Off-policy Reinforcement Learning with Optimistic Exploration and Distribution Correction

Anonymous Author(s)
Affiliation
Address
email

Abstract

Improving the sample efficiency of reinforcement learning algorithms requires effective exploration. Following the principle of *optimism in the face of uncertainty* (OFU), we train a separate exploration policy to maximize the approximate upper confidence bound of the critics in an off-policy actor-critic framework. However, this introduces extra differences between the replay buffer and the target policy regarding their stationary state-action distributions. To mitigate the off-policy-ness, we adapt the recently introduced DICE framework to learn a distribution correction ratio for off-policy RL training. In particular, we correct the training distribution for both policies and critics. Empirically, we evaluate our proposed method in several challenging continuous control tasks and show superior performance compared to state-of-the-art methods. We also conduct extensive ablation studies to demonstrate the effectiveness and rationality of the proposed method.

1 Introduction

Incorporating high capacity function approximators (e.g., neural network) with actor-critic algorithms has shown great promise in many areas [1, 2, 3, 4, 5]. However, its training instability and poor sample efficiency hinder its wider deployment in the real world. While these issues have been studied under different settings, one of the most powerful and practical frameworks is the off-policy actor-critic algorithm [6, 7, 8, 9], where a replay buffer is maintained to store previously collected transitions, and both the policy and the critic are trained by sampling transitions from it. There are two major obstacles that a successful off-policy actor-critic algorithm has to overcome. Firstly, a carefully-designed exploration strategy is required to facilitate the efficient learning of the target policy. Secondly, off-policy training is unstable when combing with function approximation and bootstrapping, known as the *deadly triad* [10, 11].

To stabilize training, state-of-the-art (SOTA) methods, e.g., SAC [7] and TD3 [8], use the minimum value between a pair of critics as an approximate lower bound to alleviate the overestimation of Q-values. To facilitate exploration, TD3 injects Gaussian white noise to the deterministic action while SAC explicitly trains its policy to have a target entropy depending on the dimensionality of the action space [7]. Given the success of the OFU principle [12, 13], OAC proposes to maintain a separate exploration policy by maximizing an approximate upper bound of the critics and empirically showing that such an exploration strategy leads to more efficient exploration and, therefore, better sample efficiency of policy learning. To limit the amount of *off-policyness* and stabilize training, OAC further constrain the KL divergence between the exploration and target policies.

While the intuition of OAC to leverage an “optimistic" exploration policy could be beneficial for efficient exploration, we argue that an important aspect of off-policy training has not been taken into account in all the above methods, i.e., a faithful policy gradient computation requires samples from the on-policy distribution which is in general not satisfied by directly sampling from the relay buffer

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

in off-policy training. This issue becomes even more severe when a separate exploration policy is adopted. As a trade-off, OAC has to impose a KL constraint between the exploration and the target policy, which, on the one hand, limits the potential of deploying a more effective exploration policy that executes informative actions that are spuriously underestimated by the target policy and on the other hand, still suffers from the distribution shift during off-policy training of the target policy.

In this work, we propose directly applying a distribution correction for off-policy actor-critic algorithms where a separate optimistic exploration policy can be trained solely on executing the most informative actions w.r.t. current target policy without worrying about its "dissimilarity" to the target policy. Estimating the distribution correction ratio, however, is non-trivial. Traditional methods based on importance sampling suffer from high variance. The recently proposed Distribution Correction Estimation (DICE) family [14, 15, 16, 17, 18, 19] provides us with a way to estimate the distribution correction ratio in a low-variance manner by solving a linear program (LP). However, these methods are designed for the *off-policy evaluation* (OPE) task, where both the target policy and replay buffer are fixed. It is unclear how to incorporate them into the more sophisticated off-policy RL.

In this work, we successfully integrate the DICE optimization scheme into the off-policy actor-critic framework and stabilize the DICE-related training process by modifying the original optimization problem. We theoretically show that the learned correction ratio can be applied to correct the distributions of the policy gradient estimator. Compared with OAC, our distribution-corrected training procedure enables learning a more effective exploration policy without enforcing any distance constraint between the exploration and the target policy. Empirically, we show that our methods achieve a higher sample efficiency in challenging continuous control tasks than SOTA off-policy RL algorithms, demonstrating the power of the proposed distribution-corrected off-policy actor-critic framework in allowing for a more effective exploration strategy design. Moreover, we examine the quality of our learned correction ratio by using it to construct the dual estimator proposed in [19] to estimate the average per-step rewards (on-policy rewards) of the RL target policy. We highlight that this is harder than the standard OPE setting, as both our target policy and replay buffer are changing during RL training. Surprisingly, the dual estimator gives a decent estimation by matching the on-policy rewards, demonstrating the effectiveness of our distribution correction scheme.

To the best of our knowledge, this is the first work to improve the exploration efficiency of RL algorithms by explicitly correcting the training distribution rather than constraining the variation of the exploration strategy w.r.t. the target policy, which opens up a direction of potentially more profound exploration strategy design.

2 Preliminaries

2.1 Reinforcement Learning

We consider the standard Reinforcement Learning (RL) setup that maximizes the expected discounted cumulative rewards in an infinite horizon Markov Decision Process (MDP). Formally, a MDP is defined by a tuple ($\mathcal{S}$, $\mathcal{A}$, R, T, γ), with state space $\mathcal{S}$, action space $\mathcal{A}$, reward function $R(s,a)$, transition function $T(s'|s,a)$, and discount factor $\gamma \in (0,1)$. At timestep t, the agent selects an action $a_t \sim \pi(\cdot|s_t)$ for current state s_t by policy $\pi(\cdot|s_t)$, receives a reward $r_t := R(s_t,a_t)$ and transits to the next state $s_{t+1} \sim T(\cdot|s_t,a_t)$. We define the action value function of policy π as $Q^\pi(s,a) = \mathbb{E}_{\tau\sim\pi:s_0=s,a_0=a}[\Sigma_{t=0}^{\infty}\gamma^t r_t]$, where $\tau = (s_0,a_0,r_0,s_1,a_1,r_1,\ldots)$ is a trajectory generated by rolling out in the environment following π. Thus, the standard RL objective can be expressed as $J = \mathbb{E}_{s_0\sim\rho_0(s),a\in\pi}[Q^\pi(s_0,a)]$, where ρ_0 is the initial state distribution. We further denote the discounted marginal state distribution of π as $d^\pi(s)$ and the corresponding state-action marginal as $d^\pi(s,a) = d^\pi(s)\pi(a|s)$.

2.2 Soft Actor-Critic (SAC)

To encourage exploration, SAC[7] instead optimizes the maximum entropy objective $J = \mathbb{E}_{\tau\sim\pi:s_0\sim\rho_0(s)}[\Sigma_{t=0}^{\infty}\gamma^t(r_t + \alpha\mathcal{H}(\cdot|\pi(s_t))]$, where $\mathcal{H}$ is the entropy function and $\alpha \in \mathbb{R}^+$. The resulting target policy π has higher entropy, thus generating diverse actions during rollout.

