Foresight Distribution Adjustment for Off-policy Reinforcement Learning

Ruifeng Chen*
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University, Nanjing, China
chenrf@lamda.nju.edu.cn

Xu-Hui Liu*
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University, Nanjing, China
Polixir Technologies, Nanjing, China
liuxh@lamda.nju.edu.cn

Tian-Shuo Liu*
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University, Nanjing, China
Polixir Technologies, Nanjing, China
liuts@lamda.nju.edu.cn

Shengyi Jiang
The University of Hong Kong
Hong Kong, China
syjiang@cs.hku.hk

Feng Xu
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University, Nanjing, China
xufeng@lamda.nju.edu.cn

Yang Yu†
National Key Laboratory for Novel Software Technology
School of Artificial Intelligence
Nanjing University, Nanjing, China
Polixir Technologies, Nanjing, China
yuy@nju.edu.cn

ABSTRACT

Off-policy reinforcement learning algorithms maintain a replay buffer to utilize samples obtained from earlier policies. The sampling strategy that prioritizes certain data in a buffer to train the value function or the policy, has been shown to significantly influence the sample efficiency and the final performance of the algorithm. However, which distribution for the experience prioritization is the best choice has not been explored thoroughly. In this paper, we proved that the post-update policy distribution (i.e. the visitation distribution of the policy after the current iteration of update) is the best Q training distribution to benefit the policy improvement. Nevertheless, accessing this "future" distribution is not straightforward. In this work, we find that the current experiences can be modulated by the critic information to simulate the post-update policy distribution. Technically, we derive the gradient of the visitation distribution with respect to the policy parameter and obtain an explicit expression to approximate the post-update policy distribution. The derived method is named as **Fo**resight **D**istribution **A**djustment (FoDA), and seamlessly integrates with conventional off-policy actor-critic algorithms. Our experiments validate FoDA's ability to closely approximate the post-update policy distribution, and demonstrate its utility in enhancing performance across continuous control task benchmarks.

KEYWORDS

Reinforcement Learning; Off-policy Reinforcement Learning; Experience Replay

*Equal Contribution †Corresponding Author.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

ACM Reference Format:
Ruifeng Chen*, Xu-Hui Liu*, Tian-Shuo Liu*, Shengyi Jiang, Feng Xu, and Yang Yu†. 2024. Foresight Distribution Adjustment for Off-policy Reinforcement Learning. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

1 INTRODUCTION

Reinforcement learning [13, 14, 25, 27, 33, 36] has achieved impressive success in a wide range of sequential decision problems, e.g., sequential recommendation systems [43, 46], auto and robotic locomotion skill learning [9, 29]. Off-policy reinforcement learning algorithms [7, 9, 19] utilize a diverse set of experience data in the replay buffer collected by previous policies along the policy improvement path, which enables higher data efficiency compared with on-policy methods. Many previous works [16, 24, 31, 35] found that emphasizing important samples in the replay buffer can bring additional benefits. Therefore, one question arises: *Which samples are of greater importance to train the Q function?*

To tackle this problem, we revisit the *distribution shift* issue in off-policy reinforcement learning. The Q function is trained under the data distribution in the replay buffer, while the induced policy runs on its own visitation distribution, named as *post-update policy distribution*. We provide a new theoretical interpretation to understand how the distribution shift affects the learning efficiency, which shows that distribution shift is one of the sources of the objective mismatch between policy improvement and Q function training, and a smaller difference between Q function training distribution and post-update policy distribution leads to a less objective mismatch issue. This motivates us to use post-update policy distribution as Q training distribution to further reduce the mismatch.

However, it is difficult to obtain the post-update policy distribution directly, because the post-update policy has not interacted with the environment and thus its visitation distribution is not accessible. In fact, previous works also faced this problem [24, 32, 33].

To deal with this problem, they either limit the update of the policy [32, 33], or directly use the distribution of the current policy as a surrogate [24]. The two methods all hinder the efficiency of policy optimization. In contrast, we notice a basic fact that the upcoming policy update is guided by the current critic and the environment transition information contained in the current experiences can be reused to estimate visitation distribution of new policies. Based on this insight, we study the gradient of the visitation distribution with respect to the policy parameter and obtain an explicit expression to predict the post-update policy distribution. Then we propose a practical method to minimize the Bellman error under the predicted post-update policy distribution. We name this method **Fo**resight **D**istribution **A**djustment (FoDA), meaning to foresee the upcoming distribution change and adjust the training distribution accordingly.

We demonstrate the effectiveness of FoDA in post-update distribution prediction, verify that FoDA is able to approximate the real visitation distribution of the post-update policy, and showcase the performance and efficiency improvement on a suite of DeepMind Control [37], Gym MuJoCo [4, 38] and MetaWorld [44] tasks when combined with Soft Actor-Critic (SAC) [9], one of the most prevalent off-policy actor-critic methods. We also conduct a hyperparameter sensitivity test to show the robustness of FoDA.

The main contributions of this paper include:

(1) identifying the post-update policy distribution as a desirable priority distribution for the Q training to benefit the policy improvement (Section 4);
(2) deriving an expression of the post-update policy distribution estimate based on the investigation of distribution shift (Section 5.1);
(3) proposing a practical method FoDA for off-policy actor-critic algorithms (Section 5.2), and demonstrating its efficacy by experiments when combined with Soft Actor-Critic algorithm.

2 RELATED WORKS

Experience replay [20] has been extensively studied and shown to be a powerful technique to improve sample efficiency and enhance performance in off-policy RL algorithms. Recent researches mainly focus on non-uniform sampling strategies. In model-based planning, prioritized sweeping [2, 28, 39] was adopted to update the states with the largest Bellman error, which makes the planning process more efficient. Similarly, Schaul et al. [31] proposed Prioritized Experience Replay (PER) to assign priorities to transition samples based on the TD error in model-free value-based deep RL, which demonstrated significant learning efficiency improvements compared to the uniform sampling strategy. Prioritized sequence experience replay [3] extended PER to propagate the priority to previous transitions in the replay buffer. However, the simple TD-error-based sampling strategy does not help much in continuous-action control tasks and can even be harmful, unless more heuristic modifications and tricks are used [6, 17].

Another line of work noticed that the near on-policy data are more valuable than those collected by stale policies even in off-policy RL algorithms. Emphasizing Recent Experience (ERE) [42] used a hierarchical sampling strategy to sample data from recent policies more frequently. Likelihood-free Importance Weighting (LFIW) [35] maintained a small fast buffer and a large slow buffer, where the data in the fast buffer were regarded as near on-policy data, and estimated the importance weight between the fast buffer and the slow buffer distribution so as to adjust the data distribution toward on-policy. Liu et al. [24] provided an overall insight into the experience data distribution from the perspective of regret minimization, where one of the emphasized factors is exactly the data on-policiness. Besides, some researchers noticed the influence on Q-target accuracy and have proposed several heuristic solutions such as re-weighting transitions according to the cumulative bellman error [16, 24] and prioritizing the transitions near the end of trajectories [10, 18, 24]. Some other works [11, 12, 23, 26] focused on changing the buffer distribution on imitation learning setting, implying the effectiveness of distribution adjustment.

