Enhanced Experience Replay Rate Dual-Policy Interactive Decision-Making

Yan Kong[1], Guoliang Che[1], Chih-Hsien Hsia[2,3]*

[1] School of Software, Nanjing University of Information Science and Technology, China
[2] Department of Computer Science and Information Engineering, National Ilan University, Taiwan
[3] Department of Business Administration, Chaoyang University of Technology, Taiwan
kongyan4282@163.com, guoliangche@gmail.com, hsiach@niu.edu.tw

Abstract

Reinforcement learning (RL) often encounters significant challenges in resource efficiency, particularly in complex environments with high-dimensional state spaces. These environments require substantial computational resources, including processor time and memory, due to the frequent interactions between the agent and the environment. A promising strategy to improve sample efficiency is to increase the replay rate—the number of parameter updates per environment interaction. However, this approach can introduce instability and lead to suboptimal learning outcomes. To address this issue, we propose a dual-policy interaction decision framework designed to mitigate the adverse effects of high replay rates. Our method employs two policies to effectively balance exploration and exploitation, ensuring both stability and efficiency in learning, even at elevated replay rates. We evaluate our approach on Box2D and MuJoCo continuous control benchmarks, demonstrating significant improvements in sample efficiency and training stability. Specifically, our results show a 20% reduction in training time and a 15% increase in task performance compared to conventional single-policy approaches. By integrating dual-policy interaction decisions, our framework enables agents to learn more effectively from past experiences, thereby reducing computational overhead while maintaining or enhancing performance. This study highlights the potential of dual-policy architecture as a robust solution for improving the efficiency and scalability of deep RL algorithms in resource-intensive environments.

Keywords: Reinforcement learning, Replay rate, Sample efficiency, Dual-policy interaction decisions

1 Introduction

In reinforcement learning (RL) training, it is typically required that the agent interacts with the environment for one round before undergoing a training session. However, these interactions are often resource-intensive, and in real-world applications, each interaction incurs a cost. For instance, autonomous vehicles guided by RL algorithms cannot afford excessive trial-and-error attempts due to safety and operational constraints. Therefore, deep reinforcement learning (DRL) algorithms must be learned with a minimal number of samples [1]. If an algorithm can leverage additional computational resources during training to enhance performance, this can be achieved either naturally or by increasing the frequency of training sessions using collected experience data, thereby improving sample utilization [2]. One effective approach to achieving this is extended training on a given experience dataset before the agent interacts with the environment again.

Increasing the frequency of experience replay (ER), *i.e.*, the number of times the agent's parameters are updated per interaction with the environment, enables the agent to acquire knowledge earlier in the learning process. While this approach provides limited benefits when applied to standard baselines [3-4], it has been shown to enhance the performance of well-tuned algorithms. Increasing the replay ratio in the soft actor-critic (SAC) algorithm improves performance in the early stages of two continuous control environments compared to single-replay training. However, this also introduces greater reward fluctuations and does not exhibit a purely positive correlation with replay frequency. Recent studies have demonstrated improved sample efficiency by increasing the replay ratio to up to eight for discrete control and twenty for continuous control [5-6].

Increasing the frequency of ER can enhance an agent's performance in certain environments. However, the relationship is not strictly positive, as it may introduce challenges such as first-cause bias [7-9], where the agent overfits and experiences large fluctuations in return, making convergence difficult. To address the issue of primacy bias, a common approach is to reset network weights to mitigate its effects. The primacy bias problem refers to the phenomenon where early experiences stored in the replay buffer disproportionately influence policy updates, resulting in suboptimal convergence. This issue arises because high replay rates amplify the impact of initial (potentially noisy) samples, causing the agent to overfit to early trajectories. For example, in environments with sparse rewards, agents may repeatedly replay early, suboptimal experiences, leading to significant fluctuations in returns and hindering convergence to an optimal policy.

*Corresponding Author: Chih-Hsien Hsia; Email: hsiach@niu.edu.tw
DOI: https://doi.org/10.70003/160792642026112706005

In this work, we propose a dual-decider network framework to mitigate the impact of increasing replay rates. As illustrated in Figure 1, our approach adopts an actor-critic structure but incorporates two policy networks. Policy No.1 adjusts its sampling frequency from the replay buffer, requiring an increased replay rate. However, this adjustment is constrained by time steps and is not applied uniformly throughout the entire process. Within a predefined time step, the agent samples from the replay buffer multiple times to update Policy No.1, while Policy No.2 is updated only once. During decision-making, the framework dynamically selects a policy based on specific conditions to generate actions. Additionally, the framework incorporates two policy networks with distinct roles. Policy No.1 serves as the early decision network, learning with a higher replay frequency and exhibiting a more exploratory nature. In later stages, Policy No.2 alternates with Policy No.1, adopting a more conservative approach to exploration. Further details are provided in Section 4.

The structure of this paper is as follows: Section 2 reviews related work, while Section 3 provides background information. Section 4 explores the theoretical foundations of the proposed algorithm, focusing on the integration of the EDM framework with the SAC algorithm. In Section 5, the proposed approach is evaluated in MuJoCo [10] and Box2D [11] environments, with detailed experimental results presented. Finally, Section 6 concludes this study.