SAC learns the Q value function as critics from the collected experience and uses it to guide the policy (actor) update. To stabilize training, SAC estimates lower confidence bound of the

true Q value by maintaining a pair of Q networks, $\hat{Q}_1$ and $\hat{Q}_2$, and taking their minimum, i.e., $\hat{Q}_{\mathrm{LB}}(s,a) = \min_{i\in\{1,2\}} \hat{Q}_i(s,a)$. The two Q networks are structured identically and trained in the same way but initialized differently. The Q value function is trained via value-based methods. The objective function minimizes the expected single-step *bellman error*. It further keeps $\hat{Q}_1'$ and $\hat{Q}_2'$ as the exponential moving average of $\hat{Q}_1$ and $\hat{Q}_2$ to provide a stable training target. To this end, the target Q value is computed by

$$\mathcal{B}^{\pi}\hat{Q}_{\mathrm{LB}}(s,a) = r - \alpha\gamma \log \pi(\cdot|s') + \gamma \min_{i\in\{1,2\}} \hat{Q}_i'\left(s',a'\right),$$

where $a' \sim \pi\left(\cdot \mid s'\right)$, and the training loss for Q functions are

$$L_Q^i = \mathop{\mathbb{E}}_{(s,a,r,s')\sim D} \left(\hat{Q}_i(s,a) - \mathcal{B}^{\pi}\hat{Q}_{\mathrm{LB}}(s,a)\right)^2, i \in \{1,2\}. \tag{1}$$

As for the actor, SAC reparameterizes the stochastic policy $\pi(\cdot|s)$ as $f_\theta(s,\varepsilon)$, where $\varepsilon \sim \mathcal{N}(0,I)$, and derives the policy gradient w.r.t its parameter θ by leveraging the reparameterization trick [20], which is given by

$$\nabla_\theta J^\pi = \int_s d^\pi(s)\left(\int_\varepsilon \nabla_\theta \hat{Q}_{\mathrm{LB}}\left(s, f_\theta(s,\varepsilon)\right)\phi(\varepsilon)d\varepsilon + \alpha \int_\varepsilon -\nabla_\theta \log f_\theta(s,\varepsilon)\phi(\varepsilon)d\varepsilon\right)ds, \tag{2}$$

where $\phi(\varepsilon)$ denotes the density function of ε. As we do not have access to the ground truth $d^\pi(s)$ during training, SAC approximates $d^\pi(s)$ with the state distribution $d^{\mathcal{D}}(s)$ of the replay buffer, and applies the following Monte-Carlo estimator of (2):

$$\nabla_\theta J^\pi \approx \sum_{s\in\mathcal{D}} \nabla_\theta \hat{Q}_{\mathrm{LB}}\left(s, f_\theta\left(s,\varepsilon\right)\right) - \alpha \nabla_\theta \log f_\theta\left(s,\varepsilon\right). \tag{3}$$

However, when $d^\pi(s)$ exhibits large difference from $d^{\mathcal{D}}(s)$, the bias introduced by (3) cannot be ignored. As we will show, correcting such a bias leads to large performance gain.

2.3 Optimistic Actor-Critic (OAC)

OAC [21] is built on top of SAC [7], and proposes to train a separate optimistic exploration policy π_E in addition to the target policy π_T for deployment. π_E is designed to maximize an approximate upper confidence bound $\hat{Q}_{\mathrm{UB}}$ of the true Q value, while π_T is trained to maximize an approximate lower bound $\hat{Q}_{\mathrm{LB}}$. OAC shows that exploration using π_E avoids *pessimistic under-exploration* and *directional uniformedness* [21], achieving higher sample efficiency over SAC.

To obtain the approximate upper bound $\hat{Q}_{\mathrm{UB}}$, OAC first computes the mean and variance of the Q estimates from $\hat{Q}_1$ and $\hat{Q}_2$,

$$\mu_Q = \frac{1}{2}(\hat{Q}_1 + \hat{Q}_2), \quad \sigma_Q = \sqrt{\sum_{i\in\{1,2\}} \frac{1}{2}\left(\hat{Q}_i - \mu_Q\right)^2} = \frac{1}{2}|\hat{Q}_1 - \hat{Q}_2|, \tag{4}$$

then define $\hat{Q}_{\mathrm{UB}} = \mu_Q + \beta_{\mathrm{UB}}\sigma_Q$, where $\beta_{\mathrm{UB}} \in \mathbb{R}^+$ controls the level of optimism. Note that the previous approximate lower bound $\hat{Q}_{\mathrm{LB}}(s,a) = \min_{i\in\{1,2\}} \hat{Q}_i(s,a)$ can be expressed as $\hat{Q}_{\mathrm{LB}} = \mu_Q - \beta_{\mathrm{LB}}\sigma_Q$ with $\beta_{\mathrm{LB}} = 1$.

OAC derives a closed-form solution of the Gaussian exploration policy $\pi_E = (\mu_E, \Sigma_E)$, by solving an optimization problem that maximizes the expectation (w.r.t. π_E) of a linear approximation of $\hat{Q}_{\mathrm{UB}}$ and constraints the maximum KL Divergence between π_E and π_T. Given the trained target policy $\pi_T = (\mu_T, \Sigma_T)$, the closed-form solution of π_E ends up having a μ_E from a *directionally informed* shifts of μ_T, and a Σ_E equal to Σ_T.

Although the KL constraint enables a closed-form solution of π_E and stabilizes the training, we argue that such a constraint limits the potential of π_E in executing more informative actions that may better correct the spurious estimates of critics, as π_E is constrained to generate actions that are not far from those generated by π_T which is trained conservatively from the critics. In our framework, we remove the KL constraint to release the full potential of an optimistic exploration policy while addressing the training stability by explicitly correcting the biased gradient estimator in policy training.

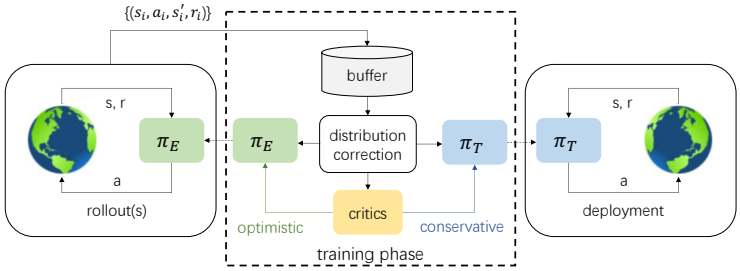

Figure 1: Overview of our off-policy actor critic framework

2.4 Distribution Correction Estimation

Distribution Correction Estimation (DICE) family [14, 15, 16, 17, 22, 18, 19] is proposed to solve the OPE task, which seeks an estimator of the policy value, i.e., the normalized expected per-step reward, from a static replay buffer $\mathcal{D}$. DICE obtains an unbiased estimator by estimating the distribution correction ratio $\zeta^*(s,a) = \frac{d^\pi(s,a)}{d^\mathcal{D}(s,a)}$, where $d^\mathcal{D}(s,a)$ is the state-action distribution of $\mathcal{D}$.

We adapt the DICE optimization scheme to the off-policy actor-critic framework, and use it to estimate the distribution correction ratio for our training.

3 Optimistic Exploration with Explicit Distribution Correction

Following the philosophy of OAC and recent advances in the off-policy RL [23], we propose to simultaneously train a conservative target policy π_T for deployment, and an optimistic exploration policy π_E for executing the most informative actions w.r.t the current Q value estimation. Figure 1 provides an overview of our proposed framework. We apply explicit distribution corrections to train both policies and critics, and no constraint is imposed on π_E in terms of its divergence from π_T.

In the rest of this section, we first introduce our off-policy training objectives and theoretically show that it forms an unbiased policy gradient estimator for π_T with the distribution correction ratio. We then elaborate on how to estimate the distribution correction ratio by adapting the DICE optimization scheme. We close this section by summarizing our algorithm with pseudo-codes.

