Q-Learning Performance on the CartPole Environment Under Observation Noise and Reward Variants

Steven Ren
Department of Computer Science
University of Massachusetts Amherst
Amherst, Massachusetts, USA
email: skren@umass.edu

Taieba Tasnim
Department of Computer Science
Tuskegee University
Tuskegee, Alabama, USA
email: ttasnim6386@tuskegee.edu

Berkeley Wu
Auburn City School
Auburn, Alabama, USA
email: tulipfan002@hotmail.com

Mohammad Rahman
Department of Computer Science
Tuskegee University
Tuskegee, Alabama, USA
email: mrahman@tuskegee.edu

Fan Wu
Department of Computer Science
Tuskegee University
Tuskegee, Alabama, USA
email: fwu@tuskegee.edu

***Abstract*—This paper evaluates Q-learning performance in the CartPole reinforcement learning environment under varying levels of observation noise and two distinct reward functions, in the broader context of designing robust learning-based controllers for cyber-physical systems. Specifically, we compare the standard step-based reward with a cosine-based reward designed to encourage upright pole balance. Observation noise is modeled as Gaussian noise, with standard deviations scaled to the range of each observation variable. Through multiple training runs at different noise levels, we evaluated convergence behavior, pole angle stability, and cumulative rewards. Our results show that observation noise significantly impairs learning under standard reward, whereas cosine-based reward improves robustness and promotes more stable policies. By linking reinforcement learning with noise-robust control design, this work directly contributes to the understanding of Q-learning under noisy environments and represents a step toward applying reinforcement learning to real-world cyber-physical systems, where noise and variability are inherent.**

***Keywords-reinforcement learning; q-learning; noise; reward; cyber-physical systems*.**

I. Introduction

The CartPole system is a classic reinforcement learning environment that is commonly used to benchmark various algorithms. In this research, the open-source Python library Gymnasium, developed by the Farama Foundation, is used to implement the CartPole environment [1]. The CartPole system—also known as the inverted pendulum—is a fundamental control problem used to test reinforcement learning algorithms. While much prior work has demonstrated successful applications of reinforcement learning algorithms to CartPole [2], real-world factors such as sensor noise and the design of reward functions have been less explored. This paper studies the impact of additive observation noise and shaped reward functions on Q-learning convergence and policy behavior.

The main contributions of this work are:

- Application of a Q-learning algorithm to CartPole under noisy observation inputs.
- Comparison of a standard reward function with a cosine-based reward function shaped by the pole angle.
- Evaluation of convergence episodes, pole angle statistics, and performance variance across noise levels.

The remainder of the paper is organized as follows. Section II provides a background on Q-learning, the epsilon-greedy policy, and CartPole. Section III outlines the methodology, including observation boundaries, noise modeling, reward functions, and training steps. Section IV presents and discusses the experimental results. Section V concludes the research with a summary and directions for future work.

II. Literature review

Q-learning, a concept first introduced by Watkins more than three decades ago, values delayed rewards in reinforcement learning [3]. It operates by estimating the optimal action-value function and aims for long-term reward maximization without requiring a model of the environment. The epsilon-greedy algorithm is a simple and commonly used method in reinforcement learning that attempts to balance exploration and exploitation [4]. In recent years, reinforcement learning has found increasing application in control problems, particularly in robotics and other cyber-physical systems where adaptive behavior is essential [5]. Q-learning, due to its simplicity and ability to handle discrete actions, has been successfully applied in robotic navigation and control [6]. Additional advancements have been made on top of the original Q-learning function, such as Efficient Q-learning, which improves computation through newly defined state and action spaces, a new reward function, and an optimized selection strategy [7]. The Deep Q-learning algorithm also extends from Q-learning by using a deep neural network to approximate the action-value function. With certain modifications, it has been applied to efficiently solve two-player zero-sum Markov games [8], in addition, it performs with good stability and optimality [9].

