Proceedings of the 4th International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/2026.38690

Adaptive Prioritized Experience Replay for Deep Q-Networks via TD-Error Dispersion

Yuxiang Liu

Intelligent Perception Engineering, Engineering Department, Shenzhen MSU-BIT University, Shenzhen, China
1120230700@smbu.edu.cn

Abstract. Prioritized Experience Replay (PER) samples transitions in proportion to their Temporal Difference (TD) error, which improves the sample efficiency of Deep Q-Networks (DQN). Conventional PER, however, keeps the prioritization strength α fixed for the entire training run. This paper proposes a TD-error adaptive PER strategy: after each optimization step, α is reset from the standard deviation of the batch's TD errors and bounded within [0.2, 0.8], so that how strongly the buffer is prioritized changes with what the network is currently learning. The method is evaluated in CartPole-v1 against Uniform DQN and fixed-α PER with several α values. Across five seeds, the adaptive strategy showed the smallest cross-seed variance in training performance (26.92) and the highest mean reward under greedy evaluation (239.10), even though its final training score of 115.97 stayed below the best-tuned fixed-α setting. Adaptive prioritization therefore looks promising, but it has not yet beaten a carefully chosen fixed-α in a consistent way.

Keywords: Deep reinforcement learning, prioritized experience replay, deep Q-network, TD-error

1. Introduction

Reinforcement learning (RL) describes the setting in which an agent improves a policy by acting in an environment and pursuing maximized cumulative rewards. Unlike supervised learning, which requires labeled examples, RL advances only through trial and error, guided by reward signals alone [1]. Through this process, an agent can pick up sophisticated decision policies with no explicit supervision. The approach has performed remarkably well in a range of domains, including game playing [2], continuous control [3], and autonomous driving [4], which highlights its importance in real-world sequential decision-making.

Value-based approaches make up a widely used branch of deep RL. For observations of high dimensionality, Mnih et al. put forward Deep Q-Networks (DQN), where a deep neural network approximates the action-value function [5]. Since then, DQN has grown into one of the most representative value-based methods in deep RL. Experience replay was proposed to counter the instability from temporally correlated samples: past transitions are kept in a buffer and later reused, lowering the correlation between successive updates [6]. Uniform sampling, by contrast, treats every stored transition equally, even though some carry far more useful learning signal than others. To

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.38690

overcome this drawback, Schaul et al. introduced prioritized experience replay (PER), which draws transitions with a probability tied to the size of their Temporal Difference (TD) error, so that high-error samples are revisited more frequently [7]. A TD error captures the mismatch between the agent's current value estimate and its TD target. Larger TD errors signal bigger gaps between estimated and target values, so those transitions are likely to be more informative for training [7]. In PER, how strongly sampling is prioritized is set by the strength α. A larger α places more weight on high-priority experiences, whereas a smaller α brings the sampling distribution closer to uniform sampling. Mathematically, the sampling probability is proportional to the priority raised to the power of α. Conventional PER keeps α fixed over the whole training run instead of treating it as something that can vary. Whether one fixed-α can stay optimal across different stages of training, however, is still unclear.

Driven by this question, this paper proposes a TD-error adaptive PER strategy for DQN and tests it on CartPole-v1. The core idea rests on a simple observation: how informative TD errors are can shift over the course of training. At first, unstable value estimates make priorities noisy; later on, TD errors are more likely to point to truly underlearned transitions. The strategy therefore adjusts α dynamically according to the current TD-error distribution rather than fixing it in advance. Specifically, this paper compares Uniform DQN, fixed-α PER settings with different α values, and the proposed adaptive strategy, looking at how each one affects training performance, variability, and the quality of the final policy. The goal of this study is to provide empirical evidence on whether adaptive prioritization can outperform conventional fixed-α PER during training.

2. Method

2.1. Environment

The experiments used CartPole-v1 shown in Figure 1, a benchmark control task from the Gymnasium library [8]. Here the agent must keep a pole upright on a cart that can move along a horizontal track. Each step supplies four continuous observations, namely cart position, cart velocity, pole angle, and pole angular velocity, and the agent selects one of two discrete actions. Every step yields +1. An episode terminates once the pole leans more than 12°, the cart strays past ±2.4, or 500 steps elapse [8]. Since this study relies on a simulation rather than a collected dataset, the training experiences are generated directly through the agent's interaction with the environment and its underlying transition dynamics.

