frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 09 January 2023
DOI 10.3389/fnbot.2022.1081242

OPEN ACCESS

EDITED BY
Huiyu Zhou,
University of Leicester,
United Kingdom

REVIEWED BY
Amit Trivedi,
University of Illinois at Chicago,
United States
Changsheng Li,
Beijing Institute of Technology, China
Zhe Min,
University College London,
United Kingdom
Ning Tan,
Sun Yat-sen University, China

*CORRESPONDENCE
Gang Wang
✉ wanggang@hrbeu.edu.cn

RECEIVED 27 October 2022
ACCEPTED 13 December 2022
PUBLISHED 09 January 2023

CITATION
Li S, Tang Q, Pang Y, Ma X and Wang G (2023) Realistic Actor-Critic: A framework for balance between value overestimation and underestimation. *Front. Neurorobot.* 16:1081242. doi: 10.3389/fnbot.2022.1081242

Realistic Actor-Critic: A framework for balance between value overestimation and underestimation

Sicen Li[1,2], Qinyun Tang[1,2], Yiming Pang[1,2], Xinmeng Ma[1] and Gang Wang[2,3]*

[1]College of Mechanical and Electrical Engineering, Harbin Engineering University, Harbin, China, [2]Science and Technology on Underwater Vehicle Laboratory, Harbin Engineering University, Harbin, China, [3]College of Shipbuilding Engineering, Harbin Engineering University, Harbin, China

Introduction: The value approximation bias is known to lead to suboptimal policies or catastrophic overestimation bias accumulation that prevent the agent from making the right decisions between exploration and exploitation. Algorithms have been proposed to mitigate the above contradiction. However, we still lack an understanding of how the value bias impact performance and a method for efficient exploration while keeping stable updates. This study aims to clarify the effect of the value bias and improve the reinforcement learning algorithms to enhance sample efficiency.

Methods: This study designs a simple episodic tabular MDP to research value underestimation and overestimation in actor-critic methods. This study proposes a unified framework called Realistic Actor-Critic (RAC), which employs Universal Value Function Approximators (UVFA) to simultaneously learn policies with different value confidence-bound with the same neural network, each with a different under overestimation trade-off.

Results: This study highlights that agents could over-explore low-value states due to inflexible under-overestimation trade-off in the fixed hyperparameters setting, which is a particular form of the exploration-exploitation dilemma. And RAC performs directed exploration without over-exploration using the upper bounds while still avoiding overestimation using the lower bounds. Through carefully designed experiments, this study empirically verifies that RAC achieves 10x sample efficiency and 25% performance improvement compared to Soft Actor-Critic in the most challenging Humanoid environment. All the source codes are available at https://github.com/ihuhuhu/RAC.

Discussion: This research not only provides valuable insights for research on the exploration-exploitation trade-off by studying the frequency of policies access to low-value states under different value confidence-bounds guidance, but also proposes a new unified framework that can be combined with current actor-critic methods to improve sample efficiency in the continuous control domain.

KEYWORDS

reinforcement learning (RL), robot control, estimation bias, exploration-exploitation dilemma, uncertainty

1. Introduction

Reinforcement learning is a major tool to realize intelligent agents that can be autonomously adaptive to the environment (Namiki and Yokosawa, 2021; Yu, 2018; Fukuda, 2020). However, current reinforcement learning techniques still suffer from requiring a huge amount of interaction data, which could result in unbearable costs in real-world applications (Karimpanal and Bouffanais, 2018; Levine et al., 2018; Sutton and Barto, 2018; Dulac-Arnold et al., 2020). This study aims to mitigate this problem by better balancing exploration and exploitation.

Undesirable overestimation bias and accumulation of function approximation errors in temporal difference methods may lead to sub-optimal policy updates and divergent behaviors (Thrun and Schwartz, 1993; Pendrith and Ryan, 1997; Fujimoto et al., 2018; Chen et al., 2022). Most model-free off-policy RL methods learn approximate lower confidence bound of Q-function (Fujimoto et al., 2018; Kuznetsov et al., 2020; Lan et al., 2020; Chen et al., 2021; Lee et al., 2021) to avoid overestimation by introducing underestimation bias. However, if the lower bound has a spurious maximum, it will discourage policy to explore potentially higher uncertain regions, resulting in stochastic local-maximum and causing pessimistic underexploration (Ciosek et al., 2019). Moreover, directionally uninformed (Ciosek et al., 2019) policies, such as Gaussian policies, cannot avoid fully explored wasteful actions.

Optimistic exploration methods (Brafman and Tennenholtz, 2002; Kim et al., 2019; Pathak et al., 2019) learn upper confidence bounds of the Q-function from an epistemic uncertainty estimate. These methods are directionally informed and encourage policy to execute overestimated actions to help agents escape local optimum. However, such upper confidence bound might cause an agent to over-explore low-value regions. In addition, it increases the risk of value overestimation since transitions with high uncertainty may have higher function approximation errors to make the value overestimated. To avoid the above problems, one must carefully adjust hyperparameters and control the bias to keep the value at a balance point between lower and higher bounds: supporting stable learning while providing good exploration behaviors. We highlight that this balance is a particular form of the exploration–exploitation dilemma (Sutton and Barto, 2018). Unfortunately, most prior works have studied the overestimation and pessimistic underexploration in isolation and have ignored the under-/overestimation trade-off aspect.

We formulate the Realistic Actor-Critic (RAC), whose main idea is to learn together values and policies with different trade-offs between underestimation and overestimation in the same network. Policies guided by lower bounds control overestimation bias to provide consistency and stable convergence. Each policy guided by different upper bounds provides a unique exploration strategy to generate overestimated actions, so that the policy family can directionally explore overestimated state-action pairs uniformly and avoid over-exploration. All transitions are stored in a shared replay buffer, and all policies benefit from them to escape spurious maximum. Such a family of policies is jointly parameterized with the Universal Value Function Approximators (UVFA) (Schaul et al., 2015). The learning process can be considered as a set of auxiliary tasks (Badia et al., 2020b; Lyle et al., 2021) that help build shared state representations and sills.

However, learning such policies with diverse behaviors in a single network is challenging since policies vary widely in behavior. We introduce punished Bellman backup, which calculates uncertainty as punishment to correct value estimations. Punished Bellman backup provides fine-granular estimation control to make value approximation shift smoothly between upper and lower bounds, allowing for more efficient training. An ensemble of critics is learned to produce well-calibrated uncertainty estimations (i.e., standard deviation) on unseen samples (Amos et al., 2018; Pathak et al., 2019; Lee et al., 2021). We show empirically that RAC controls the standard deviation and the mean of value estimate bias to close to zero for most of the training. Benefiting from well-bias control, critics are trained with a high update-to-data (UTD) ratio (Chen et al., 2021) to improve sample efficiency significantly.

Empirically, we implement RAC with SAC (Haarnoja et al., 2018) and TD3 (Fujimoto et al., 2018) in continuous control benchmarks (OpenAI Gym Brockman et al., 2016, MuJoCo Todorov et al., 2012). Results demonstrate that RAC significantly improves the performance and sample efficiency of SAC and TD3. RAC outperforms the current state-of-the-art algorithms (MBPO Janner et al., 2019, REDQ Chen et al., 2021, and TQC Kuznetsov et al., 2020), achieving state-of-the-art sample efficiency on the Humanoid benchmark. We perform ablations and isolate the effect of the main components of RAC on performance. Moreover, we perform hyperparameter ablations and demonstrate that RAC is stable in practice. The higher sample efficiency allows RAC to facilitate further applications of the RL algorithm in automatic continuous control.

This study makes the following contributions:

(i) Highlighting that agents could over-explore low-value states due to inflexible under-/overestimation trade-off in the fixed hyperparameters setting, and it is a particular form of the exploration–exploitation dilemma;
(ii) Defining a unified framework called Realistic Actor-Critic (RAC), which employs Universal Value Function Approximators (UVFA) to simultaneously learn policies with different value confidence-bond with the same neural network, each with a different under-/overestimation trade-off;
(iii) Experimental evidence that the performance and sample efficiency of the proposed method are better than state-of-the-art methods on continuous control tasks.

The study is organized as follows. Section 2 describes related works and their results. Section 3 describes the problem setting and preliminaries of RL. Section 4 explains the under-/overestimation trade-off. Section 5 introduces the punished Bellman backup and RAC algorithm. Section 6 presents experimental results that show the sample efficacy and final performance of RAC. Finally, Section 7 presents our conclusions.

2. Related works

2.1. Underestimation and overestimation of Q-function

The maximization update rule in Q-learning has been shown to suffer from overestimation bias which is cited as the reason for nonlinear function approximation fails in RL (Thrun and Schwartz, 1993).

Minimizing the value ensemble is a standard method to deal with overestimation bias. Double DQN (Van Hasselt et al., 2016) was shown to be effective in alleviating this problem for discrete action spaces. Clipped double Q-learning (CDQ) (Fujimoto et al., 2018) took the minimum value between a pair of critics to limit overestimation. Maxmin Q-learning (Lan et al., 2020) mitigated the overestimation bias by using a minimization over multiple action-value estimates. However, minimizing a Q-function set cannot filter out abnormally small values, which causes undesired pessimistic underexploration problem (Ciosek et al., 2019). Using minimization to control overestimation is coarse and wasteful as it ignores all estimates except the minimal one (Kuznetsov et al., 2020).

REDQ (Chen et al., 2021) proposed in-target minimization, which used a minimization across a random subset of Q-functions from the ensemble to alleviate the above problems. REDQ (Chen et al., 2021) showed that their method reduces the standard deviation of the Q-function bias to close to zero for most of the training. Truncated Quantile Critics (TQC) (Kuznetsov et al., 2020) truncated the right tail of the distributional value ensemble by dropping several of the topmost atoms to control overestimation. Weighted bellman backup (Lee et al., 2021) and uncertainty-weighted actor-critic (Wu et al., 2021) prevent error propagation (Kumar et al., 2020) in Q-learning by reweighing sample transitions based on uncertainty estimations from the ensembles (Lee et al., 2021) or Monte Carlo dropout (Wu et al., 2021). AdaTQC (Kuznetsov et al., 2021) proposed an auto mechanism for controlling overestimation bias. Unlike prior works, our work does not reweight sample transitions but directly adds uncertainty estimations to punish the target value.