Above are previous studies of Q-learning and their various applications. More recently, researchers have also turned their attention to the robustness of reinforcement learning methods in noisy and uncertain environments, particularly in cyber-physical

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

systems where safety is critical. A study by Krish et al. on observation noise robustness utilizes a tree-based algorithm for neural network control systems to identify the smallest amount of observation noise that can cause the neural network-based controller to violate safety constraints. They apply the algorithm on several systems such as Gymnasium's CartPole and LunarLander, along with two aircraft systems [10]. In another study, Nazrul demonstrates how reinforcement learning can be applied to optimize sampling frequency in cloud-based cyber-physical systems, enabling dynamic adjustment based on network conditions and system state. In a vehicle cruise control case, this approach outperformed fixed sampling strategies by balancing control performance with network efficiency [11].

In context of exploring how Q-learning performs under noisy environments, this paper will also briefly introduce another popular reinforcement learning algorithm, SARSA (State-Action-Reward-State-Action), which will serve as a baseline for comparison. The key difference between the two algorithms is that while Q-learning learns the value of the optimal policy, SARSA learns the value of the current policy being followed [12][13]. Details of the key terms mentioned here are explained in the following background section.

Summary of Notation

θ Pole angle in radians.
$\overline{\theta}$ Mean of the pole angle over an episode.
$\mathrm{Var}(\theta)$ Variance of the pole angle over an episode.
r Reward given to the agent.
γ Discount factor applied to future rewards.
ϵ Probability of taking a random action in ϵ-greedy policy.
$Q(s, a)$ Estimated value of taking action a in state s.
α Learning rate for Q-value updates.
n Number of steps within an episode.
$\tilde{o}_i$ Noisy observation values for observation i.
o_i True observation values for observation i.
σ_i^2 Variance of the Gaussian noise added to o_i.

III. System and Problem Formulation

The CartPole system is a simulation used to solve the cart-pole problem, described as: "A pole is attached by an unactuated joint to a cart, which moves along a frictionless track. The pendulum is placed upright on the cart and the objective is to balance the pole by applying forces in the left and right directions on the cart" [2]. In Gymnasium's implementation, the agent is rewarded for each step taken while the pole remains upright. The environment terminates when the pole falls beyond a threshold or the cart moves out of bounds. An episode is defined as a sequence of actions that begins with a reset and ends with termination—either by failure or upon reaching the maximum of 500 steps. The maximum achievable reward per episode is 500, which serves as the convergence threshold.

The goal of this work is to train a reinforcement learning model using Q-learning to solve the CartPole system of balancing a pole in the presence of observation noise and then analyze the impact of noise and reward choice on performance. The system receives continuous observation values for cart position, cart velocity, pole angle, and pole angular velocity, which are subject to additive Gaussian noise to simulate real-world inaccuracies. These noisy observations are then discretized to define a finite set of states. At each step, an action is selected to maximize the cumulative reward for an episode. Two reward functions are used along with different levels of noise, and the convergence behavior and pole stability are assessed to understand the impact of noise and reward on the learning process. Figure 1 is a block diagram showing the overall Q-learning CartPole system with noise.

Q-learning is a model-free reinforcement learning algorithm that learns action-value functions based on observed transitions [3]. The Q-function describes the Q-table, which holds all action-value pairs and their corresponding Q-values (a 1×2 array where index 0 represents the reward for the action "left" and index 1 represents the reward for the action "right"):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)\Big] \quad (1)$$

Here, $Q(s_t, a_t)$ denotes the current estimate of the action-value function, the expected return of taking action a_t in state s_t at time step t. The parameter α is the learning rate, r_{t+1} is the reward received after taking action a_t, and γ is the discount factor. The term $\max_{a'} Q(s_{t+1}, a')$ denotes the maximum predicted future reward obtainable from the next state s_{t+1} over all possible actions a'.

The Q-learning update rule can be interpreted as follows: take the current Q-value for this state-action pair and update it using the immediate reward just received, plus the best Q-value expected from the next state [11]. The learning rate α determines how strongly this new estimate influences the update, while the discount factor γ controls the importance given to future rewards.