3 BACKGROUND

A Markov decision process (MDP) is defined by $(\mathcal{S}, \mathcal{A}, p, r, \gamma, \rho_0)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $p(\cdot|s,a)$ is the transition probability, $r(s,a) \in [0, R_{\max}]$ is the reward function, $\gamma \in (0,1)$ is the discount factor, and $\rho_0(s)$ is the initial state distribution. Reinforcement learning aims to learn a policy $\pi(a|s)$ to maximize the expected return $\eta(\pi) = \mathbb{E}_{\tau\sim P_\pi} \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$, where the expectation is taken under sequential data distribution generated by interaction between policy and environment.

Given a fixed policy π, we define the state visitation distribution at timestep t as $d_t^\pi(s) = P_\pi(s_t = s)$. Then the discounted state visitation distribution and the discounted state-action visitation distribution can be defined as $\rho^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t d_t^\pi(s)$ and $d^\pi(s,a) = \rho^\pi(s)\pi(a|s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t P_\pi(s_t = s, a_t = a)$ respectively. The discounted state-action visitation distribution is also known as occupancy measure. We use the standard definition of the state-action value function $Q^\pi(s,a) = \mathbb{E}_\pi\big[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)|s_0 = s, a_0 = a\big]$ and the state value function is defined as $V^\pi(s) = Q^\pi(s,\pi)$. The optimal Q function Q^* is the Q function of some policy π^* satisfying $Q^*(s,a) \geq Q^\pi(s,a)$ for all s, a, and the policy π^* is called an optimal policy.

Define the Bellman expectation operator $\mathcal{B}^\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ as:

$$(\mathcal{B}^\pi f)(s,a) := r(s,a) + \gamma \mathbb{E}_{s'\sim p(\cdot|s,a), a'\sim\pi(\cdot|s')} f(s', a'),$$

where $f \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$. Given the current Q estimate Q_k and policy π_k, off-policy RL algorithm usually updates the Q function by sample-based Bellman error minimization under some data distribution d:

$$Q_{k+1}(s,a) = \arg\min_Q \mathbb{E}_{(s,a)\sim d}\big[(Q(s,a) - \mathcal{B}^{\pi_k} Q_k(s,a))^2\big], \quad (1)$$

where d is often chosen to be the buffer data distribution under the uniform sampling strategy or some specified priority distribution. Then the Q function guides the policy update:

$$\pi_{k+1}(\cdot|s) = \arg\max_\pi \mathbb{E}_{a\sim\pi(\cdot|s)} Q_{k+1}(s,a). \quad (2)$$

4 DISTRIBUTION SHIFT IN REINFORCEMENT LEARNING

A fundamental issue that arises in sequential decision-making is *distribution shift*. In the k-th iteration, RL algorithm trains the Q

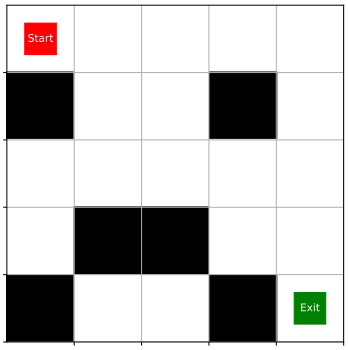


Figure 1: Visualization of the Maze MDP.

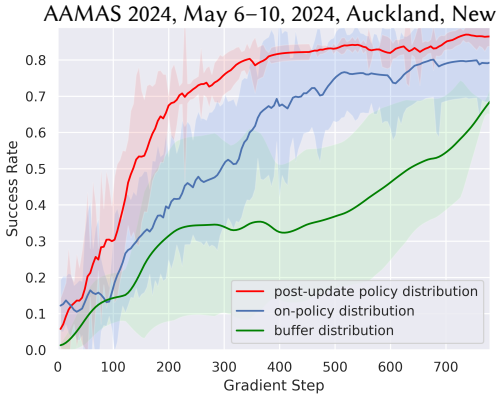


Figure 2: The learning curves for three Q learning distributions. Each curve is averaged for six seeds.

function Q_{k+1} from the collected experience subject to a data distribution d, and then updates the policy π_{k+1} under the guidance of Q_{k+1}. However, this new policy π_{k+1}, named as *post-update policy*, is evaluated on its state-action visitation distribution $d^{\pi_{k+1}}$, which is different from Q training data distribution d. This process can be described by the following chain:

$$d \xrightarrow{train} Q_{k+1} \xrightarrow{guide} \pi_{k+1} \xrightarrow{rollout} d^{\pi_{k+1}} \,\&\, \eta(\pi_{k+1}). \tag{3}$$

The performance of the post-update policy π_{k+1} is closely related to its optimality on $d^{\pi_{k+1}}$, which is further related to the Q_{k+1} estimate on this distribution. However, the Q function may not be well-trained in some important areas with high density $d^{\pi_{k+1}}$, because of the shifted probability mass from the training distribution d. Consequently, the Q estimate provides an unreliable signal for the policy update in these areas, which may impair the efficiency of policy improvement. Intuitively, this issue resembles the training-testing distribution gap in traditional supervised learning, although the mechanism here is more subtle and few researchers pay attention.

By default, common off-policy algorithms uniformly sample the data from a replay buffer D for Q training, which corresponds to the *buffer distribution* d_D. The data in the replay buffer are collected by many past policies $\pi_1, \pi_2, \ldots, \pi_k$, where the probability mass is distributed uniformly over a large number of mixed experiences and less focus on the important experiences among them. Recent works [24, 35] propose to fit the Q estimate under *on-policy distribution* d^{π_k} by reweighting the buffer data because on-policy distribution is visited by the current policy π_k. However, in the current iteration, we are optimizing the policy π_{k+1}, not π_k, which has been fixed after the previous iteration. Therefore, training the Q function under on-policy distribution cannot solve the issue. Ideally, if the Q function is trained under the visitation distribution of the post-update policy π_{k+1}, or *post-update policy distribution* $d^{\pi_{k+1}}$, the mismatch between the training and evaluation distribution seems to be diminished, and therefore the policy improvement get more reliable signals from the Q function learning.

4.1 Verification in Maze MDP

To verify this idea, we conduct an experiment in a Maze environment, which is visualized in Figure 1. With going up, down, left, and right as selectable actions, the agent starts at the upper left corner and the exit is at the lower right corner. The black blocks are occupied and inaccessible. The agent's goal is to reach the exit as quickly as possible, with every step the agent incurs a penalty or, when finally reaching the exit, a reward. Note that on-policy distribution and post-update policy distribution cannot be obtained directly in the training process. In this tabular environment, we can simulate the two distributions by rolling out π_k and π_{k+1} in this environment. The results of training on buffer distribution, on-policy distribution, and post-update policy distribution are shown in Figure 2.