The effect of underestimation bias on learning efficiency is environment-dependent (Lan et al., 2020). Therefore, choosing suitable parameters to balance under- and overestimating for entirely different environments may be hard. This work propose to solve this problem by learning about optimistic and pessimistic policy families.

2.2. Ensemble methods

In deep learning, ensemble methods are often used to solve the two key issues, uncertainty estimations (Wen et al., 2020; Abdar et al., 2021) and out-of-distribution robustness (Dusenberry et al., 2020; Havasi et al., 2020; Wenzel et al., 2020). In reinforcement learning, using an ensemble to enhance value function estimation was widely studied, such as averaging a Q-ensemble (Anschel et al., 2017; Peer et al., 2021), bootstrapped actor-critic architecture (Kalweit and Boedecker, 2017; Zheng et al., 2018), calculating uncertainty to reweight sample transitions (Lee et al., 2021), minimization over ensemble estimates (Lan et al., 2020; Chen et al., 2021), and updating the actor with a value ensemble (Kuznetsov et al., 2020; Chen et al., 2021). MEPG (He et al., 2021) introduced a minimalist ensemble consistent with Bellman update by utilizing a modified dropout operator.

A high-level policy can be distilled from a policy ensemble (Chen and Peng, 2019; Badia et al., 2020a) by density-based selection (Saphal et al., 2020), selection through elimination (Saphal et al., 2020), choosing the action that max all Q-functions (Jung et al., 2020; Parker-Holder et al., 2020; Lee et al., 2021), Thompson sampling (Parker-Holder et al., 2020), and sliding-window UCBs (Badia et al., 2020a). Leveraging uncertainty estimations of the ensemble (Osband et al., 2016; Kalweit and Boedecker, 2017; Zheng et al., 2018) simulated training different policies with a multi-head architecture independently to generate diverse exploratory behaviors. Ensemble methods were also used to learn joint state presentation to improve sample efficiency. There were two main methods: multi-heads (Osband et al., 2016; Kalweit and Boedecker, 2017; Zheng et al., 2018; Goyal et al., 2019) and UVFA (Schaul et al., 2015; Badia et al., 2020a,b). This study uses uncertainty estimation to reduce value overestimation bias, a simple max operator to get the best policy, and learning joint state presentation with UVFA.

2.3. Optimistic exploration

Pessimistic initialization (Rashid et al., 2020) and a learning policy that maximizes a lower confidence bound value could suffer a pessimistic underexploration problem (Ciosek et al., 2019). Optimistic exploration is a promising solution to ease the above problem by applying the principle of optimism in the face of uncertainty (Brafman and Tennenholtz, 2002). Disagreement (Pathak et al., 2019) and EMI (Kim et al., 2019) considered uncertainty as intrinsic motivation to encourage

agents to explore the high-uncertainty areas of the environment. Uncertainty punishment proposed in this study can also be a particular intrinsic motivation. Different from studies of Pathak et al. (2019) and Kim et al. (2019), which usually choose the weight ≥ 0 to encourage exploration, punished Bellman backup use the weight ≤ 0 to control value bias. SUNRISE (Lee et al., 2021) proposed an optimistic exploration that chooses the action that maximizes upper confidence bound (Chen et al., 2017) of Q-functions. OAC (Ciosek et al., 2019) proposed an off-policy exploration policy that is adjusted to a linear fit of upper bounds to the critic with the maximum Kullback–Leibler (KL) divergence constraining between the exploration policies and the target policy. Most importantly, our work provides a unified framework for the under-/overestimation trade-off.

3. Problem setting and preliminaries

In this section, we describe the notations and introduce the concept of maximum entropy RL.

3.1. Notation

We consider the standard reinforcement learning notation, with states $\mathbf{s}$, actions $\mathbf{a}$, reward $r(\mathbf{s}, \mathbf{a})$, and dynamics $p(\mathbf{s}' \mid \mathbf{s}, \mathbf{a})$. The discounted return $R_t = \sum_{k=0}^{\infty} \gamma^k r_k$ is the total accumulated rewards from timestep t, $\gamma \in [0, 1]$ is a discount factor determining the priority of short-term rewards. The objective is to find the optimal policy $\pi_\phi(\mathbf{s} \mid \mathbf{a})$ with parameters ϕ, which maximizes the expected return $J(\phi) = \mathbb{E}_{p_\pi}[R_t]$.

3.2. Maximum entropy RL

The maximum entropy objective (Ziebart, 2010) encourages the robustness to noise and exploration by maximizing a weighted objective of the reward and the policy entropy:

$$\pi^* = \arg\max_{\pi} \sum_t \mathbb{E}_{\mathbf{s}\sim p, \mathbf{a}\sim\pi} \left[r(\mathbf{s}, \mathbf{a}) + \alpha \mathcal{H}\left(\pi\left(\cdot \mid \mathbf{s}\right)\right) \right], \quad (1)$$

where α is the temperature parameter used to determine the relative importance of entropy and reward. Soft Actor-Critic (SAC) (Haarnoja et al., 2018) seeks to optimize the maximum entropy objective by alternating between a soft policy evaluation and a soft policy improvement. A parameterized soft Q-function $Q_\theta(\mathbf{s}, \mathbf{a})$, known as the critic in actor-critic methods, is trained by minimizing the soft Bellman backup:

$$\begin{aligned} \mathcal{L}_{\texttt{critic}}(\theta) &= \mathbb{E}_{\tau\sim\mathcal{B}}[\left(Q_\theta(\mathbf{s}, \mathbf{a}) - y\right)^2], y \\ &= r - \gamma \mathbb{E}_{\mathbf{a}'\sim\pi_\phi}\left[Q_{\bar{\theta}}(\mathbf{s}', \mathbf{a}') - \alpha \log \pi_\phi(\mathbf{a}' \mid \mathbf{s}')\right], \end{aligned} \quad (2)$$

where $\tau = (\mathbf{s}, \mathbf{a}, r, \mathbf{s}')$ is a transition, $\mathcal{B}$ is a replay buffer, $\bar{\theta}$ are the delayed parameters which are updated by exponential moving average $\bar{\theta} \leftarrow \rho\theta + (1-\rho)\bar{\theta}$, ρ is the target smoothing coefficient, and y is the target value. The target value $Q_{\bar{\theta}}(\mathbf{s}', \mathbf{a}')$ is obtained by using two networks $Q^1_{\bar{\theta}}(\mathbf{s}', \mathbf{a}')$ and $Q^2_{\bar{\theta}}(\mathbf{s}', \mathbf{a}')$ with minimum operator:

$$Q_{\bar{\theta}}(\mathbf{s}', \mathbf{a}') = min(Q^1_{\bar{\theta}}(\mathbf{s}', \mathbf{a}'), Q^2_{\bar{\theta}}(\mathbf{s}', \mathbf{a}')). \quad (3)$$

The parameterized policy π_ϕ, known as the actor, is updated by minimizing the following object:

$$\mathcal{L}_{\texttt{actor}}(\phi) = \mathbb{E}_{\mathbf{s}\sim\mathcal{B}, \mathbf{a}\sim\pi_\phi}\left[\alpha \log\left(\pi_\phi\left(\mathbf{a} \mid \mathbf{s}\right)\right) - Q_\theta\left(\mathbf{a}, \mathbf{s}\right)\right]. \quad (4)$$

SAC uses an automated entropy adjusting mechanism to update α with the following objective:

$$\mathcal{L}_{\texttt{temp}}(\alpha) = \mathbb{E}_{\mathbf{s}\sim\mathcal{B}, \mathbf{a}\sim\pi_\phi}\left[-\alpha \log \pi_\phi\left(\mathbf{a} \mid \mathbf{s}\right) - \alpha\overline{\mathcal{H}}\right], \quad (5)$$

where $\overline{\mathcal{H}}$ is the target entropy.

4. Understanding under-/overestimation trade-off

This section briefly discusses the estimation bias issue and empirically shows that a better under-/overestimation trade-off may improve learning performance.

4.1. Under-/overestimation trade-off

Under-/overestimation trade-off is a special form of the exploration–exploitation dilemma. This is illustrated in Figure 1. At first, the agent starts with a policy π_{past}, trained with lower bound $\hat{Q}_{LB}(\mathbf{s}, \mathbf{a})$, becoming π_{LB}. We divide the current action space into four regions:

(i) High uncertainty, low-value. Highly stochastic regions also have low values; overestimation bias might cause an agent to over-explore a low-value area;
(ii) High uncertainty, excessive errors. This region has high uncertainty but is full of unseen transitions that can have excessive-high approximation errors, which may cause catastrophic overestimation and need fewer samples;
(iii) High uncertainty, controllable errors. This region has high uncertainty and is closer to the π_{LB}, with controllable approximation errors, and needs more samples;
(iv) Full explored. Since π_{past} is gradually updated to π_{LB}, the area is fully explored and needs less samples.

To prevent catastrophic overestimation bias accumulation, SAC (Haarnoja et al., 2018), TD3 (Fujimoto et al., 2018), and REDQ (Chen et al., 2021) introduce underestimation bias to learn lower confidence bounds of Q-functions, similar to Equation 3. However, directionally uninformed policies, such as gaussian policies, will sample actions located in region

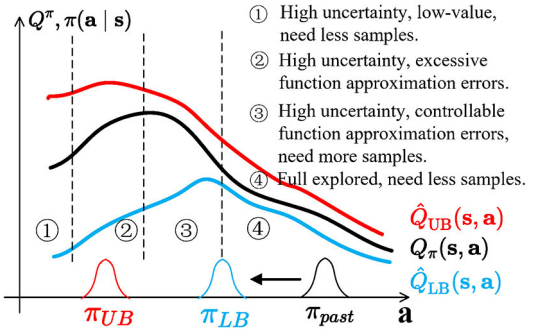

FIGURE 1
Balance between value underestimation and overestimation in actor-critic methods. The state s is fixed. The graph shows Q_π **(in black)**, which is unknown to the algorithm, estimated lower bound $\hat{Q}_{LB}$ **(in blue)**, higher bound $\hat{Q}_{UB}$ **(in red)**, two policies, π_{LB} **(in blue)** and π_{past} **(in black)**, at different time steps of the algorithm, and exploration policies π_{UB} **(in red)** for optimistic exploration.