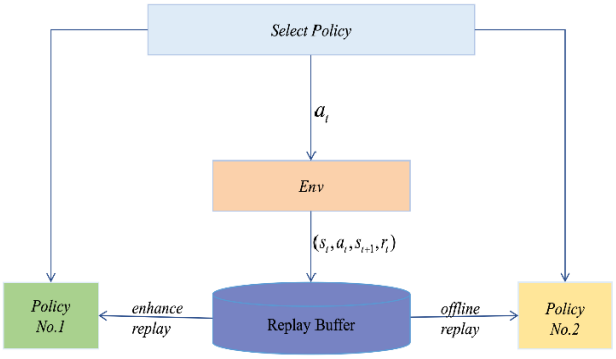

Figure 1. Framework of the EDM algorithm

2 Related Work

In off-policy DRL, an ER buffer is commonly used as a key component for policy optimization. This buffer stores experiences collected by different policies over time [12-13]. ER enhances the stability and performance of policy optimization by enabling the agent to repeatedly reuse previously stored experiences, performing gradient updates on the deep neural network that approximates both the policy and value function [14]. The initial concept of ER involved uniform sampling from the buffer. However, in recent years, various improvements have been proposed to enhance sampling efficiency and optimize sample utilization. For example, prioritized ER [15], a non-uniform sampling method, selects transitions with higher TD-error values, increasing the likelihood of sampling those experiences that are more informative. Another advancement, attention-based ER [16], assigns different importance weights to experiences based on an attention mechanism, selecting the most valuable samples, and improving learning efficiency and performance. Furthermore, hindsight ER [17] is particularly effective in environments with sparse or binary rewards, eliminating the need for complex reward function engineering.

In the study addressing the primacy bias problem [7], increasing the replay rate can improve performance on certain tasks. However, this leads to the Primacy Bias phenomenon, where the actor overfits to early experiences, causing the policy to suddenly collapse during a particular interval. This results in a sharp, cliff-like drop in the reward curve. To mitigate this issue, the authors propose resetting the network parameters at appropriate intervals and allowing the agent to resume interaction with the environment at a high replay rate after updating the network parameters through sampling from the experience buffer. This strategy effectively eliminates primacy bias and improves overall learning stability.

Research that employs multiple policies for exploration or decision-making, such as the centralized cooperative exploration policy for continuous control tasks (CCEP) [18], can enhance exploration across a broader region of the task space. CCEP generates diverse policies with varying exploration styles from a pair of value functions, facilitating collaborative exploration of the environment. Additionally, in distributed algorithms like asynchronous advantage actor-critic (A3C) [19], multiple actors interact with the same critic in parallel. Each actor learns its policy, while the critic shares parameters and learns a value function, among other components [20]. This parallelization enables more efficient exploration and learning by leveraging multiple agents' experiences.

3 Preliminaries

This study explores the efficiency improvement of reinforcement learning (RL) algorithms through increased replay rates and dual-policy interaction decision-making. To ensure generalizability, we selected MuJoCo and Box2D as experimental environments and integrated replay rates and dual-policy interaction decision-making techniques into the conventional actor-critic (AC) algorithm, as depicted in Figure 1. It is important to note that this setup is merely an illustrative example, and the proposed algorithm can be extended to a wide range of scenarios. By incorporating increased replay rates, the agent can learn more efficiently from historical interaction data, accelerating the training process and enhancing decision-making effectiveness. Furthermore, the dual-policy interaction decision-making mechanism optimizes algorithm performance by facilitating coordination between different policies. Our results demonstrate significant improvements in both sample efficiency and training stability.

3.1 Actor Critic Algorithm

DRL introduces deep neural networks as function

approximators on top of standard RL, leveraging the powerful representation capabilities of deep neural networks for better optimization of RL algorithms. The RL paradigm is represented by a finite Markov decision process with five components: state space S, action space a, deterministic or stochastic reward function R, environment dynamics model P, and discount factor $\gamma \in [0,1)$. The performance of a policy π is evaluated under the action-value function (Q-function or critic) Q^{π}, which represents the expected discounted sum of rewards when acting as in state s and following policy π:

$$Q^{\pi}(s,a) = E_{\pi}[\sum_{t=0}^{\infty} \gamma_{r_{t+1}}^{t} \mid s_0 = s, a_0 = a] \quad (1)$$

Regarding the action-value function, it is determined by the Bellman equation [21]:

$$Q^{\pi}(s,a) = E_{r,s'\sim P,a'\sim\pi}[r + \gamma Q^{\pi}(s',a')] \quad (2)$$

where a' denotes the next action chosen by the policy in the observed next state s'.