3.1 Off-policy Training Objectives

π_T and π_E are parameterized by θ_T and θ_E respectively. We start from the training objective for the target policy π_T. To prevent from overestimation, we follow previous work [7, 8, 21] in using the approximate lower bound $\hat{Q}_{\text{LB}}$ as the critic, and correct the sample distribution with $\frac{d^{\pi_T}(s,a)}{d^\mathcal{D}(s,a)}$,

$$\hat{J}^{\pi_T} = \sum_{(s,a)\in\mathcal{D}} \frac{d^{\pi_T}(s,a)}{d^\mathcal{D}(s,a)} \big(\hat{Q}_{\text{LB}}\left(s, f_{\theta_T}\left(s,\varepsilon\right)\right) - \alpha \log f_{\theta_T}\left(s,\varepsilon\right)\big). \tag{5}$$

The following proposition (proved in the Appendix) shows that the gradient of training objective (5) provides an unbiased estimation of the policy gradient (2). This is in contrast with previous off-policy actor-critic algorithms that directly apply the biased estimator (3) in training the target policy.

Proposition 3.1. $\nabla_{\theta_T}\hat{J}^{\pi_T}$ *gives an unbiased estimate of the policy gradient* (2) *with* $\pi = \pi_T$.

Regarding the exploration policy π_E, there are two considerations. Firstly, it should explore optimistically w.r.t. the estimated Q values, in order to collect informative experience to efficiently correct spurious Q estimates. Therefore, we follow OAC to use the approximate upper bound $\hat{Q}_{\text{UB}}$ as critic in the objective. Secondly, the ultimate goal of π_E and better Q value estimation is to facilitate a more accurate gradient estimate for π_T. Therefore, the training sample distribution for π_E's objective should be consistent with that for π_T's. As a result, we apply the same correction ratio as in (5) to get the following objective for π_E,

$$\hat{J}^{\pi_E} = \sum_{(s,a)\in\mathcal{D}} \frac{d^{\pi_T}(s,a)}{d^\mathcal{D}(s,a)} \big(\hat{Q}_{\text{UB}}\left(s, f_{\theta_E}\left(s,\varepsilon\right)\right) - \alpha \log f_{\theta_E}\left(s,\varepsilon\right)\big), \tag{6}$$

again, this is different from all existing works that adopt a separate exploration policy to the best of our knowledge.

As for training the critics, while the best sample distribution for performing off-policy Q learning is still an open problem [24], there exists some empirical work[25, 18] showing that using samples from the same distribution as that for policy training benefits the training of Q functions. Therefore, we also apply the same correction ratio to Q function objectives (1) and obtain the corrected objectives:

$$\hat{L}_Q^i = \sum_{(s,a)\in\mathcal{D}} \frac{d^{\pi_T}(s,a)}{d^{\mathcal{D}}(s,a)} \left(\hat{Q}_i(s,a) - \mathcal{B}^\pi \hat{Q}_{\mathrm{LB}}(s,a)\right)^2 \tag{7}$$

3.2 Estimating Distribution Correction Ratio via DICE

We estimate the distribution correction ratio $\frac{d^{\pi_T}(s,a)}{d^{\mathcal{D}}(s,a)}$ by adapting the DICE optimization scheme [19] into our framework. [19] shows that existing DICE estimators can be formulated as the following min-max distributional linear program with different regularizations,

$$\begin{aligned}\max_{\zeta\geq 0}\min_{\nu,\lambda}\ L_D(\zeta,\nu,\lambda) :=& (1-\gamma)\cdot \mathbb{E}_{\substack{a_0\sim\pi_T(s_0)\\ s_0\in\rho_0(s)}}\left[\nu\left(s_0,a_0\right)\right] \\ &+ \lambda\cdot\left(1-\mathbb{E}_{(s,a,r,s')\sim d^{\mathcal{D}}}\left[\zeta(s,a)\right]\right) - \alpha_\zeta\cdot\mathbb{E}_{(s,a)\sim d^{\mathcal{D}}}\left[g_2(\zeta(s,a))\right] \\ &+ \mathbb{E}_{(s,a,r,s')\sim d^{\mathcal{D}}}\left[\zeta(s,a)\cdot\left(\mathcal{B}^{\pi_T}\nu(s,a)-\nu(s,a)\right)\right] \\ &+ \alpha_\nu\cdot\mathbb{E}_{(s,a)\sim d^{\mathcal{D}}}\left[g_1(\nu(s,a))\right],\end{aligned} \tag{8}$$

where g_1 and g_2 are some convex and lower-semicontinuous regularization functions, α_ν and α_ζ control the corresponding regularization strength. By slightly abusing the notation, we define $\mathcal{B}^{\pi_T}\nu(s,a) = \alpha_R\cdot R(s,a) + \gamma\nu\left(s',a'\right), a' \sim \pi_T(s')$ with α_R scaling the reward. Solving the optimization problem (8) for ζ gives us $\zeta^*(s,a) = \frac{d^{\pi_T}(s,a)}{d^{\mathcal{D}}(s,a)}$, which is exactly the distribution correction ratio. Note that the formulation supports both discounted and undiscounted return cases.

We now provide more intuition for (8). We refer to RHS as the RHS of (8) for simplicity. As $\zeta^*(s,a)$ is the distribution correction ratio, λ is the Lagrangian coefficient to ensure the expectation of $\zeta^*(s,a)$ to be 1. ν can be viewed as a synonym to the Q, with the first line of the RHS giving the normalized expected per-step reward of policy π_T. Line 3 and 4 together can be viewed as the expected single-step *bellman error*, which becomes clear if referring to the Fenchel Duality and setting $g_2(\zeta) = \frac{1}{2}\zeta^2$. We refer interested readers to the original paper [19] for more details.

Directly applying the above DICE algorithms to the off-policy RL settings casts great optimization difficulties. While OPE assumes a fixed target policy and a static replay buffer with sufficient state-action space coverage [14], in the RL both the target policy and the replay buffer change during training. We thus make three modifications to (8) to adapt it to our off-policy actor-critic framework.

Our first modification to (8) is removing the $\mathbb{E}_{s_0\in\rho_0(s),a_0\sim\pi(s_0)}\left[\nu\left(s_0,a_0\right)\right]$ term from the objective. In the initial training stage, the state-action pairs stored in the replay buffer only achieve little coverage of the entire state-action space, especially for high dimensional continuous control tasks. As a result, the target policy π_T may sample some a_0 that is very different from any collected experience in the replay buffer. $\nu(s_0,a_0)$ will then be minimized pathologically low and thus incur a huge error. Note that similar phenomena also appear in the offline RL due to the lack of corrective feedback [26, 27].

Our second modification is to add an absolute value operator $|\cdot|$ to the single step bellman error $\mathcal{B}^\pi\nu(s,a) - \nu(s,a)$. We empirically find that it increases the training stability. The intuition comes from considering solving for L_ζ defined by (11), since when g_2 is square function then the absolute value automatically helps the ζ function lie on the positive region without any hard projection.

Our third modification to the original problem (8) is to set both α_ν and α_ζ greater than zero. This is different from existing DICE schemes [19] that set at least one of α_ν and α_ζ to be zero. Though such a setting introduces bias [19] and has never been applied in existing DICE algorithms, we found in practice that it dramatically stabilizes the training. Moreover, adding two regularizations on a min-max program has been widely applied for faster convergent in optimization literature [28].