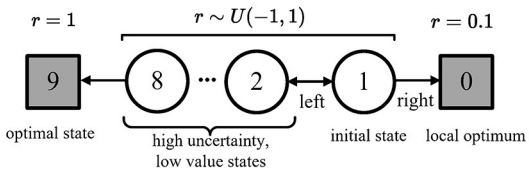

FIGURE 2
A simple episodic MDP (Lan et al., 2020), adapted from Figure 6.5 in the study of Sutton and Barto (2018). This MDP has two terminal states: state 9 and state 0. Every episode starts from state 1, which has two actions: **Left** and **Right**. The MDP is deterministic. Once the agent takes into any states, the MDP will reward back: $r = 0.1$ for terminal states 0, $r = 1$ for terminal states 9, and a reward $r \sim U(-1, 1)$ for non-terminal states 1–8. State 9 is the optimal state, state 0 is a local optimum, and states 1–8 are the high-uncertainty and low-value states.

4 with half probability. If the lower bound has a spurious maximum and policies are directionally uninformed (Ciosek et al., 2019), lower bound policy π_{LB} may be stuck at the junction of regions 3 and 4. This is wasteful and inefficient, causing pessimistic underexploration.

π_{UB}, which is used in optimistic exploration methods (Brafman and Tennenholtz, 2002; Kim et al., 2019; Pathak et al., 2019), can encourage agents to execute overestimated actions and explore potential high-value regions with high uncertainty. However, regions with high and overestimated actions, such as region 2, may have excessive function approximation errors. Alternatively, if highly uncertain regions also have low values (like region 1), overestimation bias might cause an agent to over-explore a low-value region.

Ideally, the exploration policies are located in region 3 to provide better exploration behaviors and keep stable updates. There are two ways to achieve this: (1) enforcing a KL constraint between π_{UB} and π_{LB} (like OAC Ciosek et al., 2019); and (2) balancing $\hat{Q}$ between $\hat{Q}_{LB}$ and $\hat{Q}_{UB}$, and we call it an under-/overestimation trade-offs.

However, in practical applications, Q_π is unknown, and it is not easy to tune to ideal conditions through constant hyperparameters.

4.2. A simple MDP

We show this effect in a simple Markov decision process (MDP), as shown in Figure 2. Any state's optimal policy is the left action. If the agent wants to go to state 9, it must go through states 1–8 with high uncertainty and low values.

In the experiment, we used a discount factor $\gamma = 0.9$; a replay buffer with size 5,000; a Boltzmann policy with *temperature* $= 0.1$; tabular action values with uniform noisy respect to a Uniform distribution $U(-0.1, 0.1)$, initialized with a Uniform distribution $U(-5, 5)$; and a learning rate of 0.01 for all algorithms.

The results in Figure 3 verify our hypotheses in Section 4.1. All algorithms converge, but each has a different convergence speed. $\hat{Q}_{LB}$ underestimates too much and converges to a suboptimal policy in the early learning stage, causing slow convergence. For $\beta = 0.5$ and 1.0, optimistic exploration drives the agent to escape the local optimum and learn faster. However, $\hat{Q}$ overestimates too much for $\beta = 2.0$, significantly impairing the convergence speed of the policy. In addition, no matter what parameter β takes, the agent still over-explores low-value states at different time steps (see Figure 3).

RAC avoids over-exploration in low-value states and is the fastest to converge to the optimal policy. Furthermore, RAC maintains the Q bias close to zero without catastrophic overestimation throughout the learning process, indicating that RAC keeps an outstanding balance between underestimation and overestimation.

5. Realistic Actor-Critic

We present Realistic Actor-Critic (RAC), which can be used in conjunction with the most modern off-policy actor-critic RL algorithms in principle, such as SAC (Haarnoja et al., 2018) and TD3 (Fujimoto et al., 2018). We describe only the SAC version of RAC (RAC-SAC) in the main body for the exposition. The TD3 version of RAC (RAC-TD3) follows the same principles and is fully described in Appendix B.

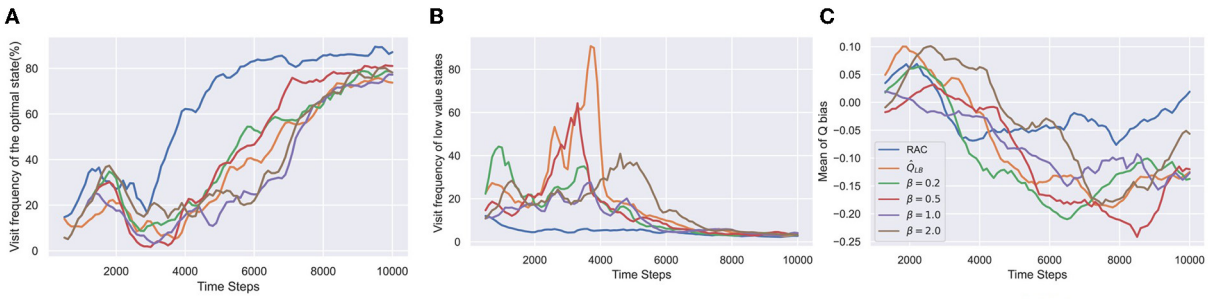

FIGURE 3
Results of the simple MDP. In-target minimization target from REDQ is used as $\hat{Q}_{LB}$. $\hat{Q}_b = mean(\hat{Q}'_{LB}) + \beta standarddeviation(\hat{Q}'_{LB})$ can perform optimistic exploration. β is a key parameter to control value bias. If $\beta = 0$, $\hat{Q}_b$ is equal to $\hat{Q}_{LB}$. As β increases, $\hat{Q}_b$ gradually approaches $\hat{Q}_{UB}$. The horizontal axis indicates the number of time steps. **(A)** Visit frequency of the optimal state is the ratio of the frequency of visiting the optimal state among all termination states. The higher the value, the lower the probability that the agent is stuck in a local optimum. **(B)** Visit frequency of low-value states is the ratio of the visit frequency of low-value state 2–8 and the optimal state 9. The lower the value, the fewer steps the agent wastes in low-value states. This value has been subtracted by 7, as the minimum step size to reach the optimum state is seven. **(C)** Q bias measures the difference between the estimated Q values and true Q values. All results are estimated by the Monte Carlo method and averaged over eight seeds.

5.1. Punished Bellman backup

Punished Bellman backup is a variant of soft Bellman backup (Equation 2). The idea is to maintain an ensemble of N soft Q-functions $Q_{\theta_i}(\mathbf{s}, \mathbf{a})$, where θ_i denotes the parameters of the $i-th$ soft Q-function, which are initialized randomly and independently for inducing an initial diversity in the models (Osband et al., 2016), but updated with the same target.

Given a transition τ_t, punished Bellman backup considers following punished target y:

$$y = r_t + \gamma \mathbb{E}_{\mathbf{a}' \sim \pi_\phi}[\bar{Q}_{\bar{\theta}}(\mathbf{s}', \mathbf{a}') - \beta \hat{s}(Q_{\bar{\theta}}(\mathbf{s}', \mathbf{a}')) - \alpha \log \pi_\phi(\mathbf{a}' \mid \mathbf{s}')], \tag{6}$$

where $\bar{Q}_{\bar{\theta}}(\mathbf{s}, \mathbf{a})$ is the sample mean of the target Q-functions and $\hat{s}(Q_{\bar{\theta}}(\mathbf{s}, \mathbf{a}))$ is the sample standard deviation of target Q-functions with bessel's correction (Warwick and Lininger, 1975). Punished Bellman backup uses $\hat{s}(Q_{\bar{\theta}}(\mathbf{s}, \mathbf{a}))$ as uncertainty estimation to punish value estimation. $\beta \geq 0$ is the weighting of the punishment. Note that we do not propagate gradient through the uncertainty $\hat{s}(Q_{\bar{\theta}}(\mathbf{s}, \mathbf{a}))$.

We write $Q^i_{\mathbf{sa}}$ instead of $Q_{\theta_i}(\mathbf{s}, \mathbf{a})$ and $Q^i_{\mathbf{s}'\mathbf{a}'}$ instead of $Q_{\theta_i}(\mathbf{s}', \mathbf{a}')$ for compactness. Assuming each Q-function has random approximation error $e^i_{\mathbf{sa}}$ (Thrun and Schwartz, 1993; Lan et al., 2020; Chen et al., 2021), which is a random variable belonging to some distribution,

$$Q^i_{\mathbf{sa}} = Q^*_{\mathbf{sa}} + e^i_{\mathbf{sa}}, \tag{7}$$

where $Q^*_{\mathbf{sa}}$ is the ground truth of Q-functions. M is the number of actions applicable at state $\mathbf{s}'$. The estimation bias Z_{MN} for a transition τ_t is defined as

$$\begin{aligned} Z_{MN} &\overset{\text{def}}{=} \left[r + \gamma \max_{\mathbf{a}'}(Q^{mean}_{\mathbf{s}'\mathbf{a}'} - \beta Q^{std}_{\mathbf{s}'\mathbf{a}'})\right] - \left(r + \gamma \max_{\mathbf{a}'} Q^*_{\mathbf{s}'\mathbf{a}'}\right) \\ &= \gamma \left[\max_{\mathbf{a}'}(Q^{mean}_{\mathbf{s}'\mathbf{a}'} - \beta Q^{std}_{\mathbf{s}'\mathbf{a}'}) - \max_{\mathbf{a}'} Q^*_{\mathbf{s}'\mathbf{a}'}\right], \end{aligned} \tag{8}$$