Training Q function under the post-update policy distribution achieves the best efficiency compared to training under the on-policy distribution and the buffer distribution, which aligns with our intuition.

4.2 Theoretical Analysis

We try to formally explain the benefit of the post-update policy distribution. If the post-update policy π_{k+1} is obtained via the chain process (3), we have the following bound on the policy suboptimality basically from the policy difference lemma [15]:

Proposition 4.1. *Let π^* be the optimal policy, then the following bound holds:*

$$\eta(\pi^*) - \eta(\pi_{k+1}) \le \frac{2}{1-\gamma}\mathbb{E}_{d^{\pi_{k+1},\pi^*}}|Q^* - Q_{k+1}|(s,a) \tag{4}$$

$$\le \frac{2}{1-\gamma}\mathbb{E}_d|Q_{k+1} - \mathcal{B}^{\pi_k}Q_k|(s,a) + \frac{4L}{1-\gamma}W_1(d, d^{\pi_{k+1},\pi^*}) + \frac{2}{1-\gamma}\mathbb{E}_{d^{\pi_{k+1},\pi^*}}|Q^* - \mathcal{B}^{\pi_k}Q_k|(s,a), \tag{5}$$

where d^{π_{k+1},π^*} is a hybrid state-action distribution

$$d^{\pi_{k+1},\pi^*}(s,a) = \rho^{\pi_{k+1}}(s)\frac{\pi_{k+1}(a|s) + \pi^*(a|s)}{2},$$

L is an upper bound of the Lipschitz constant of Q^* and Q_{k+1} and $W_1(\cdot,\cdot)$ is Wasserstein-1 metric.

RL algorithm attempts to improve policy performance $\eta(\pi_{k+1})$ at each iteration, equivalent to reducing the policy suboptimality $\eta(\pi^*) - \eta(\pi_{k+1})$ compared to the optimal policy π^*, which therefore can be seen as the loss function of policy improvement. Equation (5) provides an upper bound of the policy suboptimality, wherein the first term is exactly the loss function of Q learning, that is, the expected Bellman error under the training distribution d. In general, minimizing an upper bound solves a surrogate of the original

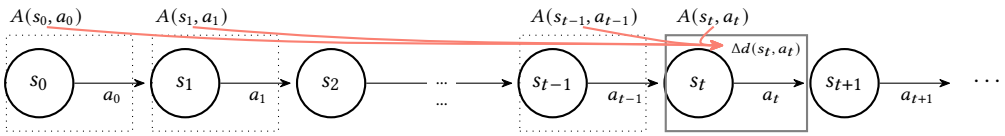


Figure 3: Illustration of the distribution shift estimation of (s_t, a_t). The estimation $\Delta(s_t, a_t)$ is proportional to the cumulative advantages of the state-action pairs leading to it.

minimization problem. Nevertheless, there remain extra terms besides Q learning loss in the upper bound (5), which result in an objective mismatch between the Q function training and the policy improvement.

We desire a better alignment of the two objectives so that the Q training could boost the policy improvement even more, which leads us to examine the source of the mismatch terms, that is, the last two terms in (5). The last term is an expectation of the difference between the optimal Q function Q^* and the Bellman target $\mathcal{B}^{\pi_k} Q_k$ at current iteration, which is induced by the bootstrapped regression nature of the TD learning inherently. The second term $\frac{4L}{1-\gamma} W_1(d, d^{\pi_{k+1},\pi^*})$ contains the distribution distance between the training distribution d and d^{π_{k+1},π^*}, which can be reduced by properly choosing the training distribution d. The state marginal distribution of d^{π_{k+1},π^*} is exactly the state visitation distribution of the post-update policy π_{k+1}, which coincides with our intuition. Therefore, post-update policy distribution $d^{\pi_{k+1}}$ is the best distribution we can choose because the π^* component in the action distribution is unknowable during the learning process and omitting it gives $d^{\pi_{k+1}}(s, a)$. This argument explains the efficiency rank shown in the Figure 2 because the buffer distribution d_D tends to be stale to result in a large mismatch and the on-policy distribution d^{π_k} gets closer, although still suboptimal compared to the post-update policy distribution. Therefore, we propose to train the Q function under the post-update policy distribution $d^{\pi_{k+1}}$.

5 FORESIGHT DISTRIBUTION ADJUSTMENT

In this section, we first frame the Q training under the post-update policy distribution as a constraint optimization problem. To solve this bi-level optimization, we need to estimate the Bellman residual expectation under the visitation distribution of the policy π_{k+1}, which has a closed form under the regularization assumption. However, it is impractical to directly obtain the exact post-update policy distribution, which requires additional environment interactions. To circumvent this issue, we develop an analysis of the visitation distribution shift, and propose to approximate the post-update policy distribution by reweighting the current experience distribution.

Choosing $d^{\pi_{k+1}}$ as the training distribution, the optimization objective of Q training in the current iteration becomes

$$\begin{aligned} &\min_{Q} \mathbb{E}_{s,a\sim d_{\pi'(Q)}} \left|Q - \mathcal{B}^{\pi_k} Q_k\right|^2(s, a) \\ &\text{s.t.} \quad \pi'(Q) = \arg\max_{\pi} \mathbb{E}_{a\sim\pi} Q(s, a) - \kappa R(\pi), \end{aligned} \tag{6}$$

where we add a regularization term $R(\pi)$ in the policy objective in order to introduce smoothness [1, 8, 40, 41]. To solve this bi-level problem, similar to the common approach in previous literature [22, 30, 34, 47], we perform an update for the inner loop to calculate the outer loop gradient:

$$\begin{aligned} \pi_{k+1} &= \arg\max_{\pi} \mathbb{E}_{a\sim\pi(\cdot|s)} Q_k(s, a) - \kappa R(\pi), \\ Q_{k+1} &= Q_k - \alpha \nabla_Q \mathbb{E}_{s,a\sim d^{\pi_{k+1}}} \left|Q(s, a) - \mathcal{B}^{\pi_k} Q_k(s, a)\right|^2. \end{aligned} \tag{7}$$

The remaining obstacle is that the post-update policy distribution $d^{\pi_{k+1}}$ cannot be obtained immediately even if we get the policy form. In the following, we try to solve this problem via the investigation of the distribution shift caused by the policy change.

5.1 Visitation Distribution Prediction of the Post-update Policy

The visitation distribution of the post-update policy π_{k+1} cannot be directly expressed as a function of the available quantities in the current iteration, because it involves the environment transition information for the actions selected by π_{k+1}, which is a black box to the agent and requires to be estimated by π_{k+1}'s interacting with the environment. However, the current experiences indeed contain much environment transition information, though not aligned with the post-update policy. Since the policy update is guided by the critic, a consequent idea is whether it is possible to use the Q information to modulate current experiences to simulate the post-update policy distribution. Our theoretical analysis in this subsection affirms this feasibility and provides a practical method in the next subsection.