Algorithm 1 Optimistic Exploration with Explicit Distribution Correction

Input: target and explore policy parameters θ_T, θ_E; $\hat{Q}_1$, $\hat{Q}_2$ parameters ω_1, ω_2; $\hat{Q}'_1$, $\hat{Q}'_2$ parameters ω'_1, ω'_2; ν, ζ parameters η_ν, η_ζ, ν' parameters $\eta_{\nu'}$, and λ; replay buffer $\mathcal{D}$

$w'_1 \leftarrow w_1$, $w'_2 \leftarrow w_2$, $\eta_{\nu'} \leftarrow \eta_\nu$, $\mathcal{D} \leftarrow \emptyset$
repeat
  for each environment step **do**
    Sample action $a_t \sim \pi_E(\cdot|s_t)$
    Observe $s_{t+1} \sim T(\cdot|s_t, a_T)$ and $r(\mathbf{s}_t, \mathbf{a}_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$
  end for
  for each training step **do**
    Sample minibatch $B = \{(s, a, r, s')\}$ from $\mathcal{D}$
    # DICE training
    Update η_ν, η_ζ, λ via $\nabla_{\eta_\nu} L_\nu$, $\nabla_{\eta_\zeta} L_\zeta$, $\nabla_{\psi_\lambda} L_\lambda$
    Calculate $\tilde{\zeta}$ with (12)
    # RL training
    Update θ_T with $\nabla_{\theta_T} \hat{J}^{\pi_T}$
    Update θ_E with $\nabla_{\theta_E} \hat{J}^{\pi_E}$
    Update w_i with $\nabla_{w_i} \hat{L}^i_Q$, $i \in \{1, 2\}$
    # Soft update
    $w'_i \leftarrow \tau w_i + (1-\tau) w'_i$, $i \in \{1, 2\}$
    $\eta_{\nu'} \leftarrow \tau \eta_\nu + (1-\tau)\eta_{\nu'}$
  end for
until convergence

To this end, we obtain as follows the loss functions for each of ν, ζ, and λ respectively,

$$L_\lambda = \lambda \cdot \left(1 - \mathbb{E}_{(s,a,r,s')\sim d^{\mathcal{D}}}[\zeta(s,a)]\right), \tag{9}$$

$$L_\nu = \mathbb{E}_{(s,a,r,s')\sim d^{\mathcal{D}}}\left[\zeta(s,a) \cdot |\mathcal{B}^{\pi_T}\nu(s,a) - \nu(s,a)|\right] + \alpha_\nu \cdot \mathbb{E}_{(s,a)\sim d^{\mathcal{D}}}\left[f_1(\nu(s,a))\right], \tag{10}$$

$$L_\zeta = \alpha_\zeta \cdot \mathbb{E}_{(s,a)\sim d^{\mathcal{D}}}\left[f_2(\zeta(s,a))\right] - \mathbb{E}_{(s,a,r,s')\sim d^{\mathcal{D}}}\left[\zeta(s,a) \cdot (|\mathcal{B}^{\pi_T}\nu(s,a) - \nu(s,a)| - \lambda)\right]. \tag{11}$$

Both ζ and ν are parameterized by neural networks with parameters η_ζ and η_ν respectively. To stabilize the training, we maintain a ν' as the exponential moving average of ν to compute $\mathcal{B}^{\pi_T}\nu(s,a)$.

When applying the learned $\zeta(s,a)$ as a surrogate of $\frac{d^{\pi_T}(s,a)}{d^{\mathcal{D}}(s,a)}$ in the RL training objectives, we need to tackle the issue of finite samples since our training is in minibatch. Inspired by [25], we apply self-normalization to $\zeta(s,a)$ with temperature T, and use the $\tilde{\zeta}(s,a)$ defined below to construct the RL objectives (5), (6) and (7).

$$\tilde{\zeta}(s,a) = \frac{\zeta(s,a)^{1/T}}{\sum_{(s,a)\sim\mathcal{D}} \zeta(s,a)^{1/T}}. \tag{12}$$

3.3 Summary of Algorithm

To this end, we summarize our overall algorithm in Algorithm 1. Like conventional off-policy actor critic algorithms [7, 8, 21], we alternatively perform environment interactions following π_E and update of all trainable parameters. In each training step, our algorithm first updates the ν, ζ, and λ related to DICE, using SGD w.r.t. losses (10), (11), and (9) respectively, and computes the distribution correction ratio $\tilde{\zeta}$ from the updated ζ. With $\tilde{\zeta}$, our algorithm then performs RL training to update π_T, π_E, $\hat{Q}_1$, and $\hat{Q}_2$. Finally, soft update of target critics $\hat{Q}'_1$, $\hat{Q}'_2$, and ν' is performed

4 Experimental Results

We validate the proposed algorithm from three perspectives. First, we compare our algorithm with SOTA off-policy actor-critic algorithms on challenging control tasks. Second, we examine the quality

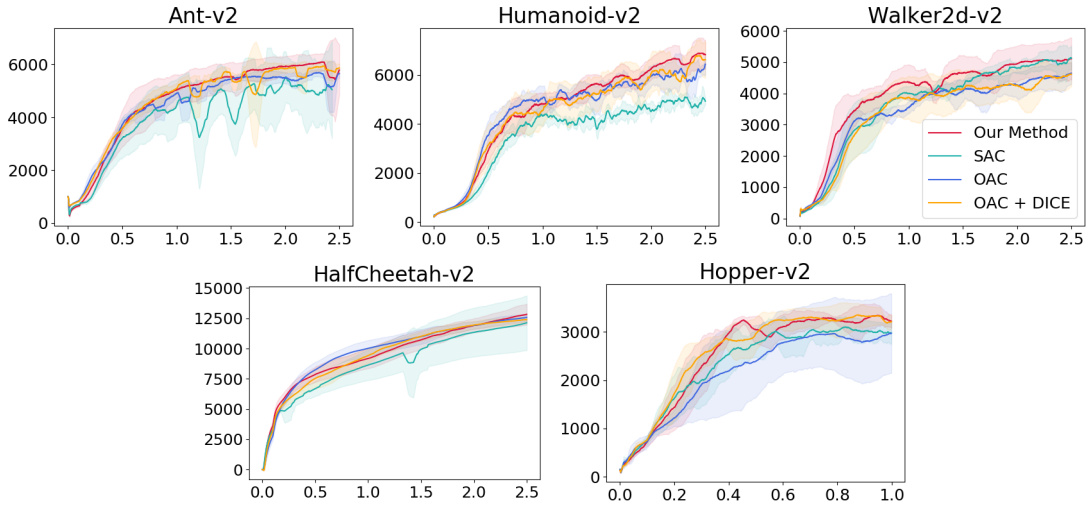

Figure 2: Our methods versus SAC, OAC, and the strong baseline OAC + DICE on 5 MuJoco tasks. x-axis indicates the number of environment steps. y-axis indicates the total undiscounted return. The shaded areas denote one standard deviation between 5 runs of different random seeds.

of our learned correction ratio by evaluating whether it helps to provide a reasonable estimate of the on-policy rewards. Third, we ablate key components of our algorithms to evaluate their contributions to the overall performance.

We evaluate on 5 continuous control tasks from the MuJoCo [29] benchmark and compare it with SAC [7] and OAC [21]. We further implement a strong baseline OAC + DICE by integrating our DICE correction into OAC, i.e., similar to (5) and (7), using the learned distribution correction ratio to correct the sample distribution for both the target policy training and the critics training.

For each task, we run 5 different random seeds for each algorithm. We evaluate each algorithm every 1000 training steps. Our algorithm introduces additional hyper-parameters (HP), including β_{LB} and β_{UB} that control the extent of conservativeness and optimism respectively, the temperature T from (7), and the set of hyper-parameters from the DICE optimization (9), (10), (11). We provide detailed hyper-parameter tuning procedures in the Appendix C. Note that the implementation of our algorithms are based on OAC's officially released codes[1], which themselves are based on the SAC implementation in RLkit [2]. While we are aware of different implementations of SAC with performance variance on different tasks, given the difficulty in reproducing the best RL results in practice [30, 31], we have tried our best to make all comparisons fair and reproducible by having all competing methods and variants implemented in a unified and credible codebase.

4.1 Evaluation on Continuous Control Tasks

As shown in Figure 2, our method consistently outperforms both OAC and SAC on 4 out of the 5 tasks, while being on par with OAC on the less challenging HalfCheetah. The superior performance of our method vs OAC and SAC demonstrates the effectiveness of the proposed framework and the benefits of both the optimistic exploration and distribution correction for off-policy RL.