where

$$\begin{aligned} Q^{mean}_{\mathbf{s}'\mathbf{a}'} &\approx \frac{1}{N}\sum_{i=1}^{N} Q^i_{\mathbf{s}'\mathbf{a}'} = \frac{1}{N}\sum_{i=1}^{N}(Q^*_{\mathbf{s}'\mathbf{a}'} + e^i_{\mathbf{s}'\mathbf{a}'}) = Q^*_{\mathbf{s}'\mathbf{a}'} \\ &+ \frac{1}{N}\sum_{i=1}^{N} e^i_{\mathbf{s}'\mathbf{a}'} = Q^*_{\mathbf{s}'\mathbf{a}'} + \bar{e}_{\mathbf{s}'\mathbf{a}'}, \end{aligned} \tag{9}$$

$$\begin{aligned} Q^{std}_{\mathbf{s}'\mathbf{a}'} &\approx \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(Q^i_{\mathbf{s}'\mathbf{a}'} - Q^{mean}_{\mathbf{s}'\mathbf{a}'}\right)^2} \\ &= \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(Q^*_{\mathbf{s}'\mathbf{a}'} + e^i_{\mathbf{s}'\mathbf{a}'} - Q^*_{\mathbf{s}'\mathbf{a}'} + \bar{e}_{\mathbf{s}'\mathbf{a}'}\right)^2} \\ &= \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(e^i_{\mathbf{s}'\mathbf{a}'} - \bar{e}_{\mathbf{s}'\mathbf{a}'}\right)^2} = \hat{s}(e_{\mathbf{s}'\mathbf{a}'}). \end{aligned} \tag{10}$$

Then,

$$Z_{MN} \approx \gamma \left[\max_{\mathbf{a}'}(Q^*_{\mathbf{s}'\mathbf{a}'} + \bar{e}_{\mathbf{s}'\mathbf{a}'} - \beta \hat{s}(e_{\mathbf{s}'\mathbf{a}'})) - \max_{\mathbf{a}'} Q^*_{\mathbf{s}'\mathbf{a}'}\right]. \tag{11}$$

If one could choose $\beta = \frac{\bar{e}_{\mathbf{s}'\mathbf{a}'}}{\hat{s}(e_{\mathbf{s}'\mathbf{a}'})}$, $Q^i_{\mathbf{sa}}$ will be resumed to $Q^*_{\mathbf{sa}}$, then Z_{MN} can be reduced to near 0. However, it's hard to adjust a suitable constant β for various state-action pairs actually. We

Initialize actor network ϕ, N critic networks $\theta_i, i = 1, \ldots, N$, temperature network ψ, empty replay buffer $\mathcal{B}$, target network $\bar{\theta}_i \longleftarrow \theta_i$, for $i = 1, 2, \ldots, N$, uniform distribution U_1 and U_2
for each iteration **do**
  // OPTIMISTIC EXPLORATION
  execute an action $a \sim \pi_\phi(\cdot \mid s, \beta), \beta \sim U_2$.
  Observe reward r_t, new state s'
  Store transition tuple $\mathcal{B} \leftarrow \mathcal{B} \cup \{(s, a, r_t, s')\}$
  for G updates **do**
    // UPDATE CRITICS *via* PUNISHED BELLMAN BACKUP
    Sample random minibatch:
    $\{\tau_j\}_{j=1}^{B} \sim \mathcal{B}$, $\{\beta_m\}_{m=1}^{B} \sim U_1$
    Compute the Q target (Equation 13)
    for $i = 1, \ldots, N$ **do**
      Update θ_i by minimize $\mathcal{L}^{\text{RAC}}_{\text{critic}}$ (Equation 13)
      Update target networks:
      $\bar{\theta}_i \leftarrow \rho\bar{\theta}_i + (1-\rho)\theta_i$
  // UPDATE ACTORS AND TEMPERATURES ACCORDING TO U_1
  Update ϕ by minimize $\mathcal{L}^{\text{RAC-SAC}}_{\text{actor}}$ (Equation 14)
  Update ψ by minimize $\mathcal{L}^{\text{RAC}}_{\text{temp}}$ (Equation 12)

Algorithm 1. RAC: SAC version.

develop vanilla RAC, which uses a constant β Appendix B.3, to research this problem.

For $\beta = 0$, the update is simple average Q-learning which causes overestimation bias (Chen et al., 2021). As β increases, increasing penalties $Q^{std}_{\mathbf{s}'\mathbf{a}'}$ decrease $E[Z_{MN}]$ gradually and encourage Q-functions to transit smoothly from higher bounds to lower bounds.

5.2. Realistic Actor-Critic agent

We demonstrate how to use punished Bellman backup to incorporate various bounds of value approximations into a full agent that maintains diverse policies, each with a different under-/overestimation trade-off. The pseudocode for RAC-SAC is shown in Algorithm 1.

RAC uses UVFA (Schaul et al., 2015) to extend the critic and actor as $Q_{\theta_i}(\mathbf{s}, \mathbf{a}, \beta)$ and $\pi_\phi(\cdot \mid \mathbf{s}', \beta)$, U_1 is a uniform training distribution $\mathcal{U}[0, a]$, a is a positive real number, and $\beta \sim U_1$ that generates various bounds of value approximations.

An independent temperature network α_ψ parameterized by ψ is used to accurately adjust the temperature with respect to β, which can improve the performance of RAC. Then, the objective (Equation 5) becomes:

$$\mathcal{L}^{\text{RAC}}_{\text{temp}}(\psi) = \mathbb{E}_{\mathbf{s}\sim\mathcal{B}, \mathbf{a}\sim\pi_\phi, \beta\sim U_1}[-\alpha_\psi(\beta)\log\pi_\phi(\mathbf{a} \mid \mathbf{s}, \beta) - \alpha_\psi(\beta)\overline{\mathcal{H}}]. \tag{12}$$

The extended Q-ensemble use punished Bellman backup to simultaneously approximate a soft Q-function family:

$$\mathcal{L}^{\text{RAC}}_{\text{critic}}(\theta_i) = \mathbb{E}_{\tau\sim\mathcal{B}, \beta\sim U_1}[\left(Q_{\theta_i}(\mathbf{s}, \mathbf{a}, \beta) - y\right)^2],$$
$$y = r + \gamma\mathbb{E}_{\mathbf{a}'\sim\pi_\phi}[\bar{Q}_{\bar{\theta}}(\mathbf{s}', \mathbf{a}', \beta) - \beta\hat{s}(Q_{\bar{\theta}}(\mathbf{s}', \mathbf{a}', \beta)) - \alpha_\psi(\beta)\log\pi_\phi(\mathbf{a}' \mid \mathbf{s}', \beta)] \tag{13}$$

where $\bar{Q}_{\bar{\theta}}(\mathbf{s}, \mathbf{a}, \beta)$ is the sample mean of target Q-functions and $\hat{s}(Q_{\bar{\theta}}(\mathbf{s}, \mathbf{a}, \beta))$ is the corrected sample standard deviation of target Q-functions.

The extended policy π_ϕ is updated by minimizing the following object:

$$\mathcal{L}^{\text{RAC-SAC}}_{\text{actor}}(\phi) = \mathbb{E}_{\mathbf{s}\sim\mathcal{B}, \beta\sim U_1}[\mathbb{E}_{\mathbf{a}\sim\pi_\phi}[\alpha_\psi(\beta)\log\left(\pi_\phi(\mathbf{a} \mid \mathbf{s}, \beta)\right) - \bar{Q}_\theta(\mathbf{a}, \mathbf{s}, \beta)]], \tag{14}$$

where $\bar{Q}_\theta(\mathbf{a}, \mathbf{s}, \beta)$ is the sample mean of Q-functions.

A larger UTD ratio G improves sample utilization. We find that a smaller replay buffer capacity slightly improves the sample efficiency of RAC in Section 6.5.

Note that we find that applying different samples, which are generated by binary masks from the Bernoulli distribution (Osband et al., 2016; Lee et al., 2021), to train each Q-function will not improve RAC performance in our experiments; therefore, RAC does not apply this method.

5.2.1. RAC circumvents direct adjustment

RAC leaners with a distribution of β instead of a constant β. One could evaluate the policy family to find the best β. We employ a discrete number H of values $\{\beta_i\}_{i=1}^{H}$ (see details in Appendix A.1) to implement a distributed evaluation for computational efficiency and apply the max operator to get best β.

5.2.2. Optimistic exploration

When interacting with the environment, we propose to sample β uniformly from a uniform explore distribution $U_2 = \mathcal{U}[0, b]$, where $b < a$ is a positive real number, to get optimistic exploratory behaviors to avoid pessimistic underexploration (Ciosek et al., 2019).

5.3. How RAC solves the under-/overestimation trade-off

Similar to the idea of NGU (Badia et al., 2020b), RAC decouples exploration and exploitation policies. RAC uses

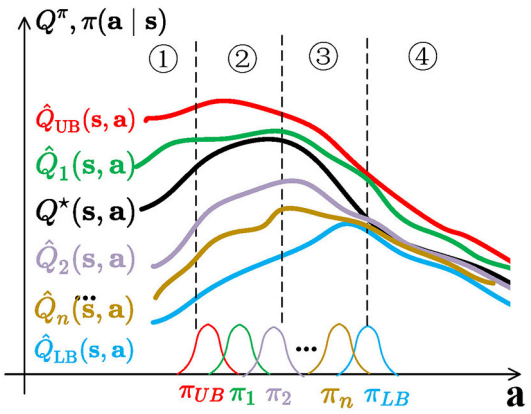

FIGURE 4
Visualization of RAC. The serial numbers in the figure correspond to Section 4.1 and Figure 1. For better illustration, $\hat{Q}$ is discretized. In fact, $\hat{Q}_n$ learned by RAC is infinite and changes continuously. $Q^{\star}(s, a)$ is the optimal Q-function that is unknown. Q-functions are distributed between $\hat{Q}_{UB}$ and $\hat{Q}_{LB}$ and their policies are distributed between π_{UB} and π_{LB}. π_{UB}, π_1, and π_2 are used as exploration policies.