To balance the trade-off between exploration (trying new or less-used actions) and exploitation (choosing the best-known action), we apply the epsilon-greedy policy, which helps choose the action based on current state [14], defined as:

$$a(s) = \begin{cases} \text{random action}, & \text{if } \varepsilon > \text{rand()} \\ \arg\max_a Q(s, a), & \text{otherwise} \end{cases} \quad (2)$$

Here, ε is the epsilon value, a probability between 0 and 1 that determines the chance of choosing a random action, and it gradually decays over time toward a small constant. The function rand() represents a randomly sampled float from a uniform distribution over the interval $[0, 1]$. The expression $\arg\max_a Q(s, a)$ denotes the action that currently has the highest Q-value for the state s. This exploration policy ensures sufficient exploration during early training episodes, while gradually favoring the exploitation of the learned Q-values as training progresses [15].

IV. Methodology

This section outlines the approach used to evaluate Q-learning performance, including the environment setup, state and action representations, and implementation details of the learning process.

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

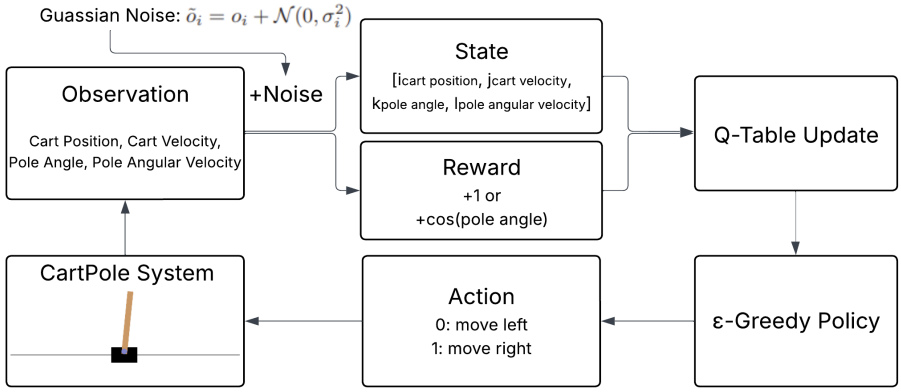

Figure 1. Block diagram of Q-learning CartPole system with noise.

A. Environment and Observations

The CartPole environment is implemented using the Gymnasium library developed by the Farama Foundation [1]. The system simulates a cart moving along a one-dimensional, frictionless track with a pole attached to it via an unactuated hinge joint. The agent receives observations in the form of a four-dimensional state vector: cart position x, cart velocity v, pole angle θ, and pole angular velocity ω.

Variables are continuous and bounded within defined limits:

TABLE I. OBSERVATION SPACE RANGES OF CARTPOLE ENVIRONMENT

Observation	Range
Cart Position (x)	$[-4.8, 4.8]$
Cart Velocity (v)	$[-5.0, 5.0]$
Pole Angle (θ)	$[-0.418, 0.418]$ radians
Pole Angular Velocity (ω)	$[-10.0, 10.0]$

Note that the observation space here differs from Gymnasium's original infinite range for Cart Velocity and Pole Angular Velocity. A limitation of this environment is its discrete action space, restricted to two binary actions: 0 for moving left and 1 for moving right. The velocity affected by the force applied to the cart is not fixed and depends on the pole's angle. We cannot directly specify a particular amount of force as an action [16].

B. Observation Noise

To simulate imperfect sensor measurements encountered in real-world systems, additive Gaussian noise is applied to each component of the observation vector:

$$\tilde{o}_i = o_i + \mathcal{N}(0, \sigma_i^2) \tag{3}$$

Here, o_i represents the true observation, and σ_i is the standard deviation of the noise applied to observation i, proportional to the variable's range. This noise is injected before state discretization, meaning it may cause the agent to misclassify its current state. Several noise levels are tested—specifically, 0.0 (no noise), 0.01, 0.05, and 0.1—to evaluate their effect on learning performance and control stability.

