage Actor-Critic
2	AI	Artificial Intelligence
3	AUC	Area Under Curve
4	DQN	Deep Q Network
5	FOSS	Free and Open Source Software
6	GPU	Graphics Processing Unit
7	MaDi	Magnitude and Direction
8	NISQ	Noisy Intermediate-Scale Quantum
9	PER	Prioritized Experience Replay
10	PPO	Proximal Policy Optimization
11	QAP	Quantum Attention Pooling
12	QC	Quantum Computing
13	QCPR	Quantum Circuit-Based Priority Replay
14	QKD	Quantum Key Distribution
15	QL	Q-Learning
16	QML	Quantum Machine Learning
17	QNQN	Quantum neural Q Network
18	QRL	Quantum Reinforcement Learning
19	RL	Reinforcement Learning
20	ROC	Receiver Operating Characteristic
21	SSH	Secure Shell
22	TD	Temporal Difference
23	VQA	Variational Quantum Algorithms
24	VNC	Virtual Network Computing

Appendix II Quantum computing special characters

$Q(s)$	Quantum circuit for state (s) that calculates action probabilities
$\|\psi(s)\|^2$	Squared norm of quantum state ($\psi(s)$), indicating action probabilities
(δ)	Temporal Difference (TD) error
(ϵ_1) (ϵ_2)	Controls the impact of TD error magnitude.
(δ_{weight})	Adjusts the importance of TD error's sign
(p)	Priority of an experience, calculated using the MaDi method
(M)	Replay Memory Stores past experiences for sampling during learning and their priorities
(e_i)	Experience in the replay memory
$p(e_i)$	Priority of experience
(e_i)	The replay memory.
$P(a\|s)$	Probability of action (a) in state (s) from the quantum circuit
$Q(s,a)$	Q-value for state (s) and action (a), updated during learning
(α)	Learning rate
(r)	Reward received after taking action (a) in state (s).
(γ)	Discount factor for future rewards
(s')	Next state after action (a).
(a')	Best action in the next state (s').