UVFA to simultaneously learn policies with the same neural network, each with different trade-offs between underestimation and overestimation. Using UVFA to learn different degrees of confidence bounds allows us to learn a powerful representation and set of skills that can be quickly transferred to the expected policy. With punished Bellman backup, RAC has a larger number of policies and values that change smoothly, allowing for more efficient training.

This is illustrated in Figure 4. Q-functions that are close to $\hat{Q}_{LB}$ (like $\hat{Q}_n$) control overestimation bias to provide consistency and stable convergence. Exploration policies (such as π_{UB}, π_1, and π_2) are far from the spurious maximum of $\hat{Q}_{LB}$ and $\hat{Q}_n$, and overestimated actions sampled from them located in regions 1, 2, and 3 lead to a quick correction to the critic estimate. All transitions are stored in a shared replay buffer, and all policies benefit from them to escape spurious maximums. Since exploration policies are not symmetric to the mean of π_{LB} and π_n, RAC also avoids directional uninformedness.

Although RAC cannot always keep the exploration policies located in region 3, the policy family avoids all behaviors concentrated in region 1 or 2. Exploration behaviors uniformly distribute in regions 1, 2, and 3, preventing over-exploration in any area.

Moreover, such policies could be quite different from a behavior standpoint and generate varied action sequences to visit unseen state-action pairs following the principle of optimism in the face of uncertainty (Chen et al., 2017; Ciosek et al., 2019; Lee et al., 2021).

6. Experiments

We designed our experiments to answer the following questions:

- Can the Realistic Actor-Critic outperform state-of-the-art algorithms in continuous control tasks?
- Can the Realistic Actor-Critic better balance between value overestimation and underestimation?
- What is the contribution of each technique in the Realistic Actor-Critic?

6.1. Setups

We implement RAC with SAC and TD3 as RAC-SAC and RAC-TD3 (see Appendix B).

The baseline algorithms are REDQ (Chen et al., 2021), MBPO (Janner et al., 2019), SAC (Haarnoja et al., 2018), TD3 (Fujimoto et al., 2018), and TQC (Kuznetsov et al., 2020). All hyperparameters we used for evaluation are the same as those in the original articles. For MBPO (https://github.com/JannerM/mbpo), REDQ (https://github.com/watchernyu/REDQ), TD3 (https://github.com/sfujim/TD3), and TQC (https://github.com/SamsungLabs/tqc_pytorch), we use the authors' code. For SAC, we implement it following the study of Haarnoja et al. (2018), and the results we obtained are similar to previously reported results. TQC20 is a variant of TQC with UTD $G = 20$ for a fair comparison.

We compare baselines on six challenging continuous control tasks (Walker2d, HalfCheetah, Hopper, Swimmer, Ant, and Humanoid) from MuJoCo environments (Todorov et al., 2012) in the OpenAI gym benchmark (Brockman et al., 2016).

The time steps for training instances on Walker2d, Hopper, and Ant are 3×10^5, and 1×10^6 for Humanoid and HalfCheetah. All algorithms explore with a stochastic policy but use a deterministic policy for evaluation similar to those in SAC. We report the mean and standard deviation across eight seeds.

For all algorithms, we use a fully connected network with two hidden layers and 256 units per layer, with Rectified Linear Unit in each layer (Glorot et al., 2011), for both actor and critic. All the parameters are updated by the Adam optimizer (Kingma and Ba, 2014) with a fixed learning rate. All algorithms adopt almost the same NN architecture and hyperparameter.

For all experiments, our learning curves show the total undiscounted return.

Using the Monte Carlo method, we estimate the mean and standard deviation of normalized Q-function bias (Chen et al., 2021) as the main analysis indicators to analyze the

TABLE 1 Performance on OpenAI gym.

	RAC-SAC	RAC-TD3	REDQ	MBPO	TQC20	TD3	SAC	TQC
Humanoid	**11,107 ± 475**	9,321 ± 1,126	5,504 ± 120	5,162 ± 350	7,053 ± 857	7,014 ± 643	7,681 ± 1,118	10,731 ± 1,296
Ant	6,283 ± 549	6,470 ± 165	5,475 ± 890	5,281 ± 699	4,722 ± 567	**6,796 ± 277**	6,433± 332	6,402± 1,371
Walker	**5,860 ± 440**	5,114 ± 489	5,034 ± 711	4,864 ± 488	5,109 ± 696	4,419 ± 1,682	5,249 ± 554	5,821 ± 457
Hopper	3,421 ± 483	3,495 ± 672	**3,563 ± 94**	3,280 ± 455	3,208 ± 538	3,433 ± 321	2,815 ± 585	3,011 ± 866
HalfCheetah	15,717 ± 1,063	15,083 ± 1,113	10,802 ± 1,179	13,477 ± 443	12,123 ± 2,600	14,462 ± 1,982	16,330 ± 323	**17,245 ± 293**
Swimmer	**143 ± 6.8**	71 ± 83	98 ± 31	-	143 ± 9.6	53 ± 8.8	51 ± 4.2	65 ± 5.8

The maximum value for each task is bolded. ± corresponds to a single standard deviation over eight runs. The best results are indicated in bold. Results of SAC, TD3, and TQC are obtained at 6×10^6 time steps for Humanoid and HalfCheetah and 3×10^6 time steps for other environments. Results of RAC, REDQ, and TQC20 are obtained at 1×10^6 time steps for Humanoid and HalfCheetah and 3×10^5 time steps for other environments. Results of MBPO are obtained at 3×10^5 time steps for Ant, Humanoid, and Walker2d, 4×10^5 for HalfCheetah and 1.25×10^5 for Hopper.

TABLE 2 Sample-efficiency comparison.

	RAC-SAC	REDQ	MBPO	TQC	TQC20	REDQ/RAC-SAC	MBPO/RAC-SAC	TQC/RAC-SAC	TQC20/RAC-SAC
Humanoid at 2,000	63 K	109 K	154 K	145 K	147 K	1.73	2.44	2.30	2.33
Humanoid at 5,000	134 K	250 K	295 K	445 K	258 K	1.87	2.20	3.32	1.93
Humanoid at 10,000	552 K	-	-	3,260 K	-	-	-	5.91	-
Ant at 1,000	21 K	28 K	62 K	185 K	42 K	1.33	2.95	8.81	2.00
Ant at 3,000	56 K	56 K	152 K	940 K	79K	1.00	2.71	16.79	1.41
Ant at 6,000	248 K	-	-	3,055 K	-	-	-	12.31	-
Walker at 1,000	27 K	42 K	54 K	110 K	50 K	1.56	2.00	4.07	1.85
Walker at 3,000	53 K	79 K	86 K	270 K	89K	1.49	1.62	10.75	1.68
Walker at 5,000	147 K	272 K	-	960 K	270 K	1.85	-	6.53	1.84

Sample efficiency (Chen et al., 2021; Dorner, 2021) is measured by the ratio of the number of samples collected when RAC and some algorithms reach the specified performance. The last four rows show how many times RAC is more sample efficient than other algorithms in achieving that performance.

value approximation quality (described in Appendix A). The average bias lets us know whether Q_θ is overestimated or underestimated, while standard deviation measures whether Q_θ is overfitting.

Sample efficiency (SE) (Chen et al., 2021; Dorner, 2021) is measured by the ratio of the number of samples collected when RAC and some algorithms reach the specified performance. Hopper is not in the comparison object as the performance of algorithms is almost indistinguishable.

6.2. Comparative evaluation

6.2.1. OpenAI gym

Figure 5 and Table 1 show learning curves and performance comparison. RAC consistently improves the performance of SAC and TD3 across all environments and performs better than other algorithms. In particular, RAC learns significantly faster for Humanoid and has better asymptotic performance for Ant, Walker2d, and HalfCheetah. RAC yields a much smaller variance than SAC and TQC, indicating that the optimistic exploration helps the agents escape from bad local optima.

6.2.2. Sample-efficiency comparison

Table 2 shows the sample-efficiency comparison with baselines. Compared with TQC, RAC-SAC reaches 3,000 and 6,000 for Ant with 16.79x and 12.31x sample efficiency, respectively. RAC-SAC performs 1.5x better than REDQ halfway through training and 1.8x better at the end of training in Walker and Humanoid. They show that a better under-/overestimation trade-off can achieve better sample-efficiency performance than the MuJoCo environments' state-of-the-art algorithms.

6.2.3. Value approximation analysis

Figure 6 presents the results for Ant, Humanoid, and Walker2d.