The main idea is approximation by Taylor's expansion, i.e., using the first-order term of the visitation distribution to approximate the distribution shift caused by small policy changes. There exists a nontrivial derivation of the gradient of the visitation distribution with respect to the policy parameters, which has not been explored in previous works. We first calculate the gradient of t-th timestep state visitation:

Proposition 5.1. *For a policy π_θ parameterized by θ and its t-th timestep state visitation distribution $d_t^{\pi_\theta}(s)$,*

$$\nabla_\theta d_t^{\pi_\theta}(s) = d_t^{\pi_\theta}(s) \sum_{i=0}^{t-1} \sum_{\bar{s},\bar{a}} \nabla_\theta \log \pi_\theta(\bar{a}|\bar{s}) \mathbb{P}_{\pi_\theta}(s_i = \bar{s}, a_i = \bar{a}|s_t = s)$$

where $\mathbb{P}_{\pi_\theta}(s_i = \bar{s}, a_i = \bar{a}|s_t = s)$ is the i-th timestep state-action visitation distribution following policy π_θ conditioned on the given t-th timestep state s.

The core of derivation in Proposition 5.1 is a gradient recursion of timestep t, in reverse order to that in policy gradient theorem, detailed in Appendix A.2. Proposition 5.1 reveals that for a certain state s, if the policy change increases the probability on the state-action pairs $(\bar{s}, \bar{a})$ that are likely to transition to s in future steps, then the new policy tends to visit s more often than the old

Algorithm 1 Off-policy Actor Critic with Foresight Distribution Adjustment (FoDA)

Initialize Q networks $Q_\phi(s,a)$, a value network $V_\xi(s)$, a policy network π, a probability network ω_ψ, a slow replay buffer $\mathcal{D}_s$ and a fast buffer $\mathcal{D}_f$. The red color highlights the new component introduced by FoDA.
for episode $l = 1, 2, \ldots$ **do**
  for $t = 1$ **to** T **do**
    Collect experiences, store $\left(s_t, a_t, r_t, s_{t+1}, \sum_{j=0}^{t} r(s_j, a_j), s_0\right)$ to buffers $\mathcal{D}_s$ and $\mathcal{D}_f$.
    Update the probability network ω_ψ with samples from $\mathcal{D}_s, \mathcal{D}_f$ and LFIW discrimination loss function (detailed in Appendix B)
    Obtain samples $\{(s_i, a_i)\}_{i=1}^N$ from (slow) replay buffer $\mathcal{D}_s$.
    Compute the on-policy weight ω on samples (s_i, a_i) and the distribution adjustment term Δ by

$$\hat{\Delta}(s_t, a_t, t) = \sum_{j=0}^{t} r(s_j, a_j) + V(s_{t+1}) - V(s_0).$$

    Update the Q functions Q_ϕ under the adjusted distribution with loss

$$L_Q(\phi) = \sum_{i=1}^{N} \omega(s_i, a_i)(1 + \eta\hat{\Delta}(s_i, a_i, t_i))\Big(Q_\phi(s_i, a_i) - r(s_i, a_i) - \gamma Q_{\bar{\phi}}(s_{i+1}, \pi(s_{i+1}))\Big)^2.$$

    Update value network V_ξ with loss $L_V(\xi) = \sum_{i=1}^{N} \left(r(s_i, a_i) + \gamma V_\xi(s_{i+1}) - V_\xi(s_i)\right)^2$.
    Update the policy network π as in base algorithm.
  end for
end for

policy. This conclusion paves the way to simulate the new visitation distribution based on the current experiences.

We then consider the distribution shift caused by the policy update. If the regularization term $R(\pi)$ in policy update (7) is chosen to be $D_{\mathrm{KL}}(\pi, \pi_k)$, then the policy update has a closed-form expression

$$\pi_{k+1}(a|s) \propto \pi_k(a|s) \exp\left(\frac{1}{\kappa} Q_k(s,a)\right), \tag{8}$$

which is a classical form in regularized value iteration literature [40, 41], and introduces the smoothness between the policy updates. Combining Proposition 5.1 with this update form, we obtain the first-order approximation of the state-action visitation distribution for the post-update policy:

Theorem 5.2. *For the policy update (8), the state-action visitation distribution of the post-update policy $\pi_{\theta_{k+1}}$ can be expressed as*

$$\begin{aligned} d^{\pi_{\theta_{k+1}}}(s,a) &= d^{\pi_{\theta_k}}(s,a) + \\ &\eta \sum_{t=0}^{\infty} \gamma^t d_t^{\pi_{\theta_k}}(s,a) \mathbb{E}_\tau\left[\sum_{i=0}^{t} A_k(s_i, a_i) \Big| s_t = s, a_t = a\right] + o(\|\Delta\theta\|) \\ &= \sum_{t=0}^{\infty} \gamma^t d_t^{\pi_{\theta_k}}(s,a)\left(1 + \eta \mathbb{E}_\tau\left[\sum_{i=0}^{t} A_k(s_i, a_i) \Big| s_t = s, a_t = a\right]\right) + o(\|\Delta\theta\|) \end{aligned} \tag{9}$$

where the advantage function $A_k(s,a) = Q_k(s,a) - \mathbb{E}_{\tilde{a}\sim\pi_{\theta_k}} Q_k(s,\tilde{a})$, τ denotes trajectories $(s_0, a_0, s_1, \ldots)$, and $\eta = \frac{1-\gamma}{\kappa}$.

Theorem 5.2 shows that the state-action distribution of the post-update policy will shift towards areas containing trajectories with high accumulated advantages, as illustrated in Figure 3. The advantages provide optimization signals for the policy update. If every action in a sampled trajectory is promoted based on this signal, the visitation probability of the succeeding state will naturally increase propositional to the expected cumulative advantages.

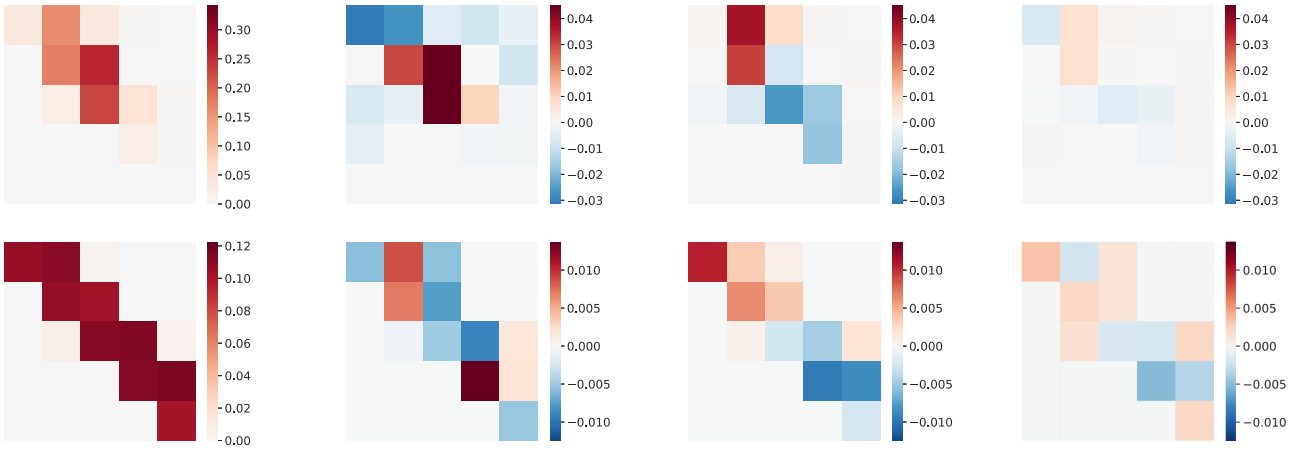
5.2 Foresight Distribution Adjustment (FoDA)