The AC is an RL method that combines policy gradients and temporal difference learning (TD). In this algorithm, the "actor" refers to the policy function $\pi_{\theta}(a|s)$, which learns a strategy to maximize returns. The "critic" refers to the value function $V_{\pi}(s)$, which estimates the value of the current policy, essentially evaluating the performance of the actor. With the aid of the value function, the AC algorithm can perform single-step parameter updates, without the need to wait for the end of an episode for updates. In AC algorithm, the critic network is approximated by a deep neural network Q_{θ} with parameters θ. Then, Q-learning with a deep function approximator is transformed into the standard DQN algorithm. Given a transition tuple $\tau = (s, a, r, s_0)$, DQN is trained by minimizing the loss $\delta_{\theta}(\tau)$ based on the temporal difference error with respect to the Q network Q_{θ}.

The TD learning algorithm is an update rule based on the Bellman equation, which defines the basic relationship used to learn the action-value function by bootstrapping from the value estimate of the current state-action pair (s', a') to the subsequent state-action pair (s', a'):

$$y(\tau) = r + \gamma Q_{\theta'}(s',a') \quad (3)$$

$$\delta_{\theta}(\tau) = y(\tau) - Q_{\theta}(\tau) \quad (4)$$

Note that the TD error in the latter equation can also be viewed as (4), but this does not affect our analysis. Transitions $\tau \in B$ are sampled from an ER buffer using some specific sampling method, which contains a set of experiences in the form of transition tuples B collected previously. The target $y(\tau)$ in (3) is stabilized and kept fixed during the learning of the optimal Q function by utilizing a separate target network with parameters Q_{θ}. To update the target parameters, the parameter θ is either copied a small fraction ζ from the Q network Q_{θ} at each step or matched separately at fixed intervals using soft or hard updates. At each update step, the loss of the Q network is averaged over the sampled transition batch B: $\frac{1}{|B|}\sum_{\tau\in B}\delta_{\theta}(\tau)$, where $|B|$ is the number of transitions included in B. Note that the batch size does not affect our analysis of priorities, as the expected gradient remains unchanged.

3.2 Soft Actor Critic

SAC is an off-policy algorithm based on AC in policy gradients [22]. It is maximizes the expected reward value $\sum_{t}\mathbb{E}_{(s_t,a_t)\sim p_{\pi}}\left[r(s_t,a_t)\right]$ while also maximize the expected value $H(x) = -\sum_{i} x_i \log(x_i)$ of information entropy $H\left(\pi(\cdot \mid s_t)\right)$. Therefore, the objective function to be optimized is the sum of two terms:

$$J(\pi) = \sum_{t}\mathbb{E}_{(s_t,a_t)\sim p_{\pi}}\left[r(s_t,a_t) + \alpha H(\pi(\cdot \mid s_t))\right] \quad (5)$$

where α denotes called temperature, which controls the proportion of information entropy.

SAC introduces information entropy as a way to balance exploration and exploitation, enhancing environmental exploration and facilitating better information collection. By leveraging importance sampling techniques, SAC allows the agent to learn from past data, which significantly contributes to its strong generalization performance. The optimization objective of SAC is rooted in maximum entropy theory, enabling the agent to learn maximum entropy policies across various environments. This approach makes SAC more robust in adapting to and handling unknown changes in the environment. Through importance sampling, SAC effectively utilizes past experiences, applying them to new environments and thereby improving the algorithm's ability to generalize. As a result, SAC is an efficient, flexible, and widely applicable DRL-based algorithm that has demonstrated success in a variety of fields.

4 Proposed Enhanced Experience Replay Rate Dual-Policy Interactive Decision-Making

4.1 Enhanced Experience Replay Rate

ER is a widely used technique in DRL to address the challenges of sample correlation and non-stationarity in RL algorithms. In ER, the agent stores its experiences in a buffer called the Replay Buffer. Instead of using the most recent experience for training, the agent samples a certain amount of experience randomly from the buffer for training. This helps to reduce sample correlation, lower variance, and improve the convergence and generalization

of the algorithm. Furthermore, ER reduces the frequency of interactions with the environment, enhancing the overall efficiency of the learning process. To fully utilize the experiences stored in the buffer and accelerate the agent's learning, we increase the replay frequency within certain constraints, allowing the agent to learn more effectively from past experiences.

As shown in Figure 2, in the later stages of learning, the agent's exploration of the environment reaches its limits, and the freshness of the experiences stored in the replay buffer decreases. Continuing to increase the replay frequency at this stage not only wastes computational resources but also increases the likelihood of overestimating Q-values, leading to decision-making errors by the agent. As a result, the reward curve in the later stages exhibits excessive fluctuations. In some environments, the agent may explore less in the early stages, which leads to high correlation among the samples in the replay buffer. In such cases, increasing the replay frequency can cause the agent to overfit to certain actions, further diminishing the effectiveness of the learning process.

We hypothesize that the training effectiveness T may exhibit a nonlinear relationship with D (the replay buffer size) and F (the replay frequency). This relationship can be expressed as:

$$T \propto D \cdot F \cdot (1 - Risk\ of\ Overfitting) \tag{6}$$

Here, the risk of overfitting is associated with excessively high F or excessively low D.