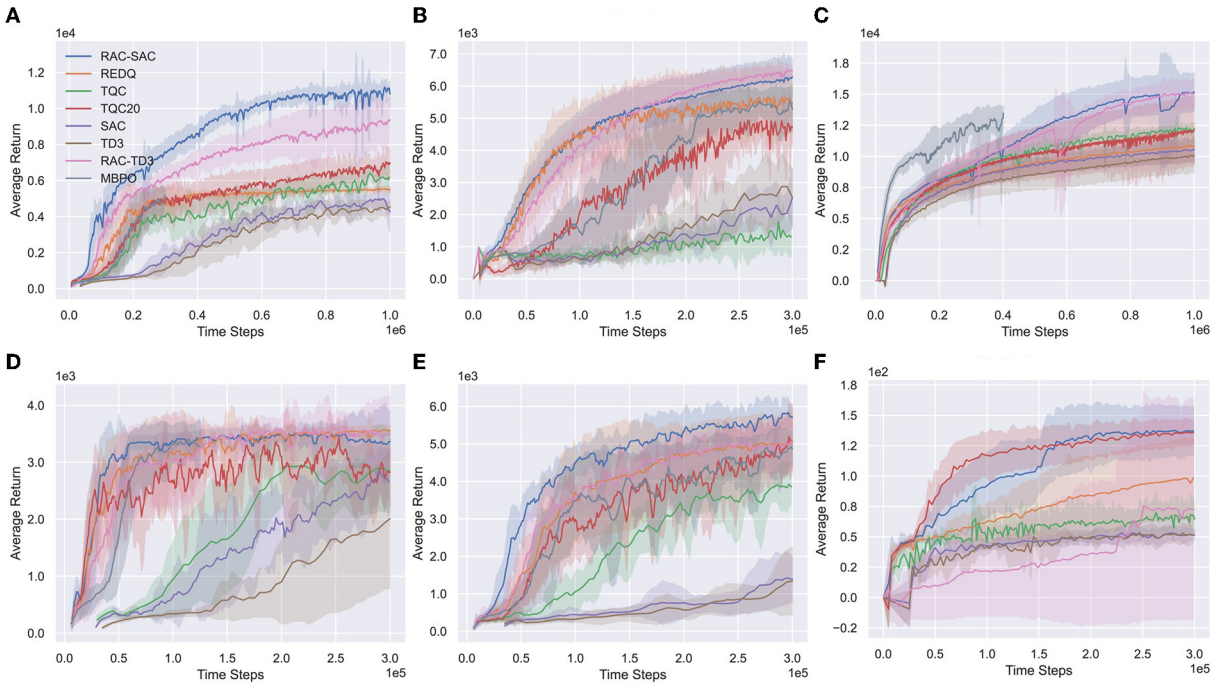

FIGURE 5
Learning curves on six Mujoco environments. The horizontal axis indicates the number of time steps. The vertical axis shows the average undiscounted return. The shaded areas denote one standard deviation over eight runs. **(A)** Humanold, **(B)** Ant, **(C)** HalfCheetah, **(D)** Hopper, **(E)** Walker2d, and **(F)** Swimmer.

In Ant and Walker2d, TQC20 has a high normalized mean of bias, indicating that TQC20 prevents catastrophic overestimation failure accumulation. TQC20 also has a high normalized standard deviation of bias, indicating that the bias is highly non-uniform, which can be detrimental. Since distributional RL is prone to overfitting with few samples, it may not be appropriate to use a high UTD ratio for TQC. In Humanoid, which has a high-dimensional state, overfitting still exists but has been alleviated.

Relative to TQC and TQC20, REDQ and RAC-SAC have a very low normalized standard deviation of bias for most of the training, indicating the bias across different state-action pairs is about the same. Thus, the Q-estimation of REDQ is too conservative in Humanoid, and the large negative bias makes REDQ trapped in a bad locally optimal policy, suffering from pessimistic underexploration. For Ant and Walker2d, although this poor exploration does not harm the performance of the policy, it still slows down convergence speed compared to RAC.

Relative to REDQ, RAC-SAC keeps the Q bias nearly zero without overestimation accumulation; this benign overestimation bias significantly improves performance. RAC-SAC strikes a good balance between overestimation bias (good performance without being trapped in a bad local optimum) and underestimation bias (slight overestimation bias and consistently small standard deviation of bias).

6.3. Why Humanoid is hard for most baselines?

Figure 7 visualizes the performance with respect to various value confidence bounds. Humanoid is extremely sensitive to the value bias. The huge state-action space of Humanoid leads to a large approximation error of the value function with small samples. The approximate lower bound inevitably has spurious maxima, while a small overestimated bias can seriously destabilize updates. It is hard to choose appropriate confidence bound for Humanoid by tuning the hyperparameters, resulting in a difficult under-/overestimation trade-off.

Algorithms (like REDQ) that rely on constant hyperparameters to control the value bias have to conservatively introduce a large underestimation error (Figure 6) to stabilize updates, leading the policy to fall into pessimistic underexploration. In contrast, other algorithms (such as TQC20) plagued by overestimation and overfitting require more samples.

Compared to Humanoid, the state-action space of other environments is much smaller. The approximate Q-functions can easily fit the true Q values accurately, significantly reducing the possibility of spurious maxima. Therefore, optimistic exploration may not be a required component for these environments. So, we can see that they are not very sensitive to various value confidence bounds from Figure 7.

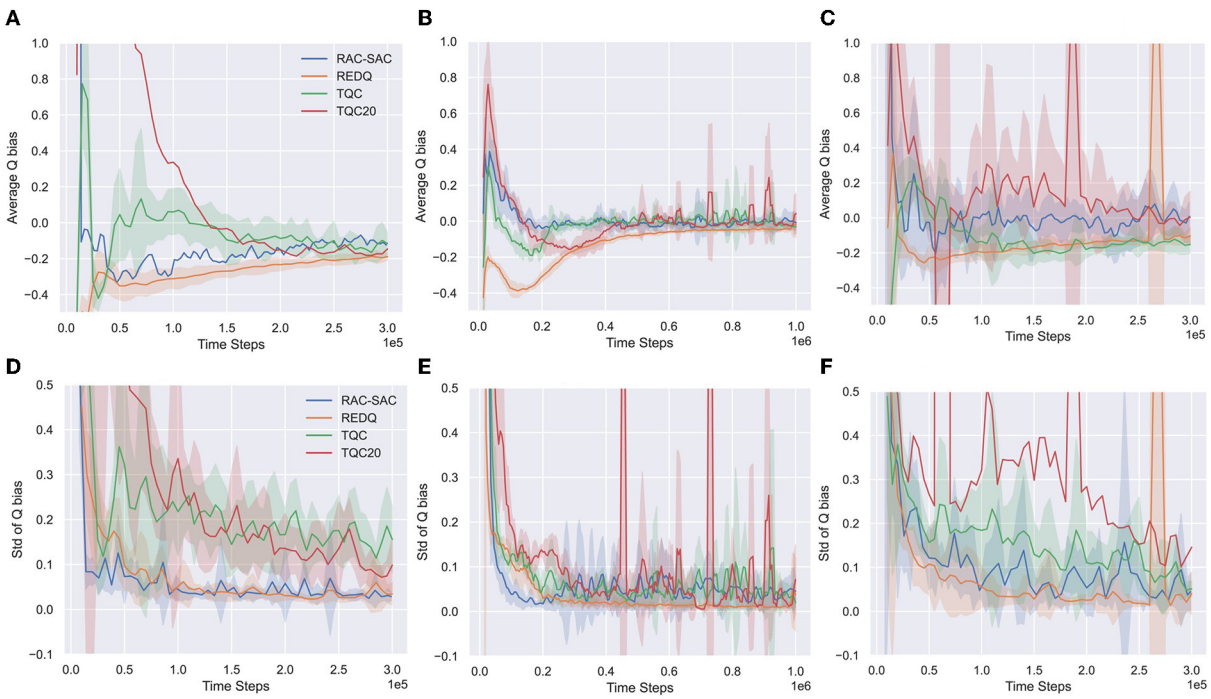

FIGURE 6
Estimated mean and standard deviation of normalized Q bias of RAC-SAC, REDQ, TQC, and TQC20 for Ant and Humanoid with Monte Carlo method. **(A)** Q bias of Ant, **(B)** Q bias of Humanoid, **(C)** Q bias of Walker2d, **(D)** Q standard deviation of Ant, **(E)** Q standard deviation of Humanoid, and **(F)** Q standard deviation of Walker2d.

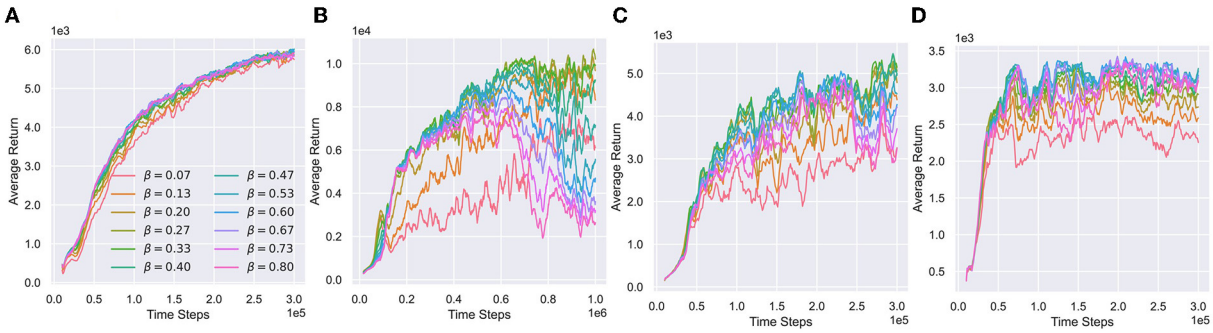

FIGURE 7
Performance of various value confidence bounds with respect to different β. **(A–D)** Performance respect to different β in Ant, Humanoid, Walker2d, and Hopper. We visualize different β belonging to training distribution $U_1 = \mathcal{U}[0, a]$ during training processes.

An underestimated value is enough to guide the policy to learn stably.

6.4. Variants of RAC

We evaluate the performance contributions of ingredients of RAC (punished Bellman backup, policy family, optimistic exploration, independent temperature network, and learning rate warm-up) on a subset of four environments (see Figure 8).

6.4.1. Punished Bellman backup

When using the in-target minimization instead of punished Bellman backup, RAC is stable, but the performance is significantly worse in Humanoid. Punished Bellman backup provides more finer-grained bias control than in-target minimization, reducing the difficulty of learning representations. Compared with other environments, Humanoid has stronger requirements for state representation learning (Chen et al., 2021). Thus, punished Bellman backup far outperforms in-target minimization in Humanoid and is almost the same in other environments.