C. State Discretization

Since Q-learning operates on a discrete state space, each continuous observation variable is divided into a fixed number of bins. These bins are uniformly spaced within each variable's range. The state is encoded as a tuple of discretized indices corresponding to the binned values of cart position, cart velocity, pole angle, and pole angular velocity. The combination of these indices uniquely identifies a state in the Q-table. In this paper, we use 8 bins for cart position, 8 bins for cart velocity, 20 bins for pole angle, and 20 bins for pole angular velocity. Note that a larger number of bins sharply increases computational complexity [17]. This discretization reduces the infinite continuous observation space to a manageable number of discrete states, at the cost of precision. Observation noise can cause transitions between neighboring bins, introducing non-determinism into state transitions.

D. Reward Functions

In this work, two reward functions are evaluated:

1) Default Reward: A constant reward of +1 is given at each step as long as the pole remains upright and the cart stays within bounds. This is the default reward under the gymnasium environment.

2) Cosine-Based Reward: The reward is defined as:

$$r = \cos(\theta) \tag{4}$$

This function rewards the agent more when the pole angle θ is near vertical ($\theta = 0$) and penalizes deviations from the position. Since $\cos(0) = 1$, this function shapes the agent's behavior toward learning actions that minimize pole deviation, instead of just surviving.

E. Training Details

All reward function and noise level combinations are trained over 10,000 episodes, with each episode capped at 500 steps. The Q-learning hyperparameters used are:

- Learning rate (α) = 0.1
- Discount factor (γ) = 0.95
- Epsilon (ε) starts at 1.0 and decays exponentially to a minimum of 0.001

These parameters were found to perform well in the local environment: a Windows laptop with modern CPU and GPU, though they can be adjusted based on performance goals.

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

For each episode, the following statistics are recorded:

- Total reward: Sum of rewards per episode
- Pole angle mean and variance: Metrics that show how well Q-learning stabilizes the pole

Pole angle mean formula:

$$\overline{\theta} = \frac{1}{n}\sum_{i=1}^{n}\theta_i \tag{5}$$

Pole angle variance formula:

$$\mathrm{Var}(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta_i - \overline{\theta})^2 \tag{6}$$

Here, n is the number of steps in the episode, and θ_i is the pole angle at step i.

F. SARSA Baseline

A baseline comparison using the SARSA algorithm is run under the same settings as the Q-learning CartPole system, with the function being:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big] \tag{7}$$

At each time step, the agent updates its action-value estimate $Q(s_t, a_t)$ based on the actual reward received, the next observed state s_{t+1}, and the next action a_{t+1} selected by the current policy. The SARSA update is policy-dependent, as the learned values directly reflect the behavior policy being followed, including any exploration strategy. The same data as Q-learning is collected, allowing for a direct baseline comparison. This enables an assessment of how the off-policy approach of Q-learning influences learning performance relative to the on-policy nature of SARSA under noisy observations.

V. Results and Discussion

The bar graph (Figure 2) shows the number of episodes required to reach convergence—defined here as achieving a total reward of 500—in the CartPole environment across different combinations of reward functions and observation noise levels. A maximum of 10,000 episodes was allowed, with bars reaching that value indicating non-convergence within the limit. We can observe that, for a default reward, only the training with no noise successfully converges within the 10,000 episode limit. However, the cosine reward function, which penalizes larger pole angles, shows the ability to converge at noise levels up to 0.01. This suggests that the cosine reward function can offer an improved Q-learning experience and encourage more stable control behavior, allowing for the CartPole system to stay upright.

The two box plots (Figure 3 and Figure 4) show the mean and variance of pole angles across episodes for different combinations of noise levels and rewards. For example, in the mean pole angle plot for cosine reward, a single dot represents the mean pole angle over all steps taken within one episode.