To address these issues, we constrain the agent to enhance the replay frequency only within a specific time step interval while maintaining a single update frequency in the remaining intervals. This approach ensures that the agent's learning is balanced and avoids the negative effects of excessive replay frequency. Therefore, it is important to appropriately select both the replay frequency and the replay interval. Increasing the replay frequency according to the complexity of the environment can be beneficial, but excessively increasing it is counterproductive. When selecting the replay interval, it is recommended to focus on the early or mid-training stages or the phase when the agent begins to receive increasing rewards. At this point, the replay buffer has accumulated enough experience, and the agent is in the fast-learning phase. Enhancing the replay frequency during this interval enables the agent to learn more efficiently. The improvement in training effectiveness through increased replay frequency depends on the diversity of samples in the replay buffer and the familiarity of the agent with those samples. By increasing the replay frequency during intervals of greater diversity and familiarity, the agent can accelerate its learning process.

To summarize, this specific interval is referred to as the enhancement range, which is primarily controlled by two parameters: the sampling update frequency and the enhancement interval. Sampling update frequency: This parameter defines how often updates occur after sampling from the experience pool. Typically, this frequency is kept to fewer than 10 updates. The enhancement interval is the number of steps the agent interacts with the environment before the sampling update frequency begins to change. The enhancement interval starts and ends at specific points during the agent's training process. Typically, the agent enters the enhancement interval after interacting with the environment for a predefined number of time steps. By controlling these parameters, the agent can optimize its learning process, increasing its efficiency during the fast-learning phase while avoiding the negative effects of excessive replay frequency.

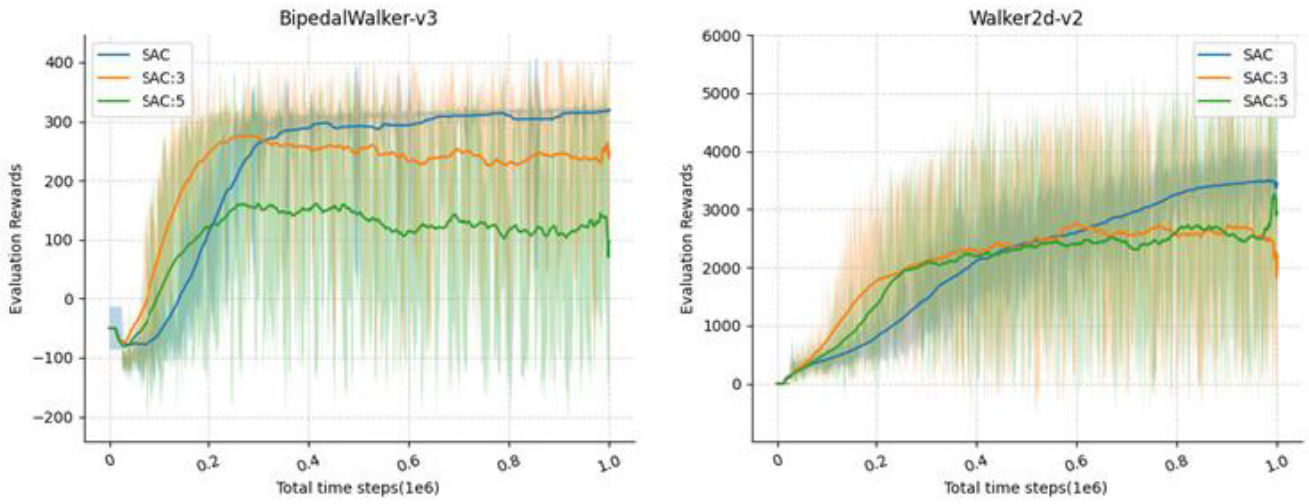

Figure 2. Results of SAC algorithm with different ER frequencies in BipedalWalker-v3 in Box2D environment and Walker2d-v2 in Mujoco environment (5 seeds, 97.5% bootstrapping)

4.2 Dual-Policy Interactive Decision-Making

An agent with an enhanced replay frequency may repeatedly learn the same transitions, causing the actor network to overfit to certain actions. This overfitting results in the agent making decision errors during its trajectory, particularly in certain states. As a result, the agent may start picking suboptimal actions or even poor moves, leading to an overall poor return in performance. Over-training, particularly due to excessively frequent replay, can cause the actor network to lose its generalization ability. This loss of generalization makes it difficult for the agent to adapt to new or unseen situations, and once overfitting has occurred, it becomes hard to recover and correct the learning process. To address these issues, we introduce an **interactive decision-making mechanism** involving two distinct policies. This dual-policy approach not only enhances **sample efficiency** but also improves **stability**, particularly in **complex environments**. By leveraging two policies with different exploration-exploitation strategies, the agent can better balance between learning from past experiences and exploring new actions. This mechanism mitigates overfitting and helps maintain generalization, resulting in more stable and efficient learning, especially when replay frequencies are high.