Theorem 5.2 allows predicting the post-update policy distribution based on the available quantities before the real update, and we can adjust the training distribution to approximate the predicted distribution, that is, **Fo**resight **D**istribution **A**djustment (FoDA). Based on the experiences from replay buffer, we can calculate the expected Bellman loss under the predicted distribution:

$$L_Q(\phi) = \mathbb{E}_{s,a,t\sim\mathcal{D}}\left[\omega(s,a)\Big(1 + \eta\Delta(s,a,t)\Big)\Big|Q_\phi(s,a) - \mathcal{B}^{\pi_k} Q_k(s,a)\Big|^2\right], \tag{10}$$

where $\mathcal{D}$ is the replay buffer distribution. There are three unfamiliar values in Equation (10), whose meanings and computation are elaborated below.

- $\Delta(s,a,t) = \mathbb{E}_\tau\left[\sum_{i=0}^{t} A_k(s_i, a_i) \Big| s_t = s, a_t = a\right]$ is the expected cumulative advantages, which can be estimated by sampled trajectories. In principle, we could sample a (s,a) pair from the replay buffer, along with the corresponding collected trajectory τ containing the (s,a) pair and its timestep t. By summing up the advantages of state-actions on this trajectory before timestep t, we obtain a Δ estimate. In practice, however, this requires sampling the whole trajectory each time, which is time-consuming. To address this problem, we propose a practical approach wherein a value function V is learned separately. We utilize the advantage estimate $A(s,a) = r(s,a) + V(s') - V(s)$, and therefore the estimate $\tilde{\Delta}(s_t, a_t, t) = \sum_{j=0}^{t} r(s_j, a_j) + V(s_{t+1}) - V(s_0)$ only relies on cumulative rewards and the initial state s_0.



(a) State visitation (b) Buffer distribution difference (c) On-policy Distribution difference (d) FoDA Distribution difference

Figure 4: Illustration of the distribution differences between the actual post-update policy distribution and the three training distributions in a Maze environment. The overall block colors indicate that FoDA provides the best approximation to the post-update policy distribution, resulting in the smallest mismatch.

These values can be stored alongside (s, a) sample during the data collection process, eliminating the need to sample entire trajectories during training.

- The coefficient η denotes $\frac{1-\gamma}{\kappa}$ as in Theorem 5.2, where κ is the unspecified regularization coefficient in the policy pseudo-update (7). Therefore, we treat the ratio $\eta = \frac{1-\gamma}{\kappa}$ as a tuneable parameter. Similar to [5], we set $\eta = \frac{\eta_0}{\sum_{i=1}^{N} \Delta(s_i, a_i, t_i)}$ to maintain a stable relative magnitude of the second term in Equation (10) by in-batch normalization, where η_0 is a hyperparameter and N is the batch size. This in-batch normalization keeps a stable gradient magnitude to avoid potential numerical explosion caused by the estimated error and sample variance.
- On-policy ratio $\omega(s, a)$. This quantity is the importance ratio between buffer distribution and on-policy distribution $d^{\pi_{\theta_k}}$, and thus with this ratio we can simulate $d^{\pi_{\theta_k}}$ by simply sampling from the replay buffer. LFIW [35] provides a method to estimate this value. It introduces an additional small fast buffer $\mathcal{D}_f$ that contains the experience collected by recent policies, uses the conventional large replay buffer as a slow buffer $\mathcal{D}_s$, trains a neural network to discriminate the two buffer distributions, and $\omega(s, a)$ is the value output by the discriminator (see details in Appendix B).

It is noteworthy that although we use the LFIW to reweight the buffer data in our implementation, it can be replaced by other methods such as DICE [21, 45]. Therefore, our method is not bound to one particular on-policy reweighting technique in principle.

The overall algorithm procedure is shown in Algorithm 1, where the red color highlights the new component introduced by FoDA.

6 EXPERIMENTS

In this section, we present experimental evaluation to answer the following questions:

- How well does our method approximate the post-update policy distribution?
- Can the foresight distribution adjustment improve the performance compared to other experience replay techniques?
- How does the hyperparameter value affect the performance?

We start with experiments on a Maze MDP to visualize the distribution difference of buffer distribution, on-policy distribution, and the adjusted distribution via FoDA compared with the post-update policy distribution respectively. Then we use Soft Actor-Critic (SAC) algorithm as the base algorithm to evaluate FoDA on several tasks from DeepMind Control suite [37] and Gym MuJoCo [4, 38] environments, and robotic manipulation tasks from Meta-World benchmark [44]. These benchmarks are representative of complex large-scale continuous control tasks, which can verify the effectiveness of FoDA in challenging realistic problems. Finally, we analyze the sensitivity of the introduced new hyperparameters to test the robustness of FoDA.

6.1 Effectiveness of Foresight Distribution Adjustment on a Maze MDP

In order to examine the effectiveness of the distribution prediction for the post-update policy inspired by Theorem 5.2, we visualize the empirical state visitation distribution and the differences between the post-update policy distribution and the three training distribution choices, that is, buffer distribution, on-policy distribution, and the predicted distribution by FoDA. The experiment is conducted on the Maze environment as shown in Figure 1.

