Exploiting Reward Shifting in Value-Based Deep RL

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this work, we study the simple yet universally applicable case of reward shaping in value-based Deep Reinforcement Learning (DRL). We show that reward shifting in the form of the linear transformation is equivalent to changing the initialization of the Q-function in function approximation. Based on such an equivalence, we bring the key insight that a positive reward shifting leads to conservative exploitation, while a negative reward shifting leads to curiosity-driven exploration. Accordingly, conservative exploitation improves offline RL value estimation, and optimistic value estimation benefits the exploration of online RL. We validate our insight on a range of RL tasks and show its improvement over baselines: (1) In offline RL, the conservative exploitation leads to improved performance based on off-the-shelf algorithms; (2) In online continuous control, multiple value functions with different shifting constants can be used to tackle the exploration-exploitation dilemma for better sample efficiency; (3) In discrete exploration tasks, a negative reward shifting brings an improvement over the previous curiosity-based exploration method.

1 Introduction

While reward shaping is a well-established practice in reinforcement learning and has a long-standing history [1, 2], specifying a certain reward to incentivize the learning agent requires domain knowledge and thorough understanding of the task [3-6]. Even with careful design and tuning, learning with a shaped reward that intends to accelerate learning may on the contrary hinder the learning performance by incurring the sub-optimal behaviors of the agent [7, 8]. Although Ng et al. [9] theoretically points the optimal policy remains unchanged under a special form of reward transformation, and in the later work of Wiewiora et al. [10] a framework is proposed to guide policies with prior knowledge under tabular setting, the investigation of how it accommodates recent Deep Reinforcement Learning (DRL) algorithms remains much less explored.

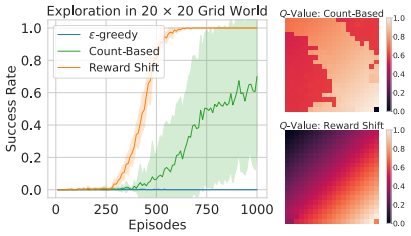

Figure 1: Our work is inspired by the observation that reward shifting remarkably helps exploration and outperforms count-based exploration in Q-learning (left). Reward shifting does not change the primal optimal Q-value landscape, and is able to learn a near-optimal Q-value (right). While count-based exploration suffers from the curse of dimensionality, reward shifting can be seamlessly applied to high dimensional tasks including continuous control.

In this work, we study a special linear transformation, which is the simplest form of reward shaping, in value-based DRL [11-14]. We start with understanding how such a specific kind of reward shaping works in value-based DRL function approximations and show that reward shifting is equivalent to different Q-value initialization, extending previous discovery of [10] to the function approximation setting. Figure 1 showcases reward shifting benefits exploration in a maze game and outperforms count-based and ϵ-greedy exploration and learns near optimal value function.

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

Based on such an equivalence, we introduce the key insight of this work:

A positive reward shifting leads to conservative exploitation, while a negative reward shifting leads to curiosity-driven exploration.

We demonstrate the application of such an insight on three Deep-RL scenarios: (1) for offline RL, we show that conservative exploitation can lead to improved learning performance based on off-the-shelf algorithms; (2) for online RL setting, we show multiple value functions with different reward shifting constants can be used as a trade-off between exploration and exploitation, thus improving learning efficiency; (3) we introduce a simple yet crucial improvement over a prevailing curiosity-based exploration method, the Random Network Distillation [15], making it compatible with value-based DRL algorithms. The experiments on both continuous and discrete action space control tasks shows our method brings the substantial improvement over previous baselines.

Our contributions can be summarized as follows:

1. We introduce the key insight that reward shifting is equivalent to diversified Q-value network initialization in value-based deep-RL, which can be applied to curiosity-driven exploration and conservative exploitation;
2. Practically, we instantiate the key insight to three different scenarios, namely the offline conservative exploitation, sample-efficient continuous control, and curiosity-driven exploration, to contrast the generality of reward shifting;
3. Empirically, we demonstrate the effectiveness of the proposed method integrated with off-the-shelf baselines on both continuous and discrete control tasks.