For the default reward function, the mean pole angle remains close to zero when there is no noise, indicating that the pole stays centered. However, as the noise level increases to 0.01 and beyond, the mean pole angle shifts and becomes more

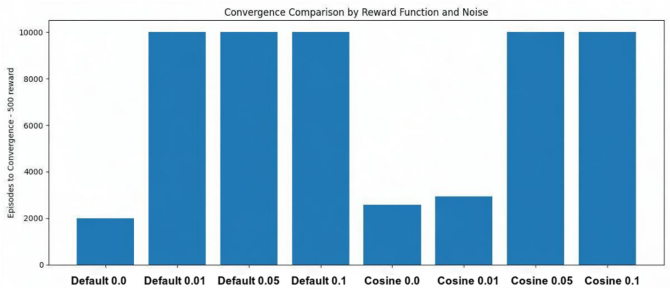

Figure 2. Q-Learning bar plot comparing episodes to convergence of different reward function and noise level combinations.

spread out, which is an expected behavior. This suggests that the training struggles to maintain balance under noisy conditions. The corresponding variance plots further reinforce this observation, showing a notable increase in pole angle variance with rising noise levels. Specifically, the median variance increases and the spread widens, indicating more frequent and extreme pole oscillations during training.

In contrast, the cosine reward function exhibits much better performance. Less outliers are observed at 0.0 noise level demonstrating the cosine reward's ability to promote steadier control even in uncertain environments.

As for the SARSA algorithm, the bar graph (Figure 5) shows the number of episodes required to reach convergence and the two box plots (Figure 6 and Figure 7) show the mean and variance of poles angles across episodes for different combinations of noise levels and rewards just like the Q-learning figures. The SARSA bar graph can be seen to have a similar points of convergences as the Q-learning bar graph. Also, similar to Q-learning, it can be seen that under cosine reward, the variance is more consistent across noise levels.

Overall, these plots show that for Q-learning the default reward function leads to unstable learning in the presence of noise, while the cosine reward function encourages more stable and consistent control. This aligns well with the convergence analysis, where the cosine reward enabled convergence at the 0.01 noise level, in contrast to the lack of convergence when noise was added under the default reward training. The CartPole system can be seen to behave similarly under the SARSA algorithm. These results demonstrate that careful reward design—such as using a cosine-based function that penalizes large pole angles—can improve robustness in reinforcement learning for the CartPole environment.

VI. Conclusion and Future Work

This paper explored the impact of observation noise and reward function design on the performance of Q-learning in the CartPole reinforcement learning environment and its relevance to cyber-physical systems. Our results demonstrate that observation noise significantly affects the stability and reliability of convergence. When the default reward function was used, even small amounts of noise impaired learning and control performance. In contrast, the cosine reward function

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

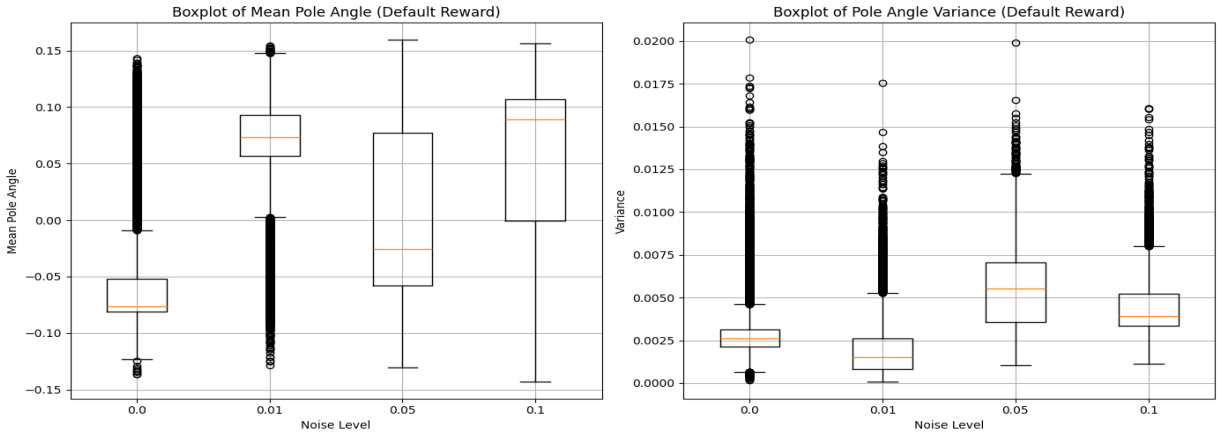

Figure 3. Q-learning box plot of default reward pole angle mean and variance.