We plot two intermediate iterations during the training process in Figure 4, where the visitation frequency is calculated by counting. The total number of training iterations is 100. The first row of Figure 4 is from an early training stage, after 20 iterations, when the agent begins to explore the environment but has not reached the exit, and the policy change is relatively significant at each step. The

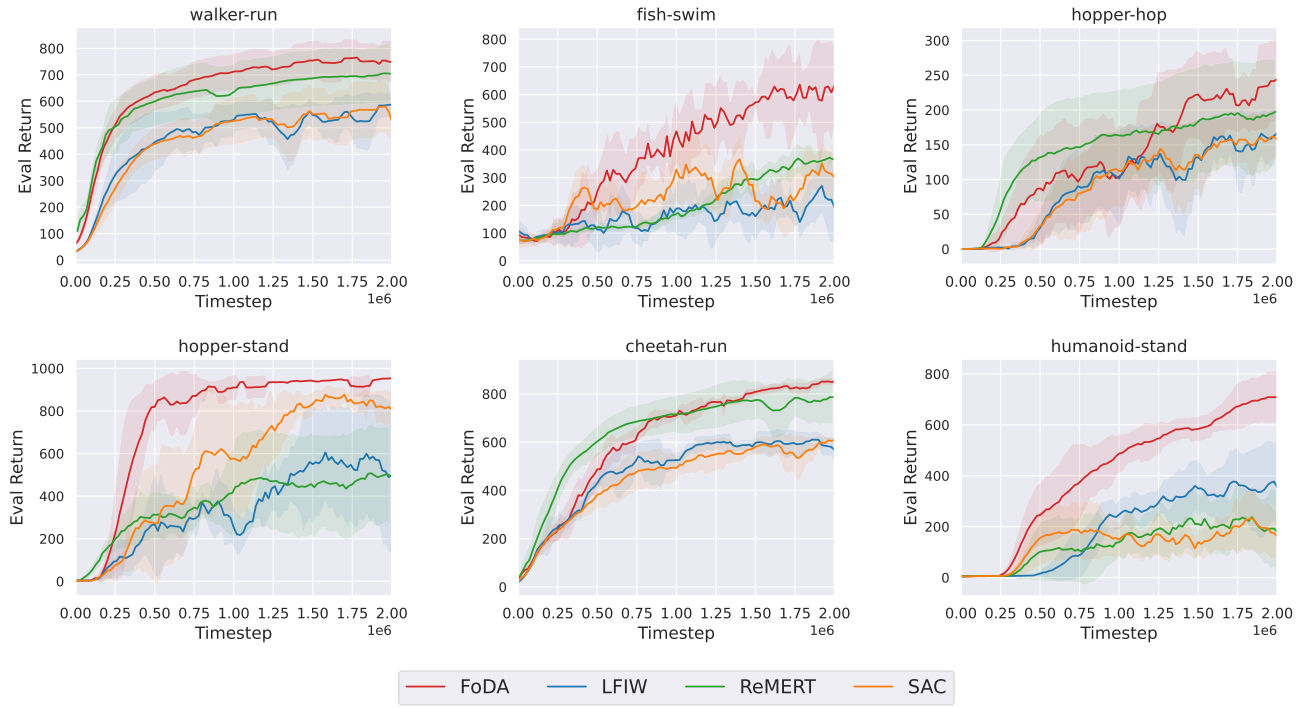


Figure 5: Learning curves of FoDA and three baselines on six DeepMind Control tasks. Solid curves depict the mean of five trials and shaded regions correspond to the standard deviation among trials.

second row comes from a later training stage, after 80 iterations of training, where the visitation probability is concentrated on a path and the policy change is moderate. We chose these two iterations because they are representative of different phases during the learning process. As shown in Figure 4(a), the state visitation frequency of the first stage is dispersed, where exploration is dominated, while for the second stage, the frequency is concentrated, where exploitation is dominant.

In Figure 4(b), 4(c) and 4(d), we plot the difference between the buffer distribution, on-policy distribution, and the adjusted distribution via FoDA and the empirical state visitation distributions of the policy at the next step (i.e., the ground-truth post-update policy distributions) respectively. Dark-colored blocks indicate significant differences. In Figure 4(b), we can find some extremely dark colors, which means the distribution shift is significant under the buffer distribution. In Figure 4(c), the dark colors are faded though there still exist some non-negligible differences. This demonstrates that on-policy distribution can indeed mitigate the distribution shift issue to some extent but cannot solve it. Figure 4(d) shows most of the blocks become light-colored, which verifies that FoDA solves the distribution shift issue except for negligible error. If measured in total variation, the distances in Figure 4(d) are reduced by 75%, 48.6% compared to those in Figure 4(c), and 96%, 87.3% compared to those in Figure 4(b) respectively.

6.2 Performance on Continuous Control Environments

We evaluate FoDA based on the SAC algorithm on different tasks from DeepMind Control (DMC) suite, Gym MuJoCo environments, and also robotic manipulation tasks from MetaWorld benchmark. Our method is compared to the uniform experience replay and the past state-of-the-art prioritization methods, LFIW [35] and ReMERN/T [24], where Q learning is based on on-policy distribution. ReMERT is used in DMC and MuJoCo tasks, while ReMERN is used in MetaWorld tasks according to the algorithm selection criterion [24]. We run a total of 2M timesteps for DMC and MuJoCo tasks and 1.5M timesteps for MetaWorld tasks. The learning curves of DMC and MuJoCo tasks are shown in Figure 5 and Figure 6, while the results for the MetaWorld tasks are shown in Figure A1 in Appendix C.1. The cumulative return for DMC and MuJoCo and the success rate for MetaWorld are the performance metrics.

The results shown in Figure 5 exhibit significant performance and efficiency improvement of FoDA compared with vanilla SAC and two prioritized methods in six DMC tasks. For MuJoCo tasks shown in Figure 6, prior methods are already able to achieve a good performance, while FoDA still gains significant improvement in some tasks. The stochastic MetaWorld environments raise a challenge for the baseline algorithms while FoDA reliably completes the tasks with a higher efficiency as shown in Figure A1. Notably, FoDA incorporates the on-policy importance weight from LFIW,