2 Preliminaries and Related Work

2.1 Online RL

We follow a standard MDP formulation in the online RL setting, i.e., $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho_0, \gamma, T\}$, where $\mathcal{S} \subset \mathbb{R}^d$ denotes the d-dim state space, $\mathcal{A}$ is the action space (note for discrete action space $|\mathcal{A}| < \infty$ and for continuous control $|\mathcal{A}| = \infty$), we consider a deterministic transition dynamics $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ and deterministic reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$. ρ_0 denotes the initial state distribution, i.e., $\rho_0 = p(s_0)$. γ is the discount factor and T is the episodic decision horizon. Online RL considers the problem of learning a policy $\pi \in \Pi : \mathcal{S} \mapsto \Delta(\mathcal{A})$ (or $\pi \in \Pi : \mathcal{S} \mapsto \mathcal{A}$ with a deterministic policy class), such that the expected cumulative reward in the Markov decision process is maximized, i.e., $\pi = \arg\max_\pi \mathbb{E}_{a_t \sim \pi, s_{t+1} \sim \mathcal{T}, s_0 \sim \rho_0} \sum_{t=0}^{T} \gamma^t r_t(s_t, a_t)$, In the online RL setting, an agent normally learns through trials and errors [11], either with an on-policy paradigm [16-18] or an off-policy manner [12-14, 19, 20]. In this work, we focus on the off-policy value-based methods which are in general more sample efficient. Specifically, our discussions assume the policy learning is based on a learned Q-value function, that approximates the cumulative reward an agent can gain in the following part of an episode. The Q-value function is defined as $Q(s_t, a_t) = \mathbb{E}_{\pi, \mathcal{T}} \sum_{\tau=t}^{T} \gamma^t r(s_\tau, a_\tau)$, and can be estimated by propagating the Bellman operator $\mathbb{B}Q(s, a) = r(s, a) + \gamma \mathbb{E} Q(s', a')$. For value-based methods, the (soft-)optimal policy is then produced by

$$\pi^*_\alpha(a|s) = \frac{\exp \frac{1}{\alpha} Q^*(s, a)}{\sum_{a'} \exp \frac{1}{\alpha} Q^*(s, a')}, \tag{1}$$

where Q^* is optimal Q-value function. We can also set the temperature parameter close as 0 to have the deterministic policy class. Simplifying the notion we have $\pi^*(s) = \arg\max_a Q^*(s, a)$. Algorithms like DPG [21] can be used to address the intractable analytical argmax issue arises in continuous action space. We choose to develop our work on top of prevailing baseline algorithms of DQN [13], BCQ [22], CQL [23] and TD3 [14] as a minimal example to isolate the source of gains. It should be straightforward to extend the method on top of other learning algorithms.

2.2 Exploration and the Curiosity-Driven Methods

One of the most important issues in online RL is the exploration-exploitation dilemma [11] that the agent must learn to exploit its accumulated knowledge on the task while exploring new states and actions. Previous works address the exploration problem from various perspectives: for discrete

Table 1: Reward shifting is flexible to be plugged into both online and offline RL algorithms to guarantee conservative exploitation or pursue optimistic exploration. It covers both discrete and continuous control tasks, with only a little additional computational expense. Moreover, the optimal policy learned with shifted reward is not biased.

Covered Topics	Related Work	Plug-in	Online	Offline	Discrete	Continuous	Unbiased	Examples
Exploration	Curiosity	✓	✓	·	✓	·	·	Burda et al. [15]
	Ensemble	·	✓	·	✓	·	✓	Osband et al. [24]
	Initialization	·	✓	·	✓	·	✓	Rashid et al. [25]
	Optimism	·	✓	·	·	✓	·	Ciosek et al. [26]
Exploitation	Conservatism	·	·	✓	✓	✓	·	Bharadhwaj et al. [23]
	Policy Constraints	✓	·	✓	✓	✓	✓	Fujimoto et al. [22]
	Reward Shifting	✓	✓	✓	✓	✓	✓	Ours

action space tasks, count-based methods like [27–29] are proposed to motivate the revisiting of under-explored state-action pairs. Specifically, Choshen et al. [30] extended the idea into general settings by constructing an additional MDP for Exploration-value estimation, as a generalized counter for count-based exploration. To boost exploration in continuous state tasks, curiosity-driven methods are investigated by [31–33, 15], where variety of intrinsic rewards are designed as supplementary to the primal task reward for better exploration. Self-imitate approaches like Oh et al. [34], Ecoffet et al. [35], Sun et al. [36] repeat success trajectories but require extra assumptions on the environment. The work of DIAYN and DADS [37, 38] show that various skills can be developed even without the primal extrinsic reward. For continuous control tasks, OAC [26] improves the SAC [20] with informative action space noise based on the optimism in face of uncertainty (OFU) [39–42]. GAC [43] addresses the exploration issue with a richer functional class for the policy.