We would also like to highlight the performance of our specific baseline OAC + DICE. While it outperforms OAC by a large margin on Hopper, there are only slight improvements over OAC on Humanoid and Ant, even with no performance gain on Walker2d and HalfCheetah. These results demonstrate that the proposed distribution correction scheme at least does not harm the performance and often leads to improvements for off-policy actor-critic algorithms beyond ours. Moreover, the performance of our method vs. OAC + DICE on the more challenging Humanoid, Ant, and Walker2d demonstrates that our exploration policy design is superior to that of OAC when paired with the DICE distribution correction. This is not surprising. Due to the use of KL constraint in OAC, the

[1] `https://github.com/microsoft/oac-explore`
[2] `https://github.com/rail-berkeley/rlkit`

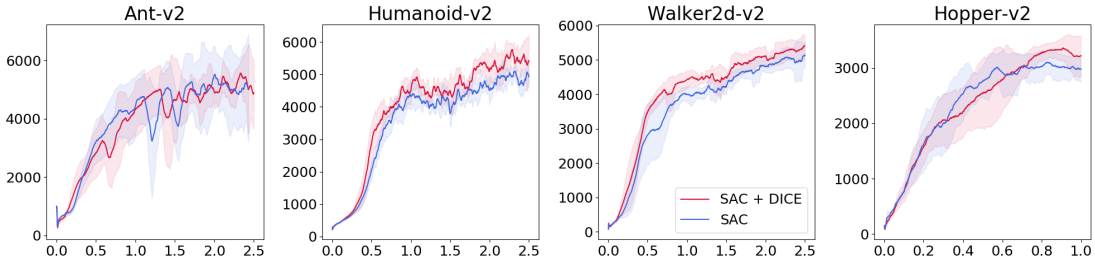

Figure 3: Comparing SAC + DICE vs. SAC on 4 Mujoco environments. x-axis indicates the number of environment steps. y-axis indicates the total undiscounted return. The shaded areas denote one standard deviation between 5 runs of different random seeds.

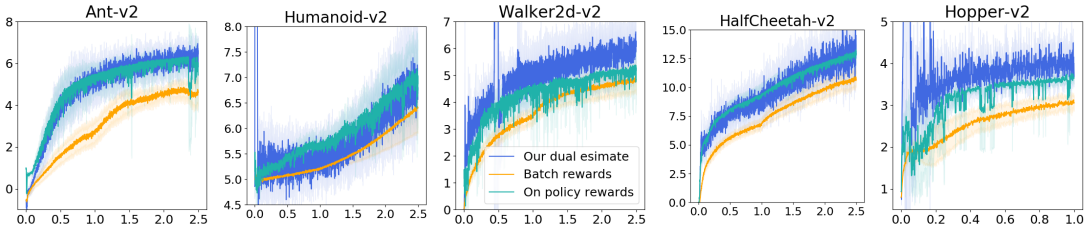

Figure 4: Reward estimation results of our method (Our dual estimate), a naive baseline (Batch rewards), and a reference (On-policy rewards) on 5 Mujoco environments. x-axis is the number of environment steps. y-axis is the reward value. The shaded areas denote one standard deviation between 5 runs of different random seeds.

actions generated by π_E can not differ much from those generated by π_T for most of the states. As a result, the potential of π_E in collecting the most informative experience for correcting the critics has been largely constrained. Our method, on the other hand, does not impose any constraints on the distance between π_E and π_T, and therefore learns a more effective exploration policy for collecting the informative experience for off-policy training.

Similarly, we implement SAC + DICE by integrating our DICE correction ratio into SAC without any HP tuning. As shown in Figure 3, SAC + DICE outperforms SAC on the challenging Humanoid and Walker2d by a clear margin while being on par with SAC on the Ant. There is also a slight performance gain on the Hopper. It is worth noting that integrating our learned DICE correction ratio into **NO DICE** (a variant of ours), OAC, and SAC all results in a performance gain. We thus highlight that our proposed DICE correction scheme effectively benefits general off-policy actor-critic training.

4.2 Examine the Correction Ratio

To examine the quality of the learned distribution correction ratio, we propose to tackle a novel OPE setting that is more challenging than the traditional counterpart, where both the target policy and the replay buffer are changing.

Following our Algorithm 1 and the same experiment settings in Section 4.1, we now use the following dual estimator defined in [19] to estimate the on-policy average per-step rewards $r_{on}^{\pi_T}$ (on-policy rewards) of π_T,

$$\hat{\mu}_\zeta(\pi_T) = \sum_{(s,a,r)\in\mathcal{D}} \zeta(s,a) r, \tag{13}$$

where ζ is the distribution correction ratio estimated in Algorithm 1.

We use the metric $|\hat{\mu}_\zeta(\pi_T) - r_{on}^{\pi_T}|$ to evaluate the OPE performance, which directly reflects the quality of $\zeta(s,a)$. If $|\hat{\mu}_\zeta(\pi) - r_{on}^{\pi_T}| < |r_{\mathcal{D}} - r_{on}^{\pi_T}|$, where $r_{\mathcal{D}}$ is the mean reward in the current replay buffer $\mathcal{D}$, correcting the training distribution of RL using the learned ζ can at least provide positive momentum to the overall training.

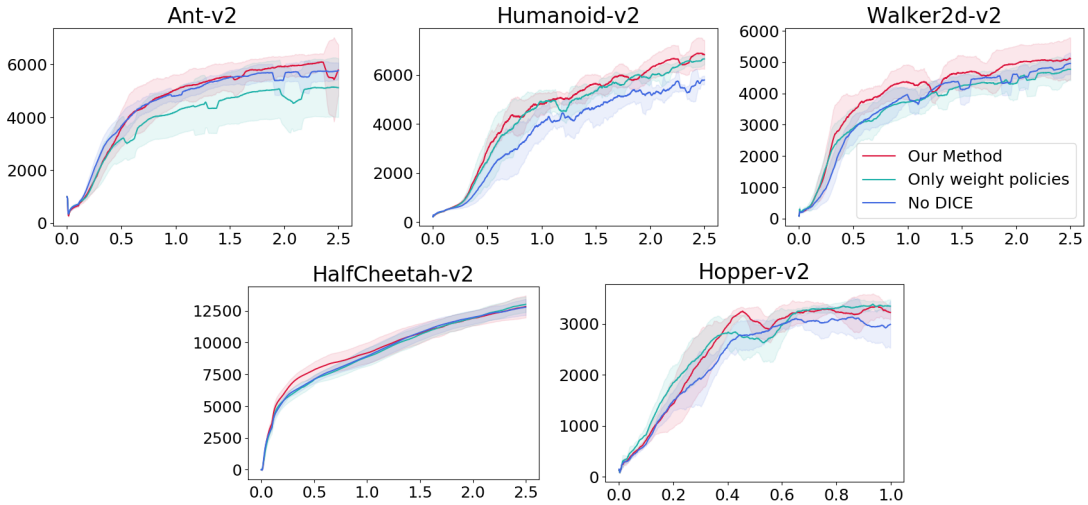

Figure 5: Ablation on our method on 5 Mujoco environments. x-axis indicates the number of environment steps. y-axis indicates the total undiscounted return. The shaded areas denote one standard deviation. Please zoom-in to see more details.