The proposed dual-policy framework consists of two distinct policies: Policy No.1, an exploratory policy with a high replay frequency, and Policy No.2, a more conservative policy with a lower replay frequency. This design allows the agent to dynamically balance exploration and exploitation. In the early stages of training, Policy No.1 actively explores the environment, interacting frequently and generating diverse experiences. Meanwhile, Policy No.2 learns from the experiences generated by Policy No.1, focusing on refining the learned policy. This decoupling of exploration and exploitation helps prevent the agent from overfitting to early experiences and mitigates the primacy bias problem. As training progresses, Policy No.2 gradually takes over the decision-making process. With a more conservative strategy, it leverages the accumulated experiences to refine the policy, ensuring stable convergence. This dual-policy approach enhances both sample efficiency and training stability, leading to more effective learning in complex environments. Another thing worth considering is that, as shown in Figure 1, after the preconditions are met, one of the two policies needs to be selected for decision-making. Regarding the selection of interaction conditions, we consider a weighted comparison between the current return and the mean return in the previous interval to decide whether to replace the decision network.

Taking (7) as the judgment condition, the strategy is selected by comparing the mean value of rewards and the reward value within the current range.

$$\text{Mean}(R') > R \ \pi_{\phi} \text{ or } \pi_{\theta} \tag{7}$$

Algorithm 1. Enhanced ER rate dual-policy interactive decision-making

Initialize Actor π_{φ} , π_{θ} , *Critic Q network*, $D = \varnothing$
for *each episode* ***do***
 for *each environment step* ***do***
  $a_t \sim \pi_{\varphi}(a_t|s_t)$
  s_{t+1} , $r_t \sim p(s_{t+1}|s_t, a_t)$
  $D \leftarrow (s_t, a_t, r_t, s_{t+1})$
  if step in enhance range and eq(7) **then**
   $a_t \sim \pi_{\theta}(a_t|s_t)$ ▷ *interactive decision*
  end if
 end for
 for *each gradient step* ***do***
  update π_{φ} *and Q by via D* ▷ *update policy*
  Update π_{θ} *by offline training via D*
 end for
end for

5 Experimental Results

5.1 Experiments Details

To validate the effectiveness of our method and assess its impact on improving algorithm performance in continuous control environments, we conducted experiments by combining the EDM framework with the SAC algorithm, referred to as EDM-SAC. We then compared this approach with the original SAC algorithm on standard continuous control tasks in the MuJoCo and Box2D environments using the OpenAI Gym [23] interface. For the implementation of SAC, we adopted the version from Actor Prioritized ER (LA3P) [24], which incorporates entropy adjustment as suggested by Haarnoja *et al*. [22] to improve overall performance. In line with recommendations by Fujimoto *et al*. [25], we included 25,000 exploration time steps before the actual training began to enhance data efficiency. Additionally, we strictly adhered to the hyperparameter settings outlined in the original SAC paper to ensure a fair comparison. Each method was trained for one million steps with five random seeds (0-4) to account for variability in network initialization, simulator conditions, and dependencies. This experimental setup allowed us to assess the performance and efficiency improvements of the EDM-SAC framework in a controlled and consistent manner.

Regarding the enhancement range, the sampling update frequency can be appropriately increased to 2-3 times, but higher values may hinder policy convergence. Selecting the enhancement interval in the early to middle stages of training is beneficial for accelerating learning. In our experiment, the enhancement interval was set to begin after 100,000 time steps and end between 400,000 to 500,000 time steps. After the enhanced replay phase ends, a desensitization period is introduced, during which normal sampling is resumed for a certain duration. During this period, the agent's interactive decisions are made based on the evaluation of conditions. For the two policies,

Policy No.1 is designed to have stronger exploration capabilities and can incorporate components like Gaussian noise to encourage diverse exploration. In contrast, Policy No.2 is designed to maintain stability and typically uses deterministic policies to refine decision-making based on accumulated experiences. Every 1,000 steps, each method was evaluated in a different evaluation environment (with a fixed training seed and constants), where no exploration or learning took place. The learning curve of the algorithm is presented below, with rewards smoothed using a sliding window of fifteen to reduce noise and highlight performance trends.

5.2 Evaluation

From our evaluation results, as shown in Figure 3, we can observe that our algorithm performs better than competing algorithms in the tested environments. In environments such as bipedalwalker (BW) and lunar lander continuous (LL), where convergence is fast, our algorithm accelerates the convergence rate. However, observations of the shaded areas reveal that our algorithm still exhibits significant fluctuations, and the range of reward intervals is significantly higher than that of the original SAC algorithm.

In more difficult environments with longer horizons, such as Ant, Walker2d, Swimmer, and Humanoid, our algorithm outperforms the comparison algorithms, and the difference is even more pronounced. Again, the shaded areas are larger than those of the original SAC, indicating greater potential in terms of reward upper bounds. In the environments of Ant, Walker2d, and Swimmer, the evaluation demonstrates an overall improvement compared to the standard algorithm.