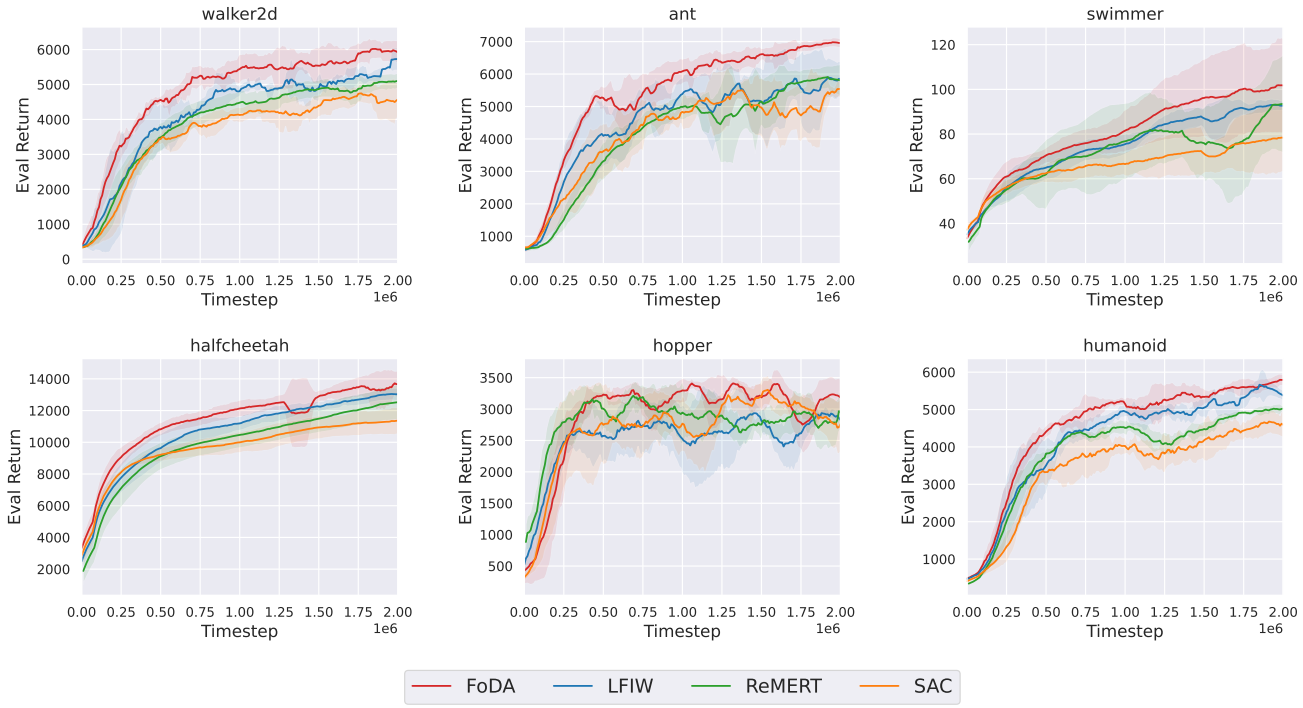


Figure 6: Learning curves of FoDA and three baselines on Gym MuJoCo tasks. Solid curves depict the mean of five trials and shaded regions correspond to the standard deviation among trials.

making the comparison between LFIW and FoDA an ablation study. The superior performance of FoDA demonstrates the effectiveness of our novel distribution adjustment term.

6.3 Hyperparameter Sensitivity

In our method, there are three new hyperparameters compared to the base algorithm SAC. Two of them, i.e., the fast buffer size and the on-policy weight temperature, are introduced by LFIW, and the algorithm is robust to the selection of the two parameters [35]. FoDA introduces a new hyperparameter, the adjustment coefficient η_0. We test the robustness of our method to the value of η_0 in two DMC environments and show the results in Figure 7. We use $\eta_0 = 10$ for the DMC tasks in the previous performance comparison, and we find that modifying the value of η_0 in a reasonable range, from 5 to 15 in Figure 7, will not cause significant performance degradation, which demonstrates that our method FoDA is relatively robust to the introduced hyperparameter.

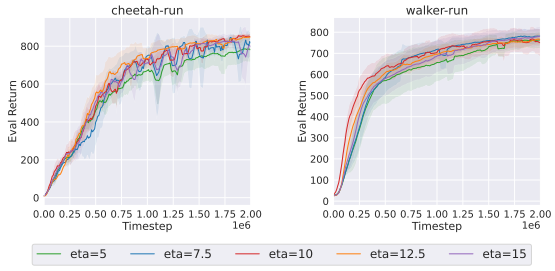


Figure 7: Sensitivity analysis for the value of adjustment coefficient η_0. Changing the value of η_0 within a reasonable range (5 to 15 in the picture) only slightly hurts the performance.

7 CONCLUSION

We find that the distribution shift between the Q training distribution and the visitation distribution of the post-update policy hinders more efficient policy optimization. To explain this phenomenon, we theoretically show that such a distribution shift exacerbates the objective mismatch between the Q function training and the policy improvement, and propose to train the Q function under the post-update policy distribution. To approximate the post-update policy distribution, we propose a novel method Foresight Distribution Adjustment (FoDA), and verify its effectiveness in simple environments. When combined with SAC, we evaluate the new algorithm in challenging continuous control tasks. Compared to other prioritization methods, FoDA exhibits superior performance in many tasks, demonstrating FoDA helps improve the sample efficiency of off-policy actor-critic algorithms. It should be noted that we only study the relatively small distribution shift in the online RL setting. In the offline RL setting, the distribution mismatch is much more challenging, making experience prioritization or data selection even more important. Although some work has been done in this area, offline data selection remains a promising direction for future research.

ACKNOWLEDGMENTS

This work is supported by National Science Foundation of China (61921006). We thank the anonymous reviewers for their support and helpful discussions on improving the paper.

REFERENCES

[1] Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. 2019. Understanding the impact of entropy on policy optimization. In *Int'l Conf. on machine learning*. PMLR, 151–160.

[2] David Andre, Nir Friedman, and Ronald Parr. 1997. Generalized Prioritized Sweeping. In *Proceedings of the 10th Conf. on Neural Information Processing Systems (NeurIPS'97)*. Denver, CO.

[3] Marc Brittain, Joshua R. Bertram, Xuxi Yang, and Peng Wei. 2019. Prioritized Sequence Experience Replay. *CoRR* abs/1905.12726 (2019).

[4] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. *CoRR* abs/1606.01540 (2016).

[5] Scott Fujimoto and Shixiang Shane Gu. 2021. A Minimalist Approach to Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems 34 (NeurIPS'21)*. Virtual Event.

[6] Scott Fujimoto, David Meger, and Doina Precup. 2020. An Equivalence between Loss Functions and Non-Uniform Sampling in Experience Replay. In *Proceedings of the 33rd Conf. on Neural Information Processing Systems (NeurIPS'20)*. Virtual Event.

[7] Scott Fujimoto, Herke van Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th Int'l Conf. on Machine Learning (ICML'18)*. Stockholmsmässan, Sweden.

[8] Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. 2019. A theory of regularized markov decision processes. In *Int'l Conf. on Machine Learning (ICML'19)*.

[9] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th Int'l Conf. on Machine Learning (ICML'18)*. Stockholmsmässan, Sweden.

[10] Zhang-Wei Hong, Tao Chen, Yen-Chen Lin, Joni Pajarinen, and Pulkit Agrawal. 2021. Topological Experience Replay. In *Int'l Conf. on Learning Representations (ICLR'21)*.

[11] Ryan Hoque, Ashwin Balakrishna, Ellen R. Novoseller, Albert Wilcox, Daniel S. Brown, and Ken Goldberg. 2021. ThriftyDAgger: Budget-Aware Novelty and Risk Gating for Interactive Imitation Learning. In *Proceedings of the 5th Conf. on Robot Learning (CoRL'21)*. London, UK.

[12] Ryan Hoque, Ashwin Balakrishna, Carl Putterman, Michael Luo, Daniel S. Brown, Daniel Seita, Brijen Thananjeyan, Ellen R. Novoseller, and Ken Goldberg. 2021. LazyDAgger: Reducing Context Switching in Interactive Imitation Learning. In *Proceedings of the 17th Int'l Conf. on Automation Science and Engineering (CASE'21)*. Lyon, France.