Figure 4 shows the performance of two OPE methods, i.e., "Our dual estimate" (13) and the naive "Batch rewards" baseline, both are estimated using a batch of transitions. We also plot the "on policy rewards" $r_{on}^{\pi_T}$ as a reference which averages over the rewards collected by the target policy during evaluation. As shown in Figure 4, our OPE performance is surprisingly decent. The $\hat{\mu}_\zeta(\pi)$ learned by our method nicely matches the on-policy reward estimate on Humanoid, Ant, and HalfCheetah, while staying close to the on-policy reward estimate on Hopper and Walker2d. These experimental results demonstrate the effectiveness of our adaptation of the DICE scheme from standard OPE to the off-policy RL setting, which also explains the success of our proposed algorithm in experiments of Section 4.1 where the power of this adapted DICE scheme has been leveraged.

4.3 Ablation Study

We emphasize the key of our framework is to explicitly correct the RL training distribution while training an explore policy solely to correct the flawed Q value that undermines the target policy's gradient estimate. In Section 4.1, we already show that having such an exploration is beneficial. We now show by ablation that such an aggressive training strategy is indeed backed by our distribution correction techniques. The first ablated version of our method, **No DICE**, does not have the DICE component, i.e., its training procedures can be summarized as Algorithm 1 without Line 10-12, and always set $\tilde{\zeta}(s, a) = 1$. As shown in Figure 5, the performance drops substantially on Humanoid, Walker2d, and Hopper, and somewhat on Ant and HalfCheetah, demonstrating the necessity of performing an effective distribution correction when applying an aggressive exploration strategy.

The second ablated version is to check the necessity of correcting the state-action distribution w.r.t the Q function loss. We term the version of our method without applying the correction ratio to the Q function objective as **Only weight policies**. From Figure 5, we can see the performance dropped dramatically on Ant and slightly on the other tasks. We include more ablation studies in the Appendix, showing how different HP influences the performance of our method.

5 Conclusion

In this paper, we proposed a novel off-policy RL paradigm that enables the training of an exploration policy that solely focuses on executing the most informative actions without constraining its divergence from the target policy. We support such an efficient exploration by learning the distribution correction ratio to correct the training distribution, which is realized by successfully adapting the DICE [19] optimization schemes into the off-policy actor-critic framework. By evaluating in challenging continuous control tasks, we demonstrate the effectiveness of our method.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[2] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[3] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[4] Guy Shani, David Heckerman, Ronen I Brafman, and Craig Boutilier. An mdp-based recommender system. *Journal of Machine Learning Research*, 6(9), 2005.

[5] Jiachen Li, Quan Vuong, Shuang Liu, Minghua Liu, Kamil Ciosek, Keith Ross, Henrik Iskov Christensen, and Hao Su. Multi-task batch reinforcement learning with metric learning. *arXiv preprint arXiv:1909.11373*, 2019.

[6] Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

[7] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[8] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *CoRR*, abs/1802.09477, 2018.

[9] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

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

[11] Hado Van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

[12] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

[13] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143. PMLR, 2020.

[14] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019.

[15] Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. Gendice: Generalized offline estimation of stationary values. *arXiv preprint arXiv:2002.09072*, 2020.

[16] Shangtong Zhang, Bo Liu, and Shimon Whiteson. Gradientdice: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, pages 11194–11203. PMLR, 2020.

[17] Masatoshi Uehara, Jiawei Huang, and Nan Jiang. Minimax weight and q-function learning for off-policy evaluation. In *International Conference on Machine Learning*, pages 9659–9668. PMLR, 2020.

[18] Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019.

[19] Mengjiao Yang, Ofir Nachum, Bo Dai, Lihong Li, and Dale Schuurmans. Off-policy evaluation via the regularized lagrangian. *arXiv preprint arXiv:2007.03438*, 2020.

[20] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[21] Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better exploration with optimistic actor-critic. *arXiv preprint arXiv:1910.12807*, 2019.

[22] Ofir Nachum and Bo Dai. Reinforcement learning via fenchel-rockafellar duality. *arXiv preprint arXiv:2001.01866*, 2020.

[23] Kimin Lee, Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In *International Conference on Machine Learning*, pages 6131–6141. PMLR, 2021.

[24] Aviral Kumar, Abhishek Gupta, and Sergey Levine. Discor: Corrective feedback in reinforcement learning via distribution correction. *arXiv preprint arXiv:2003.07305*, 2020.

[25] Samarth Sinha, Jiaming Song, Animesh Garg, and Stefano Ermon. Experience replay with likelihood-free importance weights. *arXiv preprint arXiv:2006.13169*, 2020.

[26] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062, 2019.

[27] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019.

[28] Zeyuan Allen-Zhu, Zhenyu Liao, and Yang Yuan. Optimization Algorithms for Faster Computational Geometry. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 53:1–53:6, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[29] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[30] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*, 2020.

[31] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. Improving reproducibility in machine learning research (a report from the neurips 2019 reproducibility program). *Journal of Machine Learning Research*, 22, 2021.

[32] Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

[33] Michael Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In *IJCAI*, volume 16, pages 740–747, 1999.

[34] Jonathan Sorg, Satinder Singh, and Richard L Lewis. Variance-based rewards for approximate bayesian reinforcement learning. *arXiv preprint arXiv:1203.3518*, 2012.

[35] Manuel Lopes, Tobias Lang, Marc Toussaint, and Pierre-Yves Oudeyer. Exploration in model-based reinforcement learning by empirically estimating learning progress. In *Neural Information Processing Systems (NIPS)*, 2012.

[36] Mauricio Araya, Olivier Buffet, and Vincent Thomas. Near-optimal brl using optimistic local transitions. *arXiv preprint arXiv:1206.4613*, 2012.

[37] Finale Doshi-Velez, David Wingate, Nicholas Roy, and Joshua Tenenbaum. Nonparametric bayesian policy priors for reinforcement learning. 2010.

[38] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295*, 2017.

[39] Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. *arXiv preprint arXiv:1706.01905*, 2017.

[40] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

[41] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

[42] Bradly C Stadie, Sergey Levine, and Pieter Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015.

[43] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. In *31st Conference on Neural Information Processing Systems (NIPS)*, volume 30, pages 1–18, 2017.

[44] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *International conference on machine learning*, pages 5062–5071. PMLR, 2019.

[45] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *arXiv preprint arXiv:1605.09674*, 2016.

[46] Rui Zhao and Volker Tresp. Curiosity-driven experience prioritization via density estimation. *arXiv preprint arXiv:1902.08039*, 2019.

[47] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

[48] Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. *arXiv preprint arXiv:1810.02274*, 2018.

[49] Leshem Choshen, Lior Fox, and Yonatan Loewenstein. Dora the explorer: Directed outreaching reinforcement action-selection. *arXiv preprint arXiv:1804.04012*, 2018.

[50] Misha Denil, Pulkit Agrawal, Tejas D Kulkarni, Tom Erez, Peter Battaglia, and Nando De Freitas. Learning to perform physics experiments via deep reinforcement learning. *arXiv preprint arXiv:1611.01843*, 2016.

[51] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29:4026–4034, 2016.

[52] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

[53] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Appendix

Outline of the Appendix

In this Appendix, we organize the content in the following ways:

- In Appendix A, we additional related works.
- In Appendix B, we present the proof of Proposition 3.1.
- In Appendix C, we present experiment details and hyper-parameter settings
- In Appendix D, we present additional ablation studies to examine how the hyper-parameters β_{UB}, β_{LB} and T impacts the performance. We also show the necessity of correcting the distribution of both the target and exploration policy objectives via comparing with another variant of our approach that only weights the Q function training.

A Additional Related Work

Incorporating the DICE family [14, 15, 16, 17, 22, 18, 19] into the training process of off-policy actor-critic is never easy, as the DICE optimization itself already poses great optimization difficulties. Moreover, in RL, the target policy is always changing and remains unstable. When tackling complicated continuous control tasks with high dimensional state-action space, the instability gets even exacerbated by the limited state-action coverage at the initial training stage [14]. We tackle such difficulties by directly making modifications to the objectives and successfully stabilize the training. Though suffering from bias, experimental results show that our optimization objective actually leads to surprisingly good OPE performance.