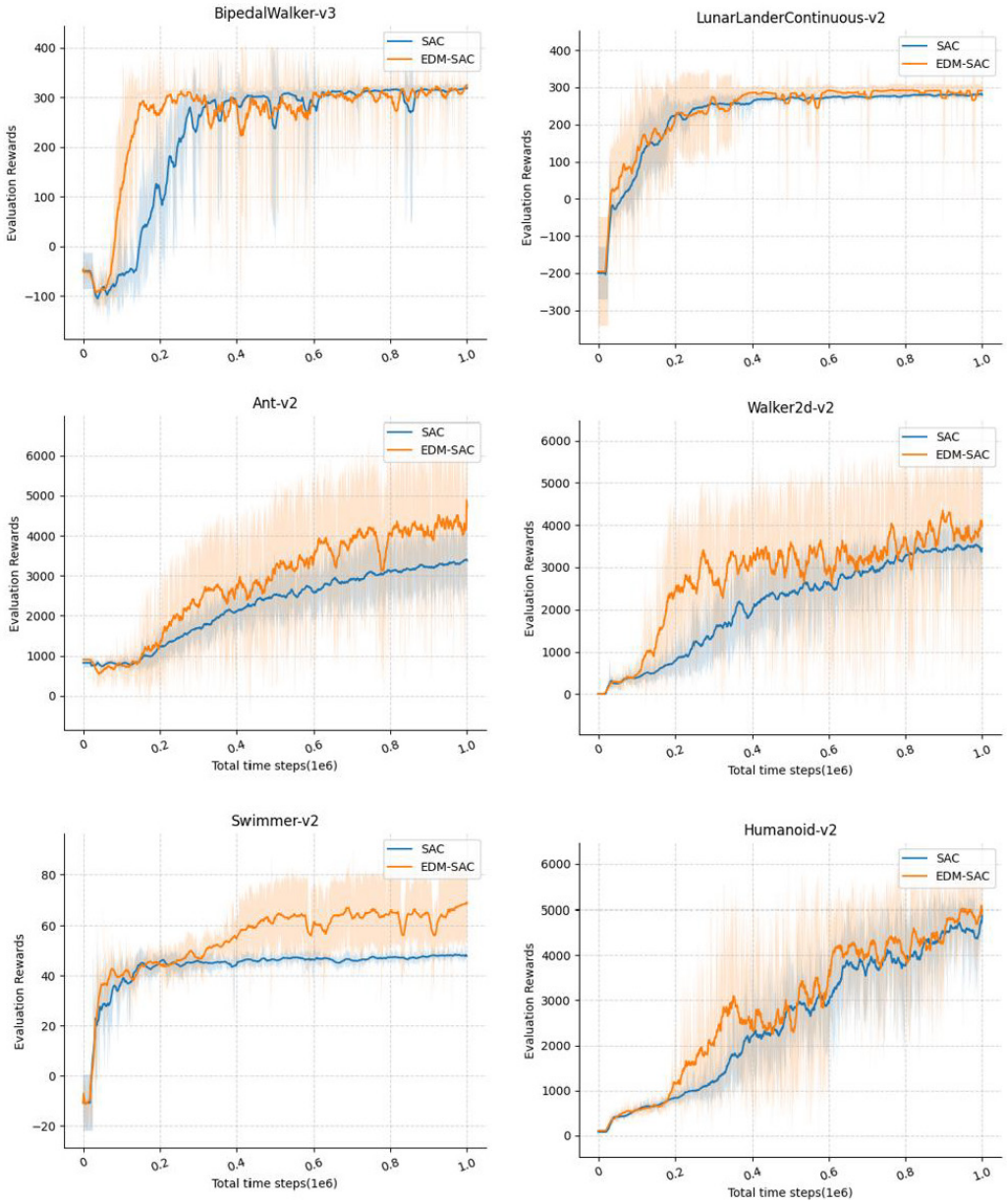

Figure 3. Learning curves for the set of MuJoCo and Box2D continuous control tasks under the SAC and E_SAC algorithm (A sliding window of size 15 smoothers the curves for visual clarity. Each curve is made using 5 seeds.)

The sample efficiency gains (30-40% fewer steps to match SAC's performance) are particularly valuable for real-world applications where physical trials are costly. For example, in quadruped robot training, EDM-SAC could reduce hardware wear-and-tear by minimizing exploration phases while ensuring policy diversity through structured replay scheduling. This aligns with recent analyses of primacy bias in RL, where excessive replay of early experiences limits long-term optimization. By decoupling data collection (Policy No.1) and optimization (Policy No.2), EDM-SAC mitigates this issue, enabling sustained policy improvement, evidenced by the monotonically increasing reward curves in Swimmer and Ant beyond 1M steps. As results indicate that our algorithm can improve the performance of the algorithm in some environments and can be used as a method to enhance algorithm performance. In Figure 4, EDM-SAC can achieve the same level of performance as regular SAC in fewer time steps.

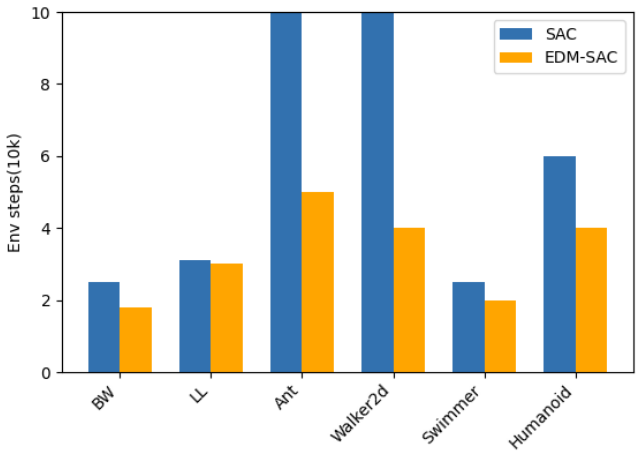

Figure 4. Data efficiency description (The time steps required to achieve similar performance between standard SAC and EDM-SAC.)