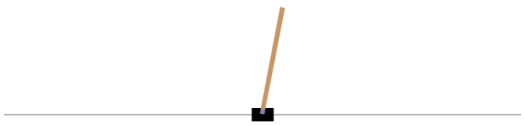


Figure 1. The CartPole-v1 environment (picture credit: original)

DOI: 10.54254/2755-2721/2026.38690

2.2. Models

This study implemented and compared three value-based methods: Uniform DQN, fixed-α PER, and the proposed TD-error Adaptive PER.

2.2.1. Deep Q-network

Deep Q-networks (DQN) pair Q-learning with a deep neural network that estimates action values, and the framework has since been extended in multiple directions [5, 9, 10]. As in Mnih et al. [5], a deep network approximated the action-value function Q(s,a), which estimates the expected return of taking action a in state s. The policy network had a 4-128-128-2 structure: four inputs, two hidden layers of 128 ReLU units each, and two outputs, with a second, identical network acting as the target, refreshed from the policy network every 10 episodes. Each experience was represented as (s,a,r,s',d), corresponding to the state, action, reward, next state, and termination flag, and was stored in a replay buffer of capacity 10,000. Mini-batches of 64 samples were drawn from this buffer for training, which reduces temporal correlations and improves sample efficiency [6].

2.2.2. Prioritized experience replay

PER samples transitions in proportion to their priority rather than uniformly [7]. In fixed-α PER, the sampling priority of each transition is determined by its TD error. For transition i, the TD error is defined as Equation (1):

$$\delta_i = r_i + \gamma(1 - d_i)\max_{a'} Q_{\text{target}}(s'_i, a') - Q(s_i, a_i) \tag{1}$$

where r_i denotes the reward, γ is the discount factor, d_i is the termination indicator, s'_i is the next state, Q_{target} denotes the action value estimated by the target network, and Q represents the action value estimated by the online network. The priority for transition i is then defined as Equation (2):

$$p_i = |\delta_i| + \epsilon \tag{2}$$

where $\epsilon = 10^{-5}$ prevents the priority from becoming zero, ensuring that every transition keeps a non-zero chance of being sampled [7]. The sampling probability for transition i is proportional to its priority raised to the power of α, defined as Equation (3):

$$P(i) = \frac{p_i^\alpha}{\sum_j p_j^\alpha} \tag{3}$$

where the prioritization strength α controls how strongly high-priority transitions are favored, and when $\alpha = 0$, sampling falls back to uniform sampling [7]. To correct the bias brought in by non-uniform sampling, importance sampling weights, Equation (4):

DOI: 10.54254/2755-2721/2026.38690

$$w_i = (N \cdot P(i))^{-\beta} \tag{4}$$

were employed, where N denotes the total number of transitions stored in the replay buffer, and β increases linearly from 0.4 to 1.0 over the course of training [7].

2.2.3. TD-error adaptive PER

Unlike conventional PER, which keeps α fixed, the proposed method allows α to change during training, guided by the spread of the TD errors. This approach is conceptually related to prior work on adaptive experience prioritization [11]. After each optimization step, the standard deviation σ_δ of the batch's TD errors was computed, and α was set to

$$\alpha = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})(1 - e^{-k\sigma_\delta}) \tag{5}$$

with $\alpha_{\min} = 0.2$, $\alpha_{\max} = 0.8$, and $k = 1.0$. At the very beginning, before reliable statistics existed, α was initialized to 0.4. Equation (5) works as follows. When the batch's TD errors are spread apart, σ_δ is large and α rises toward $\alpha_{\max}$, so sampling favors high-priority transitions more strongly. When the errors are close together, σ_δ is small and α falls toward $\alpha_{\min}$, so sampling becomes closer to uniform. In this way, the strength of prioritization follows what the network is actually learning, instead of a schedule decided in advance.

2.3. Hyperparameter configuration and evaluation

For a fair comparison, the three methods shared the same architecture, optimizer, and training budget, so performance gaps trace back only to the sampling strategy. The Adam optimizer [12] ran with a learning rate of 1×10^{-3}, and the objective was Huber loss (Smooth L1 loss), with gradients clipped to a maximum norm of 10. The discount factor γ was 0.99. Exploration used an ϵ-greedy policy, where ϵ started at 1.0, decayed by a factor of 0.995 after every episode, and stopped at a floor of 0.01. For fixed-α PER, four α values were tested, $\{0.2, 0.4, 0.6, 0.8\}$; for the adaptive method, α stayed within [0.2, 0.8] and changed with the TD-error dispersion. Every configuration was trained for 500 episodes on five random seeds, $\{1, 7, 42, 123, 2024\}$. The training results were summarized by three metrics: Final-100 Mean (the average reward over the last 100 episodes), Training Mean (the average over all 500 episodes), and Mean Maximum Reward (the largest reward in each run, averaged across seeds). The standard deviation of Final-100 Mean across seeds was also reported to show cross-seed variability. After training, exploration was switched off, and each model was evaluated for 20 greedy episodes with a fixed evaluation seed, so that the score reflected the learned policy rather than any leftover exploration.