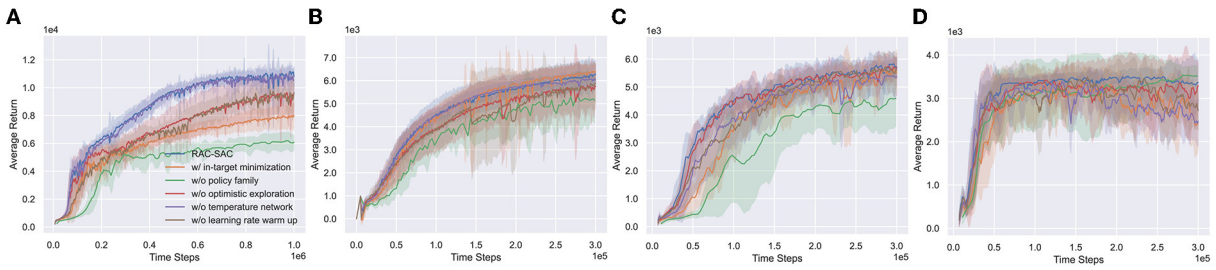

FIGURE 8
Performance of RAC and its variants. **(A–D)** Humanoid, Ant, Walker2d, and Hopper. The in-target minimization version of RAC is shown in Appendix B.4. RAC without policy family is named Vanilla RAC (see Appendix B.3 for more details. In this case, OAC (Ciosek et al., 2019) is used as optimistic exploration method).

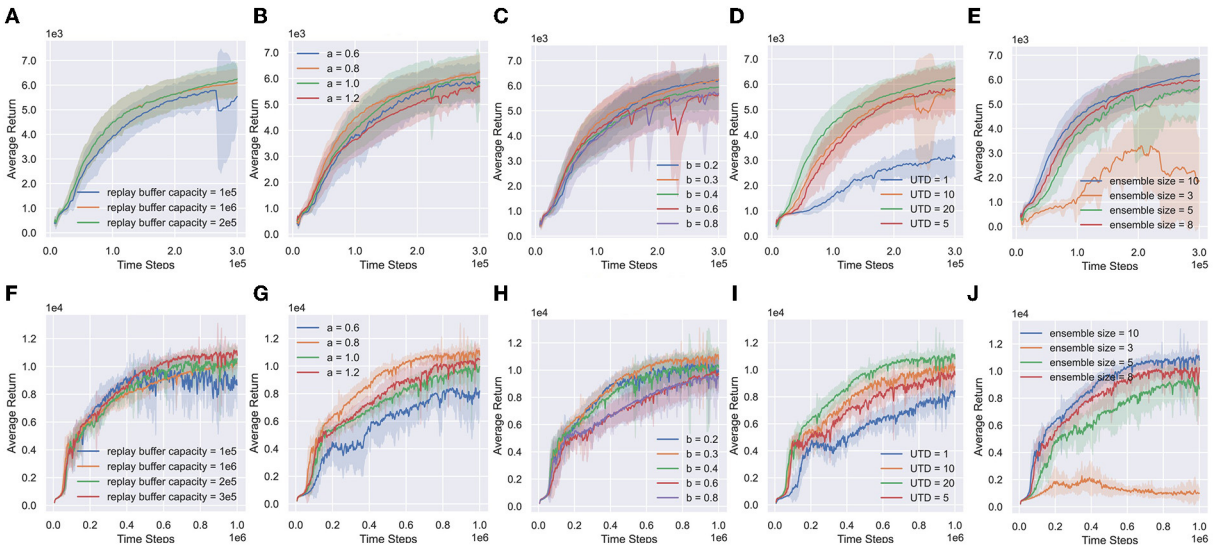

FIGURE 9
Hyperparameter ablations of RAC. **(A–E)** Replay buffer capacity, right side of exploitation distribution (a), right side of exploration distribution (b), the UTD ratio and the ensemble size for ant. **(F–J)** Replay buffer capacity, right side of exploitation distribution (a), right side of exploration distribution (b), the UTD ratio and the Ensemble size for Humanoid.

6.4.2. Policy family

The policy family is paramount to performance. This is consistent with Section 5.3's conjecture. Even with OAC, an agent can only converge to a local optimum without the policy family in Humanoid, indicating that a single optimistic exploration method cannot solve the pessimistic underexploration well. In addition, the convergence speed of the policy has decreased in Walker2d and Ant.

6.4.3. Optimistic exploration

Experimental results support the point in Section 6.3. Optimistic exploration can help the agent escape from local optima in Humanoid. However, in simple environments (like Ant, Walker2d, and Hopper), optimistic exploration has little impact on performance.

6.4.4. Independent temperature network

Except for Walker2d, the independent temperature network has little effect on RAC performance. The learned temperatures are shown in Appendix C. In practice, we find that the independent temperature network can control the entropy of the policy more quickly and stably.

6.4.5. Learning rate warm-up

A high UTD ratio can easily lead to an excessive accumulation of overestimation errors in the early stage of learning. The learning rate warm-up can alleviate this problem and stabilize the learning process. Without the learning rate warm-up, RAC learns slower at the beginning of the training process.

6.5. Hyperparameter ablations

RAC introduces some hyperparameters: (1) replay buffer capacity; (2) right side of exploitation distribution U_1 (a); (3) right side of exploration distribution U_2 (b); (4) UTD ratio G in Algorithm 1; and (5) Ensemble size. Figure 9 shows the numerical results.

Replay buffer capacity (Figures 9A, F). RAC can benefit from a smaller capacity but will be hurt when the capacity is excessively small.

The right side of U_1 (a) (Figures 9B, G). a is a key hyperparameter of RAC. Because a controls the underestimation bias of RAC, which determines the lower bound of Q-functions. The learning process becomes stable with a increasing. However, if a is too large, it will reduce the learning opportunity of optimistic policies, thereby reducing the learning efficiency.

The right side of U_2 (b) (Figures 9C, H). Exploration policies become more conservative with b increasing, and the performance of RAC gradually declines. The increasing standard deviation means that more and more agents fall into local-optimal policies. However, if b is too small, policies may over-explore the overestimated state, resulting in a decrease in learning efficiency.

The ensemble size (Figures 9E, J) and the UTD ratio (Figures 9D, I). RAC appears to benefit greatly from the ensemble size and UTD ratio. When the ensemble size and UTD ratio are increased, we generally get a more stable average bias, a lower standard deviation of bias, and stronger performance.

7. Conclusion

In this study, we empirically discussed under-/overestimation trade-off on improving the sample efficiency in DRL and proposed the Realistic Actor-Critic (RAC), which learns together values and policies with different trade-offs between underestimation and overestimation in the same network. This study proposed Punished Bellman backup that provides fine-granular estimation bias control to make value approximation smoothly shift between upper bounds and lower bounds. This study also discussed the role of the various components of RAC. Experiments show advantageous properties of RAC: low-value approximation error and brilliant sample efficiency. Furthermore, continuous control benchmarks suggest that RAC consistently improves performances and sample efficiency of existing off-policy RL algorithms, such as SAC and TD3. It is of great significance for promoting reinforcement learning in the robot control domain.

Our results suggest that directly incorporating uncertainty to value functions and learning a powerful policy family can provide a promising avenue for improved sample efficiency and performance. Further exploration of ensemble methods, including high-level policies or more rich policy classes, is an exciting avenue for future work.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

SL implemented the code and drafted the manuscript. QT assisted in implementing the code and discussed the manuscript. YP assisted in implementing the code and discussed the manuscript. XM guided the research and discussed the results. GW guided the research, implemented parts of the code, and revised the manuscript. All authors contributed to the article and approved the submitted version.

Funding

This study was funded by the National Natural Science Foundation of Heilongjiang Province (Grant No. YQ2020E028), the National Natural Science Foundation of China (Grant No. 51779059), and in part by the Research Fund from the Science and Technology on Underwater Vehicle Technology under Grant No. 2021-SYSJJ-LB06909.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2022.1081242/full#supplementary-material

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., et al. (2021). A review of uncertainty quantification in deep learning: techniques, applications and challenges. *Inf. Fusion*. 76, 243–297. doi: 10.1016/j.inffus.2021.05.008

Amos, B., Dinh, L., Cabi, S., Rothörl, T., Colmenarejo, S. G., Muldal, A., et al. (2018). Learning awareness models. *arXiv preprint arXiv:1804.06318*. doi: 10.48550/arXiv.1804.06318

Anschel, O., Baram, N., and Shimkin, N. (2017). "Averaged-DQN: variance reduction and stabilization for deep reinforcement learning," in *International Conference on Machine Learning* (PMLR), 176–185. Available online at: http://proceedings.mlr.press/v70/anschel17a/anschel17a.pdf

Badia, A. P., Piot, B., Kapturowski, S., Sprechmann, P., Vitvitskyi, A., Guo, Z. D., et al. (2020a). "Agent57: outperforming the atari human benchmark," in *International Conference on Machine Learning* (PMLR), 507–517. Available online at: http://proceedings.mlr.press/v119/badia20a/badia20a.pdf

Badia, A. P., Sprechmann, P., Vitvitskyi, A., Guo, D., Piot, B., Kapturowski, S., et al. (2020b). Never give up: learning directed exploration strategies. *arXiv preprint arXiv:2002.06038*. doi: 10.48550/arXiv.2002.06038

Brafman, R. I., and Tennenholtz, M. (2002). R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *J. Mach. Learn. Res.* 3, 213–231. doi: 10.1162/153244303765208377

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., et al. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*. doi: 10.48550/arXiv.1606.01540

Chen, G., and Peng, Y. (2019). Off-policy actor-critic in an ensemble: achieving maximum general entropy and effective environment exploration in deep reinforcement learning. *arXiv preprint arXiv:1902.05551*. doi: 10.48550/arXiv.1902.05551

Chen, L., Jiang, Z., Cheng, L., Knoll, A. C., and Zhou, M. (2022). Deep reinforcement learning based trajectory planning under uncertain constraints. *Front. Neurorobot.* 16, 883562. doi: 10.3389/fnbot.2022.883562

Chen, R. Y., Sidor, S., Abbeel, P., and Schulman, J. (2017). Ucb exploration *via* q-ensembles. *arXiv preprint arXiv:1706.01502*. doi: 10.48550/arXiv.1706.01502

Chen, X., Wang, C., Zhou, Z., and Ross, K. (2021). Randomized ensembled double q-learning: Learning fast without a model. *arXiv preprint arXiv:2101.05982*. doi: 10.48550/arXiv.2101.05982