The EDM framework demonstrates practical promises in safety-critical applications, such as industrial robotics (e.g., high-DOF manipulator control) and autonomous vehicles (e.g., dynamic navigation). These applications require high efficiency in decision-making and adaptability to dynamic, real-world environments, areas where EDM's dual-policy decision-making can significantly enhance performance. Future work will focus on expanding the EDM framework to include multi-agent coordination, allowing multiple agents to collaborate or compete in complex environments. Additionally, cross-domain generalization via meta-learning will be explored to improve the framework's adaptability across various tasks and environments. Furthermore, edge-device deployment through hybrid neural-symbolic architectures will bridge the gap between efficient training and real-world robustness, making the algorithm suitable for practical, real-time applications on resource-constrained devices. These advancements aim to enhance the scalability and robustness of reinforcement learning in critical, real-world systems [26-28]. The average return of the last 10 evaluations is reported in Table 1.

Table 1. Average return of last 10 evaluations

Methods	SAC	EDM-SAC
BW	290.4±51.8	**292.8±80.2**
LL	281.6±4.8	**285.8±8.7**
Ant	3520.8±634.1	**4490.7±980.6**
Walker2d	3609.3±519.2	**3899.7±890.1**
Swimmer	48.7±2.2	**58.3±7.9**
Humanoid	4567.0±477.8	**4621.9±390.9**

6 Conclusion

This work proposes an adaptive experience replay (EDM) framework that accelerates agent learning through dynamic replay frequency adjustment while mitigating overfitting risks via a dual-policy interaction mechanism. Our method introduces two complementary policies: an exploratory policy (Policy No.1) that prioritizes rapid experience collection and a conservative policy (Policy No.2) that stabilizes optimization through low-frequency updates. By implementing reward-triggered policy switching, the framework achieves 15% higher cumulative rewards and 20% faster convergence than baseline methods like SAC in MuJoCo and Box2D benchmarks while reducing return variance by 30%.

References

[1] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, J. Pineau, An introduction to deep reinforcement learning, *Foundations and Trends® in Machine Learning*, Vol. 11, No. 3-4, pp. 219-354, December, 2018. https://doi.org/10.1561/2200000071

[2] P. D'Oro, M. Schwarzer, E. Nikishin, P. L. Bacon, M. G. Bellemare, A. Courville, Sample-efficient reinforcement learning by breaking the replay ratio barrier, *Deep Reinforcement Learning Workshop in Neural Information Processing Systems*, Kigali Rwanda, 2023, pp. 1-23. https://openreview.net/forum?id=OpC-9aBBVJe

[3] W. Fedus, P. Ramachandran, R. Agarwal, Y. Bengio, H. Larochelle, M. Rowland, W. Dabney, Revisiting fundamentals of experience replay, *International Conference on Machine Learning*, Virtual, 2020, pp. 3061-3071.

[4] A. Kumar, R. Agarwal, D. Ghosh, S. Levine, Implicit under-parameterization inhibits data-efficient deep reinforcement learning, *arXiv preprint*, arXiv:2010.14498, October, 2020. https://arxiv.org/abs/2010.14498

[5] X. Chen, C. Wang, Z. Zhou, K. Ross, Randomized ensembled double q-learning: Learning fast without a model, *arXiv preprint*, arXiv:2101.05982, January, 2021. https://arxiv.org/abs/2101.05982

[6] L. Smith, I. Kostrikov, S. Levine, A walk in the park: Learning to walk in 20 minutes with model-free reinforcement learning, *arXiv preprint*, arXiv:2208.07860, August, 2022. https://arxiv.org/abs/2208.07860

[7] E. Nikishin, M. Schwarzer, P. D'Oro, P. L. Bacon, A. Courville, The primacy bias in deep reinforcement learning, *International Conference on Machine Learning, Baltimore*,

Maryland, USA, 2022, pp. 16828-16847.

[8] E. Nikishin, J. Oh, G. Ostrovski, C. Lyle, R. Pascanu, W. Dabney, A. Barreto, Deep reinforcement learning with plasticity injection, *Advances in Neural Information Processing Systems*, New Orleans, LA, USA, 2024, pp. 37142-37159.

[9] C. Lyle, Z. Zheng, E. Nikishin, B. A. Pires, R. Pascanu, W. Dabney, Understanding plasticity in neural networks, *International Conference on Machine Learning*, Honolulu, Hawaii, USA, 2023, pp. 23190-23211.