DOI: 10.54254/2755-2721/2026.38690

3. Results and discussion

3.1. Fixed-α PER comparison

The four fixed-α PER configurations were evaluated first to establish a baseline. Table 1 lists the training performance of every configuration across the five seeds. Among them, $\alpha = 0.4$ posted the highest Final-100 Mean (175.85) and therefore served as the fixed-α baseline. Performance did not rise steadily with α: $\alpha = 0.8$ had the highest Training Mean (99.23), yet its final score did not follow suit. A stronger prioritization, in other words, was no guarantee of a better late-stage policy.

Table 1. Training performance across five random seeds

Method	Final-100 Mean	Cross-seed SD	Training Mean	Mean Max Reward
PER α=0.4	175.85	76.01	86.35	411.00
PER α=0.2	156.29	55.92	84.87	434.20
PER α=0.8	140.61	81.81	99.23	454.20
Uniform DQN	139.93	33.88	74.22	413.80
PER α=0.6	120.35	44.92	81.76	446.40
TD-error Adaptive PER	115.97	26.92	89.96	489.40

3.2. Training performance comparison

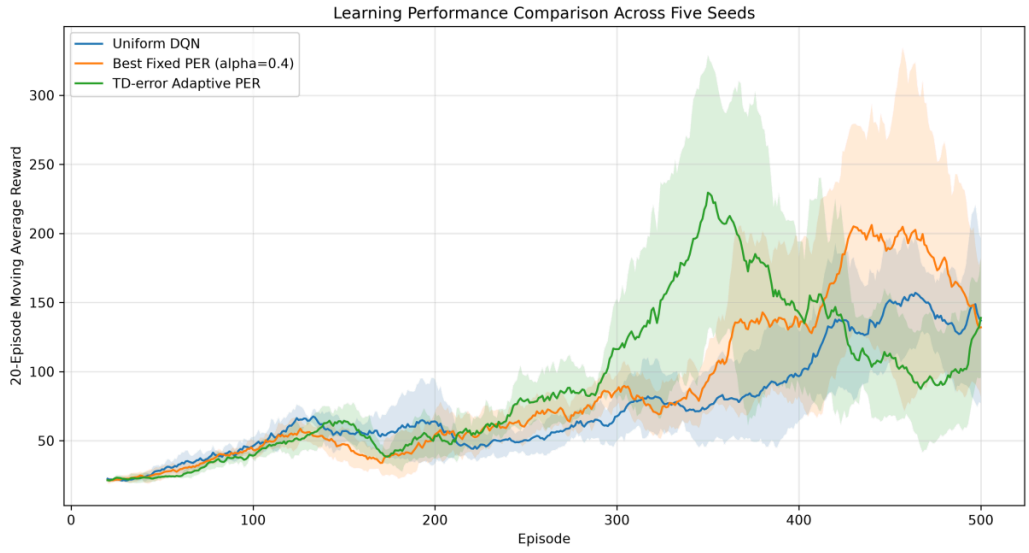


Figure 2. Learning performance comparison across five seeds (picture credit: original)

Uniform DQN ended with a Final-100 Mean of 139.93 and a cross-seed standard deviation of 33.88, so its final performance was reasonably steady across seeds. That score, however, still fell short of the best fixed-α setting. The proposed TD-error Adaptive PER reached a Final-100 Mean of 115.97, below the best fixed-α PER, but its cross-seed standard deviation was only 26.92, which points to an even steadier final performance. Its Training Mean came to 89.96, close to the 86.35 of the best fixed-α PER and above the 74.22 of Uniform DQN. It also posted the highest Mean

DOI: 10.54254/2755-2721/2026.38690

Maximum Reward (489.40) of all the configurations, meaning it occasionally reached very high reward levels. Figure 2 compares the learning curves of the three primary methods across the five seeds.

3.3. Adaptive α dynamics

Figure 3 shows how the adaptive α evolved, averaged across the five seeds. Starting at 0.4, the mean rose through training and then stayed near the 0.8 ceiling for most of the run. This saturation means the adaptive method behaved much like a high-α fixed PER in the later stages, leaving it little room to keep adjusting α.