In our work, by explicitly correcting the training distribution, we open up the direction to a potentially more profound exploration strategy design. There are a bunch of literature investigate to design better exploration objective [32, 33, 34, 35, 36, 37]. While most of the pioneering works mainly tackle the problem in discrete or small state-action space via directly operate on the transition matrix, which cannot be directly extended to complicated continuous control tasks. Noise-based methods [38, 39] improve exploration by perturbing the observation, state, or the parameters in the function approximator for obtaining diverse actions. Intrinsic motivation has also been introduced to encourage the agent to discover novel states [40, 41, 42, 43, 44] or better understand the environment [45, 46, 47, 48, 49, 50]. Inspired by Thompson sampling, Osband *et al.*[51] propose to improve exploration of DQN [1] by modeling uncertainty. Eysenbach *et al.*[52] propose to encourage exploration through maximizing the diversity of the learned behavior. SAC [7] tries to improve exploration through adding extra regularization on the entropy term, while OAC [21] derives an upper confidence bound on the state-action value function for facilitating exploration. Although these methods demonstrate that good exploration will benefit the learning process, these works neglect the issue of distribution shift and instability, which is also proven to be a non-trival problem in reinforcement learning [24, 25].

B Proof of Proposition 3.1

Proposition 3.1. $\nabla_{\theta_T} \hat{J}^{\pi_T}$ gives an unbiased estimate of the policy gradient (2) with $\pi = \pi_T$.

Proof. Recall that

$$\nabla_\theta J^\pi = \int_s d^\pi(s) \bigg(\int_\varepsilon \nabla_\theta \hat{Q}_{\mathrm{LB}}\left(s, f_\theta(s,\varepsilon)\right) \phi(\varepsilon) d\varepsilon + \alpha \int_\varepsilon -\nabla_\theta \log f_\theta(s,\varepsilon) \phi(\varepsilon) d\varepsilon \bigg) ds$$

$$\hat{J}^{\pi_T} = \sum_{(s,a)\in\mathcal{D}} \frac{d^{\pi_T}(s,a)}{d^{\mathcal{D}}(s,a)} \big(\hat{Q}_{\mathrm{LB}}\left(s, f_{\theta_T}\left(s,\varepsilon\right)\right) - \alpha \log f_{\theta_T}\left(s,\varepsilon\right) \big).$$

Therefore, $\nabla_{\theta_T} \hat{J}^{\pi_T}$ is an unbiased estimator for

$$\begin{aligned}
&\int_s \int_a d^{\mathcal{D}}(s,a) \frac{d^{\pi_T}(s,a)}{d^{\mathcal{D}}(s,a)} \left(\int_\varepsilon \nabla_{\theta_T} \left(\hat{Q}_{\mathrm{LB}}\left(s, f_{\theta_T}(s,\varepsilon)\right) - \alpha \log f_{\theta_T}(s,\varepsilon) \right) \phi(\varepsilon) d\varepsilon \right) dads \\
&= \int_s \int_a d^{\pi_T}(s,a) \left(\int_\varepsilon \nabla_{\theta_T} \left(\hat{Q}_{\mathrm{LB}}\left(s, f_{\theta_T}(s,\varepsilon)\right) - \alpha \log f_{\theta_T}(s,\varepsilon) \right) \phi(\varepsilon) d\varepsilon \right) dads \\
&= \int_s d^{\pi_T}(s) \left(\int_\varepsilon \nabla_{\theta_T} \left(\hat{Q}_{\mathrm{LB}}\left(s, f_{\theta_T}(s,\varepsilon)\right) - \alpha \log f_{\theta_T}(s,\varepsilon) \right) \phi(\varepsilon) d\varepsilon \right) ds = \nabla_{\theta_T} J^{\pi_T}
\end{aligned}$$

□

C Hyper-parameters (HP) and Experiment Details

The implementation of our method is based on OAC's officially released codes[3], which themselves are based on the SAC implementation in **RLkit** [4]. We are aware of the fact that OAC's officially released codes[3] use a set of HP that is slightly different from its original paper. Specifically, they set $\beta_{\text{LB}} = 1.0$ instead of 3.65 as per (Table 1, Appendix E) of OAC's original paper [21].

The implementation to solve the DICE optimization is based on the official **dice_rl** repo[5]. We set the regularization function $g_1(x) = g_2(x) = \frac{1}{m}|x|^m$ in Equation (10) and (11), where we treat the exponential p as a hyper-parameter. All of our results are obtained by averaging across 5 fixed random seeds as per Appendix D of OAC's original paper [21].

We list the HP specific to our method and those different from SAC and OAC in Table 1. For completeness, we include the HP for SAC and OAC[3] in Table 2. We set both $\alpha_\nu = 1$ and $\alpha_\zeta = 1$ as we found this is the only setting to stabilize the DICE optimization in our HP space. We set the temperature $T = 3.0$ as our preliminary experiment finds the value of T from our HP space will not significantly impact the performance, which is proved in our ablation study (Appendix D.1). As for β_{UB} and β_{LB}, we tune their values simultaneously. However, we do not conduct the full sweep in the HP space due to limited computational resources. We empirically find it beneficial to set a similar value for β_{UB} and β_{LB} (Appendix D.2).

Table 1: Hyper-parameters of our method

Hyper-parameters	Value	Range used for tuning
β_{UB}	2.0	$\{1.5, 2.0, 2.5, 3.0\}$
β_{LB}	2.5	$\{1.5, 2.0, 2.5, 3.0\}$
Temperature T	3.0	{2.0, 3.0, 5.0}
Learning rate for ν, ζ, λ	0.0001	Does not tune
α_ν	1	{0, 1}
α_ζ	1	{0, 1}
discount(γ) for DICE	0.99	Does not tune
Regularization function exponential m	1.5	Does not tune
ν, ζ network hidden sizes	[256, 256]	Does not tune
ν, ζ, λ optimizer	Adam [53]	Does not tune
ν, ζ network nonlinearity	ReLU	Does not tune

Table 2: Hyper-parameters of SAC and OAC[3]

Parameter	Value
policy and Q optimizer	Adam [53]
learning rate for policy and Q	$3 \cdot 10^{-4}$
discount(γ)	0.99
replay buffer size	10^6
policy and Q network hidden sizes	[256, 256]
minibatch size	256
policy and Q nonlinearity	ReLU
target smoothing rate(τ)	0.005
target update interval	1
gradient steps	1
OAC-specific Parameter	**Value**
shift multiplier $\sqrt{2\delta}$	6.86
β_{UB}	4.66
β_{LB}	1.0