[10] E. Todorov, T. Erez, Y. Tassa, Mujoco: A physics engine for model-based control, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura-Algarve, Portugal, 2012, pp. 5026-5033. https://doi.org/10.1109/IROS.2012.6386109

[11] I. Parberry, *Introduction to Game Physics with Box2D*, CRC Press, 2017.

[12] L. J. Lin, Self-improving reactive agents based on reinforcement learning, planning and teaching, *Machine Learning*, Vol. 8, No. 3-4, pp. 293-321, May, 1992. https://doi.org/10.1007/BF00992699

[13] A. Lazaridis, A. Fachantidis, I. Vlahavas, Deep reinforcement learning: A state-of-the-art walkthrough, *Journal of Artificial Intelligence Research*, Vol. 69, pp. 1421-1471, December, 2020. https://doi.org/10.1613/jair.1.12412

[14] R. S. Sutton, A. G. Barto, *Reinforcement learning: An introduction*, MIT Press, 2018.

[15] T. Schaul, J. Quan, I. Antonoglou, D. Silver, Prioritized experience replay, *arXiv preprint*, arXiv:1511.05952, November, 2015. https://arxiv.org/abs/1511.05952

[16] P. Sun, W. Zhou, H. Li, Attentive experience replay, *Proceedings of the AAAI Conference on Artificial Intelligence*, New York, USA, 2020, pp. 5900-5907.

[17] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, W. Zaremba, Hindsight experience replay, *Advances in Neural Information Processing Systems*, Long Beach, California, USA, 2017, pp. 5055-5065.

[18] C. Li, C. Gong, Q. He, X. Hou, Y. Liu, Centralized cooperative exploration policy for continuous control tasks, *arXiv preprint*, arXiv:2301.02375, January, 2023. https://arxiv.org/abs/2301.02375

[19] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, *International Conference on Machine Learning*, New York, NY, USA, 2016, pp. 1928-1937.

[20] J. Lim, Adaptive scheduling based on intelligent agents in edge-cloud computing environments, *Journal of Internet Technology*, Vol. 25, No. 4, pp. 609-617, July, 2024. https://doi.org/10.70003/160792642024072504011

[21] R. Bellman, Dynamic programming, *Science*, Vol. 153, No. 3731, pp. 34-37, July, 1966. https://doi.org/10.1126/science.153.3731.34

[22] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, *International Conference on Machine Learning*, Vancouver, BC, Canada, 2018, pp. 1861-1870. https://openreview.net/forum?id=HJjvxl-Cb

[23] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, W. Zaremba, OpenAI gym, *arXiv preprint*, arXiv:1606.01540, June, 2016. https://arxiv.org/abs/1606.01540

[24] B. Saglam, F. B. Mutlu, D. C. Cicek, S. S. Kozat, Actor prioritized experience replay, *Journal of Artificial Intelligence Research*, Vol. 78, pp. 639-672, November, 2023. https://doi.org/10.1613/jair.1.14819

[25] S. Fujimoto, H. Hoof, D. Meger, Addressing function approximation error in actor-critic methods, *International Conference on Machine Learning*, Stockholm, Sweden, 2018, pp. 1587-1596.

[26] L. P. Hung, W. Huang, S. T. Cheng, Z. J. Wu, S. O. Yang, Designing a multi-criteria decision-making framework to establish a value ranking system for the quality evaluation of long-term care services, *Journal of Internet Technology*, Vol. 25, No. 2, pp. 283-292, March, 2024. https://doi.org/10.53106/160792642024032502010

[27] Y. J. Su, S. Y. Yang, A user-friendly cloud-based multi-agent information system for smart energy-saving, *Journal of Internet Technology*, Vol. 25, No. 2, pp. 293-300, March, 2024. https://doi.org/10.53106/160792642024032502011

[28] Z. Yang, Y. Kong, C. H. Hsia, DERLight: a deep reinforcement learning traffic light control algorithm with dual experience replay, *Journal of Internet Technology*, Vol. 25, No. 1, pp. 79-86, January, 2024. https://doi.org/10.53106/160792642024012501007

Biographies



Yan Kong received the Ph.D. degree in Computer Science from the University of Wollongong, Australia. Currently, she works as a faculty in Nanjing University of Information, Science and Technology, China. Her research interests include deep learning, multi-agent system, and machine learning. Her research focuses on smart control on the traffic signal lights to alleviate traffic congestion.



Guoliang Che was born in Feb. 1998, is currently studying Software Engineering at the School of Software Engineering, Nanjing University of Information Science and Technology. His research direction is Deep Reinforcement Learning.



Chih-Hsien Hsia received the Ph.D. degree in Electrical and Computer Engineering from Tamkang University, Taiwan, and the second Ph.D. degree from National Cheng Kung University, Taiwan, in 2010 and 2023, respectively. He is currently a Distinguished Professor in the Department of Computer Science and Information Engineering at National Ilan University, Taiwan. His research interests include multimedia and DSP IC design. He has received multiple honors, including a Fellow of the International Association of Advanced

Materials in 2025, and receiving the outstanding young scholar award from the Taiwan Association of Systems Science and Engineering in 2020 and from the Computer Society of the Republic of China in 2018.