Ciosek, K., Vuong, Q., Loftin, R., and Hofmann, K. (2019). "Better exploration with optimistic actor critic," in *Advances in Neural Information Processing Systems 32*. Available online at: https://papers.nips.cc/paper/2019/file/a34bacf839b923770b2c360eefa26748-Paper.pdf

Dorner, F. E. (2021). Measuring progress in deep reinforcement learning sample efficiency. *arXiv preprint arXiv:2102.04881*. doi: 10.48550/arXiv.2102.04881

Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., et al. (2020). An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*. doi: 10.48550/arXiv.2003.11881

Dusenberry, M., Jerfel, G., Wen, Y., Ma, Y., Snoek, J., Heller, K., et al. (2020). "Efficient and scalable bayesian neural nets with rank-1 factors," in *International Conference on Machine Learning* (PMLR), 2782–2792. Available online at: http://proceedings.mlr.press/v119/dusenberry20a/dusenberry20a.pdf

Fujimoto, S., Hoof, H., and Meger, D. (2018). "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning* (PMLR), 1587–1596. Available online at: http://proceedings.mlr.press/v80/fujimoto18a/fujimoto18a.pdf

Fukuda, T. (2020). Cyborg and bionic systems: Signposting the future. *Cyborg Bionic Syst*. 2020, 1310389. doi: 10.34133/2020/1310389

Glorot, X., Bordes, A., and Bengio, Y. (2011). "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (JMLR Workshop and Conference Proceedings), 315–323. Available online at: http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf

Goyal, A., Sodhani, S., Binas, J., Peng, X. B., Levine, S., and Bengio, Y. (2019). Reinforcement learning with competitive ensembles of information-constrained primitives. *arXiv preprint arXiv:1906.10667*. doi: 10.48550/arXiv.1906.10667

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., et al. (2018). Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*. doi: 10.48550/arXiv.1812.05905

Havasi, M., Jenatton, R., Fort, S., Liu, J. Z., Snoek, J., Lakshminarayanan, B., et al. (2020). Training independent subnetworks for robust prediction. *arXiv preprint arXiv:2010.06610*. doi: 10.48550/arXiv.2010.06610

He, Q., Gong, C., Qu, Y., Chen, X., Hou, X., and Liu, Y. (2021). MEPG: a minimalist ensemble policy gradient framework for deep reinforcement learning. *arXiv preprint arXiv:2109.10552*. doi: 10.48550/arXiv.2109.10552

Janner, M., Fu, J., Zhang, M., and Levine, S. (2019). "When to trust your model: Model-based policy optimization," in *Advances in Neural Information Processing Systems 32*. Available online at: https://dl.acm.org/doi/10.5555/3454287.3455409

Jung, W., Park, G., and Sung, Y. (2020). Population-guided parallel policy search for reinforcement learning. *arXiv preprint arXiv:2001.02907*. doi: 10.48550/arXiv.2001.02907

Kalweit, G., and Boedecker, J. (2017). "Uncertainty-driven imagination for continuous deep reinforcement learning," in *Conference on Robot Learning* (PMLR), 195–206. Available online at: http://proceedings.mlr.press/v78/kalweit17a/kalweit17a.pdf

Karimpanal, T. G., and Bouffanais, R. (2018). Experience replay using transition sequences. *Front. Neurorobot.* 12, 32. doi: 10.3389/fnbot.2018.00032

Kim, H., Kim, J., Jeong, Y., Levine, S., and Song, H. O. (2019). "EMI: exploration with mutual information," in *International Conference on Machine Learning* (PMLR), 3360–3369. Available online at: https://arxiv.org/pdf/1810.01176.pdf

Kingma, D. P., and Ba, J. (2014). Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. doi: 10.48550/arXiv.1412.6980

Kumar, A., Gupta, A., and Levine, S. (2020). Discor: Corrective feedback in reinforcement learning *via* distribution correction. *Adv. Neural Inf. Process. Syst.* 33, 18560–18572. doi: 10.48550/arXiv.2003.07305

Kuznetsov, A., Grishin, A., Tsypin, A., Ashukha, A., and Vetrov, D. (2021). Automating control of overestimation bias for continuous reinforcement learning. *arXiv preprint arXiv:2110.13523*. doi: 10.48550/arXiv.2110.13523 Available online at: https://arxiv.org/pdf/2110.13523.pdf

Kuznetsov, A., Shvechikov, P., Grishin, A., and Vetrov, D. (2020). "Controlling overestimation bias with truncated mixture of continuous distributional quantile critics," in *International Conference on Machine Learning* (PMLR), 5556–5566.

Lan, Q., Pan, Y., Fyshe, A., and White, M. (2020). Maxmin q-learning: controlling the estimation bias of q-learning. *arXiv preprint arXiv:2002.06487*. doi: 10.48550/arXiv.2002.06487

Lee, K., Laskin, M., Srinivas, A., and Abbeel, P. (2021). "Sunrise: a simple unified framework for ensemble learning in deep reinforcement learning," in *International Conference on Machine Learning* (PMLR), 6131–6141. Available online at: http://proceedings.mlr.press/v139/lee21g/lee21g.pdf

Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., and Quillen, D. (2018). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int. J. Rob. Res.* 37, 421–436. doi: 10.1177/0278364917710318

Lyle, C., Rowland, M., Ostrovski, G., and Dabney, W. (2021). "On the effect of auxiliary tasks on representation dynamics," in *International Conference on Artificial Intelligence and Statistics* (PMLR), 1–9. Available online at: http://proceedings.mlr.press/v130/lyle21a/lyle21a.pdf

Namiki, A., and Yokosawa, S. (2021). Origami folding by multifingered hands with motion primitives. *Cyborg Bionic Syst.* 2021, 9851834. doi: 10.34133/2021/9851834

Osband, I., Blundell, C., Pritzel, A., and Van Roy, B. (2016). "Deep exploration *via* bootstrapped DQN," in *Advances in Neural Information Processing Systems 29*. Available online at: https://papers.nips.cc/paper/2016/file/8d8818c8e140c64c743113f563cf750f-Paper.pdf

Parker-Holder, J., Pacchiano, A., Choromanski, K. M., and Roberts, S. J. (2020). Effective diversity in population based reinforcement learning. *Adv. Neural Inf. Process. Syst.* 33, 18050–18062. doi: 10.48550/arXiv.2002.00632

Pathak, D., Gandhi, D., and Gupta, A. (2019). "Self-supervised exploration *via* disagreement," in *International Conference on Machine Learning* (PMLR), 5062–5071. Available online at: http://proceedings.mlr.press/v97/pathak19a/pathak19a.pdf

Peer, O., Tessler, C., Merlis, N., and Meir, R. (2021). Ensemble bootstrapping for q-learning. *arXiv preprint arXiv:2103.00445*. doi: 10.48550/arXiv.2103.00445

Pendrith, M. D., and Ryan, M. R. (1997). *Estimator variance in reinforcement learning: Theoretical problems and practical solutions*. University of New South Wales, School of Computer Science and Engineering.

Rashid, T., Peng, B., Böhmer, W., and Whiteson, S. (2020). "Optimistic exploration even with a pessimistic initialization," in *International Conference on Learning Representations (ICLR)*. doi: 10.48550/arXiv.2002.12174

Saphal, R., Ravindran, B., Mudigere, D., Avancha, S., and Kaul, B. (2020). SEERL: sample efficient ensemble reinforcement learning. *arXiv preprint arXiv:2001.05209*. doi: 10.48550/arXiv.2001.05209

Schaul, T., Horgan, D., Gregor, K., and Silver, D. (2015). "Universal value function approximators," in *International Conference on Machine Learning* (PMLR), 1312–1320. Available online at: http://proceedings.mlr.press/v37/schaul15.pdf

Sutton, R. S., and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press. Available online at: http://www.incompleteideas.net/sutton/book/first/Chap1PrePub.pdf

Thrun, S., and Schwartz, A. (1993). "Issues in using function approximation for reinforcement learning," in *Proceedings of the Fourth Connectionist Models Summer School* (Hillsdale, NJ), 255–263.

Todorov, E., Erez, T., and Tassa, Y. (2012). "MuJoCo: a physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Vilamoura-Algarve: IEEE), 5026–5033.

Van Hasselt, H., Guez, A., and Silver, D. (2016). "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30*. Available online at: https://ojs.aaai.org/index.php/AAAI/article/download/10295/10154

Warwick, D. P., and Lininger, C. A. (1975). *The Sample Survey: Theory and Practice*. McGraw-Hill. Available online at: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=The+Sample+Survey%3A+Theory+and+Practice&btnG=

Wen, Y., Tran, D., and Ba, J. (2020). Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*. doi: 10.48550/arXiv.2002.06715

Wenzel, F., Snoek, J., Tran, D., and Jenatton, R. (2020). Hyperparameter ensembles for robustness and uncertainty quantification. *Adv. Neural Inf. Process. Syst.* 33, 6514–6527. doi: 10.48550/arXiv.2006.13570

Wu, Y., Zhai, S., Srivastava, N., Susskind, J., Zhang, J., Salakhutdinov, R., et al. (2021). Uncertainty weighted actor-critic for offline reinforcement learning. *arXiv preprint arXiv:2105.08140*. doi: 10.48550/arXiv.2105.08140

Yu, Y. (2018). "Towards sample efficient reinforcement learning," in *IJCAI*, 5739–5743. Available online at: https://www.ijcai.org/proceedings/2018/0820.pdf

Zheng, Z., Yuan, C., Lin, Z., and Cheng, Y. (2018). "Self-adaptive double bootstrapped DDPG," in *International Joint Conference on Artificial Intelligence*. Available online at: https://www.ijcai.org/proceedings/2018/0444.pdf

Ziebart, B. D. (2010). *Modeling Purposeful Adaptive Behavior With the Principle of Maximum Causal Entropy*. Carnegie Mellon University. Available online at: http://reports-archive.adm.cs.cmu.edu/anon/anon/home/ftp/usr/ftp/ml2010/CMU-ML-10-110.pdf