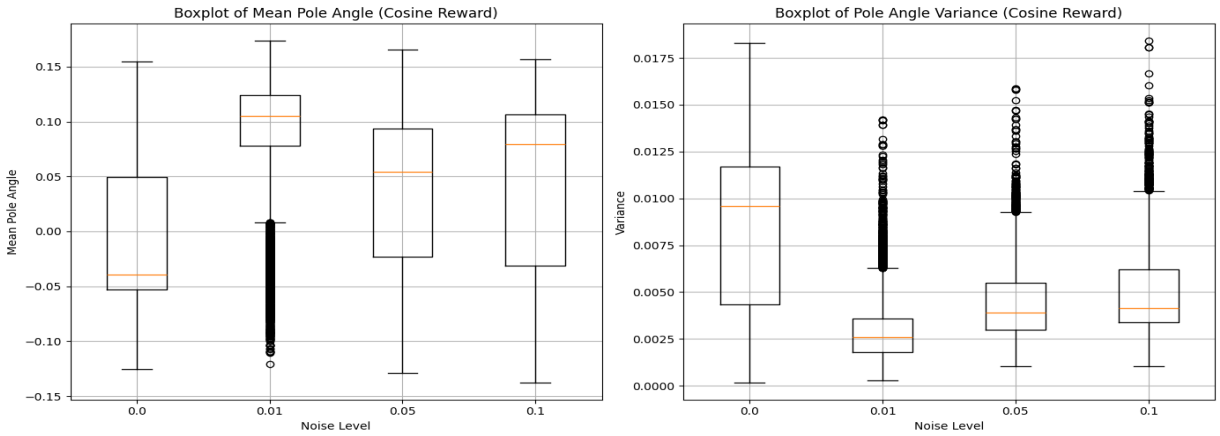

Figure 4. Q-learning box plot of cosine reward pole angle mean and variance.

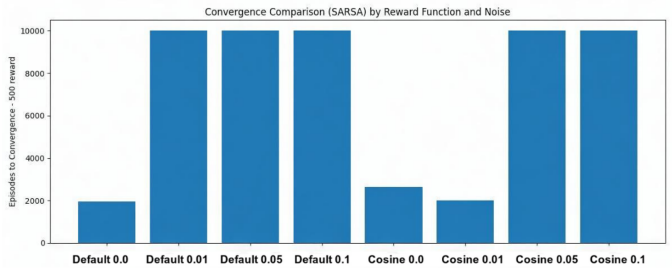

Figure 5. SARSA bar plot comparing episodes to convergence of different reward function and noise level combinations.

showed improvement in robustness, guiding the system to learn more stable policies under the noisy conditions.

Future work should extend this investigation beyond simulation by applying the experimental setup to a real-world physical system, where noise and variability are inherent and unavoidable. This would validate whether the observed benefits of different rewards translate into performance on real hardware.

Additionally, since this work used tabular Q-learning with discretized state spaces, a future direction is to examine how such methods can generalize to more complex or continuous environments. Although discretization provides interpretability and simplicity, it is often limited in scalability. Extending this framework using neural networks could bridge the tabular approach and deep reinforcement learning, enabling policies learned in idealized environments such as CartPole to generalize more effectively to higher-dimensional control tasks.

Finally, another promising direction is to develop or integrate noise detection and filtering techniques to help the system adapt its reinforcement learning process under uncertainty. Exploring combinations of reward functions, noise adaptation, and learning strategies can offer new insights into building intelligent, robust, and fault-tolerant cyber-physical systems capable of operating effectively in complex real-world environments.