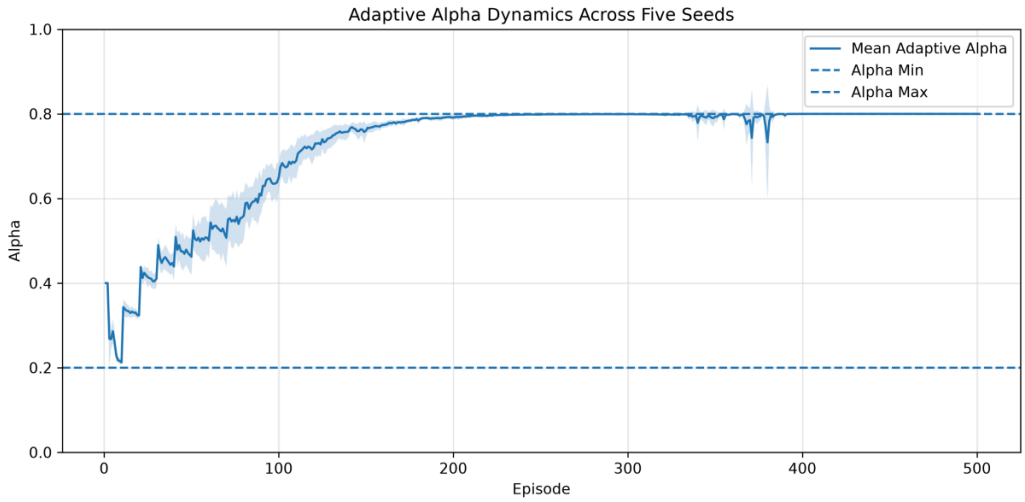


Figure 3. Adaptive α dynamics across five seeds (picture credit: original)

3.4. Greedy policy evaluation

The TD-error Adaptive PER produced the highest mean greedy reward, 239.10 against 132.75 for the best fixed-α PER and 114.30 for Uniform DQN (Table 2). Two of its five models even reached the maximum reward of 500. In some runs, then, the adaptive method was capable of a much stronger learned policy. Its cross-seed standard deviation, however, was the highest of the three at 163.29, so policy quality swung widely from one seed to the next. The high average was therefore pulled up by a few very strong runs rather than earned consistently, and the final policy remained highly sensitive to the random seed. Stepping back, no single replay strategy won on every metric. Adapting the prioritization strength from TD-error dispersion is clearly a workable idea, and it can produce stronger policies in some runs, but against a well-tuned fixed-α PER these gains did not show up consistently across seeds.

Table 2. Greedy policy evaluation across five seeds

Method	Evaluation Mean	Cross-seed SD	Mean Eval Max
TD-error Adaptive PER	239.10	163.29	358.60
Best Fixed-α PER (α=0.4)	132.75	37.70	175.20
Uniform DQN	114.30	49.89	151.40

DOI: 10.54254/2755-2721/2026.38690

4. Conclusion

This paper studied whether the prioritization strength of PER should stay fixed or change during training in DQN. A TD-error adaptive strategy was proposed, which recomputes α after every optimization step from the standard deviation of the batch's TD errors, rather than holding α constant. The method was trained in the CartPole-v1 environment and compared with Uniform DQN and four fixed-α PER configurations across five random seeds. The adaptive strategy delivered the lowest cross-seed variance in training performance and the highest mean reward under greedy evaluation, which means its prioritization really does follow the learning and can occasionally reach a much stronger policy. At the same time, α tended to saturate near its upper bound in the later stages, leaving little room for further adjustment, and the improvement over the best-tuned fixed α did not appear consistently across seeds. Future work could explore smoother adaptation schedules and ways to keep α from saturating, and test the approach on environments harder than CartPole.

References

[1] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.
[2] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature, 529*(7587), 484–489.
[3] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2016). Continuous control with deep reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
[4] Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., & Pérez, P. (2022). Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems, 23*(6), 4909–4926.
[5] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529–533.
[6] Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning, 8*(3–4), 293–321.
[7] Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2016). Prioritized experience replay. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
[8] Farama Foundation. (n.d.). Cart pole. *Gymnasium documentation*. Retrieved from https://gymnasium.farama.org/environments/classic_control/cart_pole/
[9] van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
[10] Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., & Silver, D. (2018). Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
[11] Hassani, H., Nikan, S., & Shami, A. (2025). Improved exploration-exploitation trade-off through adaptive prioritized experience replay. *Neurocomputing, 614*, 128836.
[12] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.