[3] https://github.com/microsoft/oac-explore

[4] https://github.com/rail-berkeley/rlkit

[5] https://github.com/google-research/dice_rl

D Additional Ablation Study

D.1 Ablation study of the temperature T

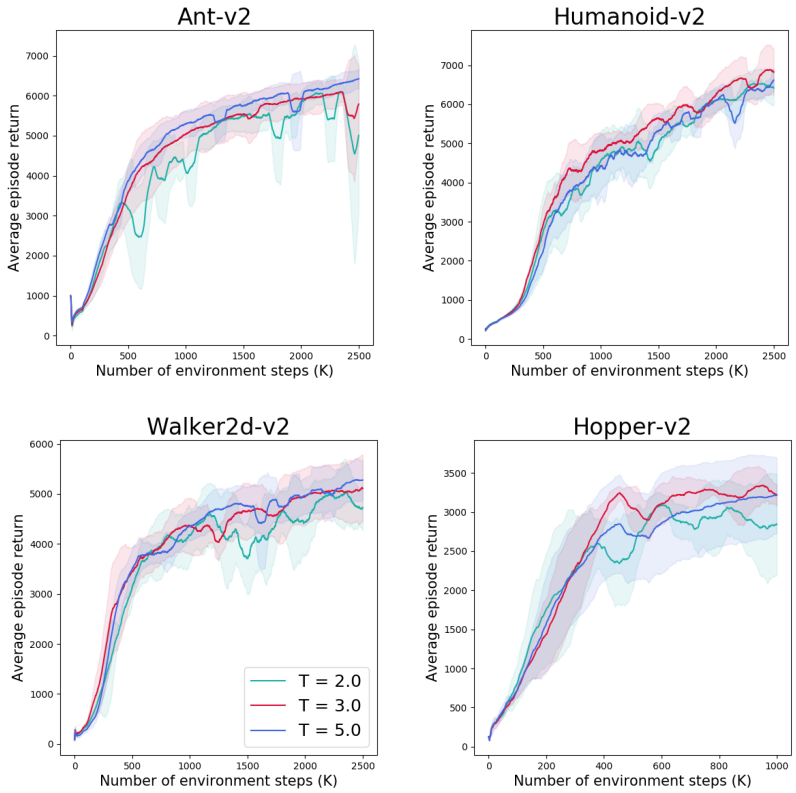

Figure 6: Ablation study on the temperature T of our method on 4 Mujoco environments. x-axis indicates the number of environment steps and y-axis indicates the total undiscounted return. The shaded areas denote one standard deviation between 5 runs of different random seeds.

In this section, we examine how the temperature T influences the performance. Note that we keep the other HP unchanged as per Table 1. For $T \in \{2.0, 3.0, 5.0\}$, we present the performance of Our Method on 4 MuJoCo tasks as shown in $Figure$ 6. A low value of T will introduce instability to the training process, especially for Ant and Walker2d, while the performance on Humanoid remains stable w.r.t different T. With $T \geq 3.0$, we can achieve a stable training process and decent final performance in all 4 tasks.

D.2 Ablation study of β_{UB} and β_{LB}

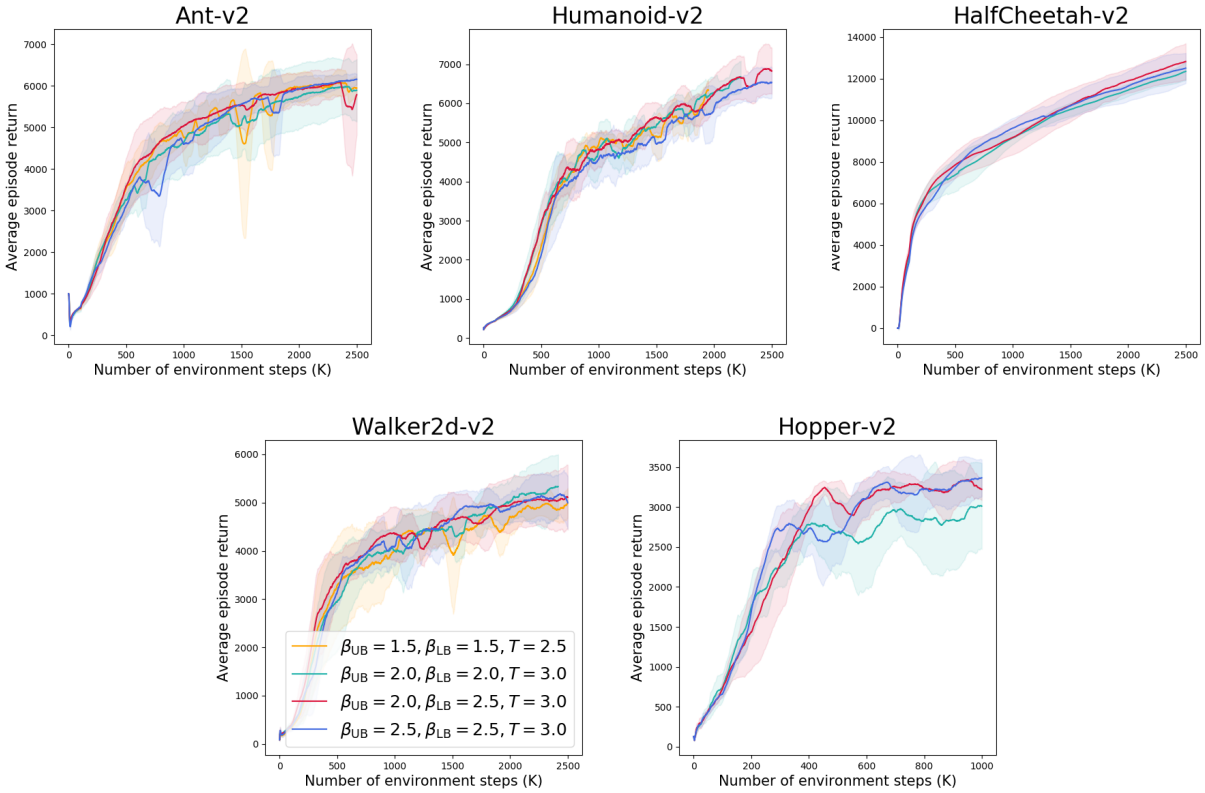

Figure 7: Ablation study on the $\beta_{\text{UB}}, \beta_{\text{LB}}$ of our method on 4 MuJoco tasks. x-axis indicates the number of environment steps. y-axis indicates the total undiscounted return. The shaded areas denote one standard deviation between 5 runs of different random seeds.

In this section, we examine how different values of β_{UB} and β_{LB} influence the performance of our method. We empirically find that setting similar values to β_{UB} and β_{LB} generally lead to good performance.

Note that the performance of $\beta_{\text{UB}} = 1.5$, $\beta_{\text{LB}} = 1.5$, and $T = 2.5$ is to show that Our Method is robust to a wide range of β_{UB} and β_{LB} beyonds $\{2.0, 2.5\}$.

D.3 Ablations on correcting the training distributions of the policies

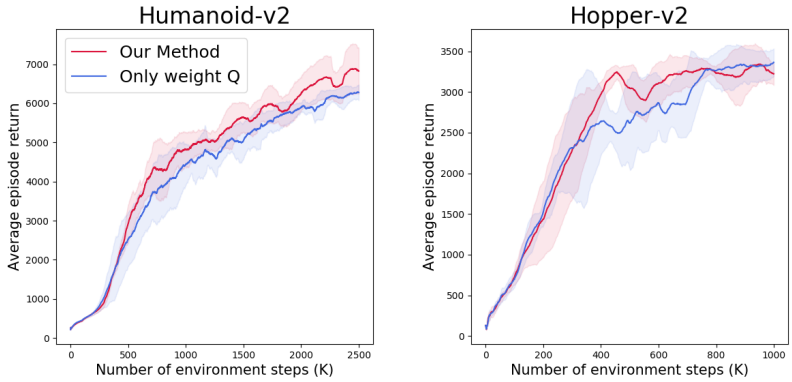

Figure 8: Ablation study on correcting the training distributions of the policies on 2 MuJoCo tasks. x-axis indicates the number of environment steps. y-axis indicates the total undiscounted return. The shaded areas denote one standard deviation between 5 runs of different random seeds.

Similar to **Only weight policies**, we construct the ablated version of our method **Only weight Q** by removing the distribution correction applied to the target policy's objective (5) and exploration policy's objective (6). As shown in Figure 8, ablating the distribution correction for both target and exploration policies of Our Method leads to a clear performance loss on Humanoid. And the ablated version of Our Method also converges slower on the Hopper. Therefore, we can conclude that it is necessary to correct the training distributions of the target and exploration policy.