ACKNOWLEDGMENT

The work is supported in partial by the National Science Foundation under NSF grants #2417608, and #2234911, #2209637, #2131228, and #2100134, Any opinions, findings, recommendations, expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

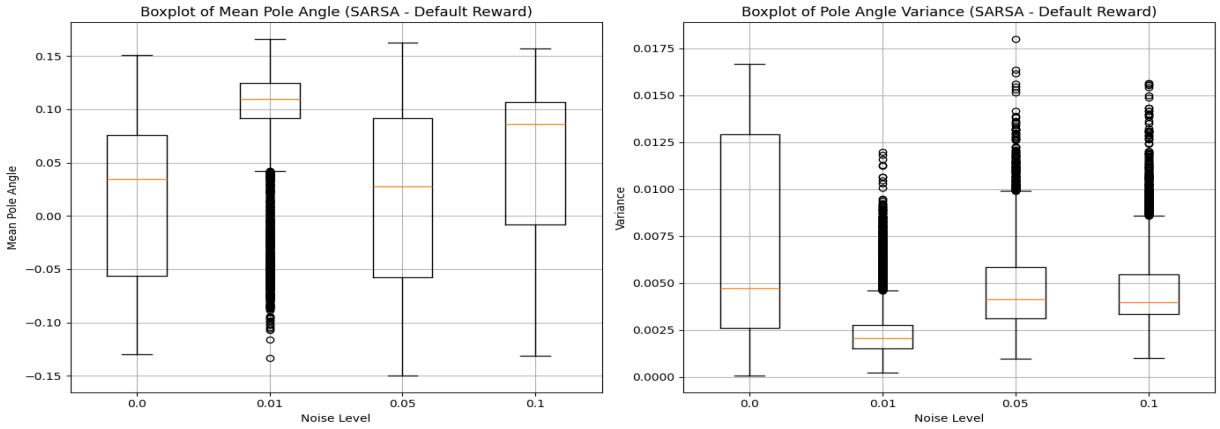

Figure 6. SARSA box plot of default reward pole angle mean and variance.

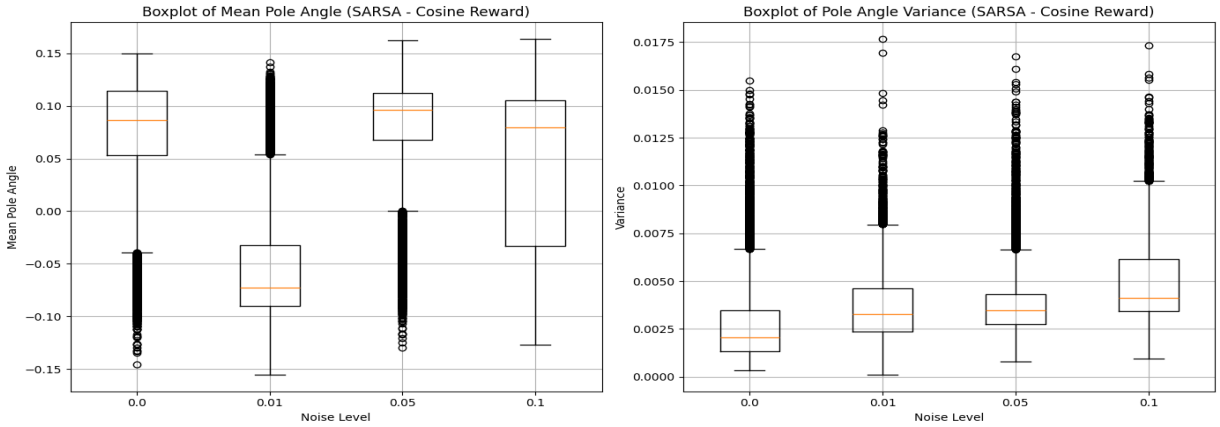

Figure 7. SARSA box plot of cosine reward pole angle mean and variance.