[13] Chengxing Jia, Fuxiang Zhang, Tian Xu, Jing-Cheng Pang, Zongzhang Zhang, and Yang Yu. 2023. Model gradient: unified model and policy learning in model-based reinforcement learning. In *Frontiers of Computer Science*. 18:184339.

[14] Xue-Kun Jin, Xu-Hui Liu, Shengyi Jiang, and Yang Yu. 2022. Hybrid Value Estimation for Off-policy Evaluation and Offline Reinforcement Learning. *CoRR* abs/2206.02000 (2022).

[15] Sham M. Kakade and John Langford. 2002. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of the 19th Int'l Conf. on Machine Learning (ICML'02)*. Sydney, Australia.

[16] Aviral Kumar, Abhishek Gupta, and Sergey Levine. 2020. DisCor: Corrective Feedback in Reinforcement Learning via Distribution Correction. In *Proceedings of 33rd Conf. on Neural Information Processing Systems (NeurIPS'20)*.

[17] Sanghwa Lee, Jaeyoung Lee, and Ichiro Hasuo. 2021. Predictive PER: balancing priority and diversity towards stable deep reinforcement learning. In *2021 Int'l Joint Conf. on Neural Networks (IJCNN'21)*. IEEE, 1–10.

[18] Su Young Lee, Choi Sungik, and Sae-Young Chung. 2019. Sample-efficient deep reinforcement learning via episodic backward update. In *Proceedings of 32nd Conf. on Neural Information Processing Systems (NeurIPS'19)*.

[19] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *Proceedings of the 4th Int'l Conf. on Learning Representations (ICLR'16)*. San Juan, Puerto Rico.

[20] Long Ji Lin. 1992. Self-Improving Reactive Agents Based On Reinforcement Learning, Planning and Teaching. *Journal of Machine Learning Research* 8 (1992), 293–321.

[21] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. 2018. Breaking the Curse of Horizon: Infinite-Horizon Off-Policy Estimation. In *Proceedings of the 31st Neural Information Processing Systems (NeurIPS'18)*. Montréal, Canada.

[22] Runze Liu, Fengshuo Bai, Yali Du, and Yaodong Yang. 2022. Meta-reward-net: Implicitly differentiable reward learning for preference-based reinforcement learning. *Advances in Neural Information Processing Systems* 35 (2022).

[23] Xu-Hui Liu, Feng Xu, Xinyu Zhang, Tianyuan Liu, Shengyi Jiang, Ruifeng Chen, Zongzhang Zhang, and Yang Yu. 2023. How To Guide Your Learner: Imitation Learning with Active Adaptive Expert Involvement. *CoRR* abs/2303.02073 (2023).

[24] Xu-Hui Liu, Zhenghai Xue, Jingcheng Pang, Shengyi Jiang, Feng Xu, and Yang Yu. 2021. Regret Minimization Experience Replay in Off-Policy Reinforcement Learning. In *Proceedings of 34th Conf. on Neural Information Processing Systems (NeurIPS'21)*.

[25] Fan-Ming Luo, Tian Xu, Hang Lai, Xiong-Hui Chen, Weinan Zhang, and Yang Yu. 2024. A survey on model-based reinforcement learning. In *SCIENCE CHINA Information Sciences*. 67(2): 121101.

[26] Kunal Menda, Katherine Rose Driggs-Campbell, and Mykel J. Kochenderfer. 2019. EnsembleDAgger: A Bayesian Approach to Safe Imitation Learning. In *Proceedings of Int'l Conf. on Intelligent Robots and Systems (IROS'19)*. Macau, China.

[27] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[28] Andrew W. Moore and Christopher G. Atkeson. 1993. Prioritized Sweeping: Reinforcement Learning With Less Data and Less Time. *Journal of Machine Learning Research* 13 (1993).

[29] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Edward Lee, Jie Tan, and Sergey Levine. 2020. Learning Agile Robotic Locomotion Skills by Imitating Animals. In *Proceedings of the 14th Robotics: Science and Systems (RSS'20)*. Virtual Event.

[30] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. 2019. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems (NeurIPS'19)*, Vol. 32. Vancouver, BC, Canada.

[31] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2016. Prioritized Experience Replay. In *Proceedings of the 4th Int'l Conf. on Learning Representations (ICLR'16)*. San Juan, Puerto Rico.

[32] John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd Int'l Conf. on Machine Learning (ICML'15)*. Lille, France, 1889–1897.

[33] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017).

[34] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems* 32 (2019).

[35] Samarth Sinha, Jiaming Song, Animesh Garg, and Stefano Ermon. 2022. Experience replay with likelihood-free importance weights. In *Learning for Dynamics and Control Conf. (L4RC'22)*. Stanford, USA.

[36] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[37] Yuval Tassa, Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, and Nicolas Heess. 2020. dm-control: Software and Tasks for Continuous Control. *CoRR* abs/2006.12983 (2020).

[38] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. MuJoCo: A physics engine for model-based control. In *Proceedings of 24th Int'l Conf. on Intelligent Robots and Systems (IROS'12)*. Vilamoura, Portugal.

[39] Harm van Seijen and Richard S. Sutton. 2013. Planning by Prioritized Sweeping with Small Backups. In *Proceedings of the 30th Int'l Conf. on Machine Learning (ICML'13)*. Atlanta, USA.

[40] Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, and Matthieu Geist. 2020. Leverage the average: an analysis of KL regularization in reinforcement learning. In *Proceedings of the 33rd Conf. on Neural Information Processing Systems (NeurIPS'20)*.

[41] Nino Vieillard, Olivier Pietquin, and Matthieu Geist. 2020. Munchausen reinforcement learning. In *Proceedings of the 33rd Conf. on Neural Information Processing Systems (NeurIPS'20)*.

[42] Che Wang, Yanqiu Wu, Quan Vuong, and Keith Ross. 2020. Striving for Simplicity and Performance in Off-Policy DRL: Output Normalization and Non-Uniform Sampling. In *Proceedings of the 37th Int'l Conf. on Machine Learning (ICML'20)*. Virtual Event.

[43] Xiting Wang, Yiru Chen, Jie Yang, Le Wu, Zhengtao Wu, and Xing Xie. 2018. A Reinforcement Learning Framework for Explainable Recommendation. In *Proceedings of the 18th Int'l Conf. on Data Mining (ICDM'18)*. Singapore.

[44] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. 2019. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning. In *Proceedings of the 3rd Conf. on Robot Learning (CoRL'19)*. Osaka, Japan.

[45] Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. 2020. GenDICE: Generalized Offline Estimation of Stationary Values. In *Proceedings of the 8th Int'l Conf. on Learning Representations (ICLR'20)*. Addis Ababa, Ethiopia.

[46] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning. In *Proceedings of the 24th Int'l Conf. on Knowledge Discovery & Data Mining (KDD'18)*. London, UK.

[47] Zeyu Zheng, Junhyuk Oh, and Satinder Singh. 2018. On learning intrinsic rewards for policy gradient methods. *Advances in Neural Information Processing Systems* 31 (2018).