REFERENCES

[1] M. Towers *et al.*, "Gymnasium: A Standard Interface for Reinforcement Learning Environments," *arXiv*, 2024. Available: https://arxiv.org/abs/2407.17032.

[2] S. Kumar, "Balancing a CartPole System with Reinforcement Learning - A Tutorial," *arXiv*, 2020. Available: https://arxiv.org/abs/2006.04938.

[3] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, pp. 279–292, 1992. doi: 10.1007/BF00992698.

[4] Q. Nguyen, N. Teku, and T. Bose, "Epsilon Greedy Strategy for Hyper Parameters Tuning of A Neural Network Equalizer," in *Proc. 2021 12th Int. Symp. on Image and Signal Processing and Analysis (ISPA)*, Zagreb, Croatia, 2021, pp. 209–212. doi: 10.1109/ISPA52656.2021.9552055.

[5] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: a comprehensive survey," *Artificial Intelligence Review*, vol. 55, pp. 945–990, 2022. doi: 10.1007/s10462-021-09997-9.

[6] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," in Tsinghua Science and Technology, vol. 26, no. 5, pp. 674-691, Oct. 2021, doi: 10.26599/TST.2021.9010012.

[7] A. Maoudj and A. Hentout, "Optimal path planning approach based on Q-learning algorithm for mobile robots," Optimal path planning approach based on Q-learning algorithm for mobile robots, Applied Soft Computing, Volume 97, Part A,2020, doi: 10.1016/j.asoc.2020.106796.

[8] J. Fan, Z. Wang, Y. Xie, and Z. Yang, "A Theoretical Analysis of Deep Q-Learning," *arXiv*, 2020, Available: https://arxiv.org/abs/1901.00137.

[9] N. Gholizadeh, N. Kazemi, and P. Musilek, "A Comparative Study of Reinforcement Learning Algorithms for Distribution Network Reconfiguration With Deep Q-Learning-Based Action Sampling," in IEEE Access, vol. 11, pp. 13714-13723, 2023, doi: 10.1109/ACCESS.2023.3243549.

[10] V. Krish, A. Mata, S. Bak, K. Hobbs, and A. Rahmati, "Provable observation noise robustness for neural network control systems," Research Directions: Cyber-Physical Systems, vol. 2, p. e1, 2024. doi:10.1017/cbp.2023.5.

[11] S. Nazrul, M. Rahman, and F. Wu, "Reinforcement Learning for Adaptive Sampling: Managing Network Resources in Cloud-Based Cyber-Physical Systems," SoutheastCon 2025, Concord, NC, USA, 2025, pp. 1136-1141, doi: 10.1109/SoutheastCon56624.2025.10971596.

[12] F. AlMahamid and K. Grolinger, "Reinforcement Learning Algorithms: An Overview and Classification," 2021 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), ON, Canada, 2021, pp. 1-7, doi: 10.1109/CCECE53047.2021.9569056.

[13] G. A. Rummery and M. Niranjan, "On-line Q-learning Using Connectionist systems," Technical Report CUED/F-INFENG/TR 166, Nov. 1994. Available: https://www.researchgate.net/publication/2500611_On-Line_Q-Learning_Using_Connectionist_Systems. Accessed: September, 2025.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[15] J. Lin, Y. Ye, S. Li, H. Zhang, and P. Zhao, "Improving Exploration in Deep Reinforcement Learning for Incomplete Information Competition Environments," in IEEE Transactions on Emerging Topics in Computational Intelligence, doi: 10.1109/TETCI.2025.3555250.

[16] "CartPole Environment," Gymnasium Documentation. [Online]. Available: https://gymnasium.farama.org/environments/classic_control/cart_pole. Accessed: September, 2025.

[17] D. Cao *et al.*, "Reinforcement Learning and Its Applications in Modern Power and Energy Systems: A Review," in Journal of Modern Power Systems and Clean Energy, vol. 8, no. 6, pp. 1029-1042, November 2020, doi: 10.35833/MPCE.2020.000552.

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

Copyright (c) IARIA, 2025. ISBN: ISBNFILL