In the work of Rashid et al. [25], the problematic pessimistic initialization is addressed for better exploration, yet the work focuses on specific settings of tabular and discrete control exploration. In the work of Osband et al. [44, 24], ensemble models with diverse initialization and randomized priors are used to resemble the insight of bootstrap sampling and facilitate better value estimation, yet those methods are only applicable to discrete control tasks. Noted that although the reward shifting can be regarded as a special case of these random priors, it can be distinguished by not changing the optimal Q-value, and its flexibility to be plugged in to both continuous and discrete control algorithms.

Random Network Distillation (RND) [15] defines the intrinsic reward as the output difference between a fixed neural network ϕ_1 and a trainable network ϕ_2 given state-actions as the inputs e.g.,

$$r_{\text{int}}(s, a) = |\phi_2(s, a) - \phi_1(s, a)|, \tag{2}$$

when outputs of both networks are activated by a sigmoid function, and ϕ_2 is optimized to approximate ϕ_1 for the visited (s, a) pairs[1]. Hence, the value of $r_{\text{int}}(s, a)$ will decay to 0 when such state-action pairs are visited frequently but remain high for seldom visited pairs.

In this work, we show that exploratory behavior can be achieved simply by shifting the reward function with a constant. Thus our method is orthogonal to those previous approaches in the sense that our intrinsic exploration behavior is motivated by high function approximation error in the under-explored state-action pairs. We demonstrate such an insight by showing that RND in its original design is not suitable for value-based methods in developing exploratory behaviors, but integrating RND with a shifted reward function remarkably improves the learning performance.

2.3 Offline RL

The offline RL, also known as batch-RL, focuses on the problems where the interaction with the environment is impossible, and the policy can only be optimized based on logged dataset. In those tasks, a fixed buffer $\mathcal{B} = \{s_i, a_i, r_i, s'_i\}_{i=[N]}$, collected from some unknown behavior policy π_β, is provided. In general, such a dataset can either be generated by rolling out an expert that generates high-quality solutions to the task [22, 45, 46] or a non-expert that executes sub-optimal behaviors [47–49, 46, 50] or be a mixture of both [23]. As the agent in the offline RL setting can not correct its potentially biased knowledge through interactions, the most important issue is to address the extrapolation error [22] induced by distributional mismatch [48]. To address such an issue, a series of algorithms optimize the policy learning under the constraint of distributional similarity [48, 47, 51].

[1] Or computed with only states, i.e., $r_{\text{int}}(s) = |\phi_2(s) - \phi_1(s)|$.

Bharadhwaj et al. [23] propose CQL to solve the offline RL tasks with a conservative value estimation. Specifically, CQL learns the Q-value estimation by jointly maximizing the Q-values of actions sampled from the behavior offline dataset and minimizing the Q-values of actions sampled with pre-defined prior distributions (e.g., uniform distribution over the action space). As we will show in this work, an alternative approach to have a lower bound for the optimal Q-value function is to use an appropriately shifted reward function. This idea leads to the direct application of our proposed framework in the offline setting. In general, reward shift can be plugged in many distribution-matching offline-RL algorithms [22, 48, 47, 51] to further improve the performance with conservative Q-value estimation.

Table 1 contextualizes reward shifting with respect to related works we discussed above, more discussions on related works can be found at Appendix E.

3 A Motivating Example

We start with two intuitive remarks and a "counter-intuitive" motivating example in this section before introducing our method.

Remark 1. Given an MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho_0, \gamma\}$, where $|\mathcal{A}| < \infty$, scaling the reward function with linear transformation, i.e., $\mathcal{R}_{k,b} = k \cdot \mathcal{R} + b, \forall k > 0, b \in \mathbb{R}$, such that $r'_t = kr_t + b \in \mathcal{R}_{k,b}$, does not change the optimal policy induced by the corresponding value function $Q^*_{k,b}(s,a) := \sum_t \gamma^t r'_t$:

$$\pi^*(s) = \arg\max_{a\in\mathcal{A}} Q^*_{k,b}(s,a) = \arg\max_{a\in\mathcal{A}} kQ^*(s,a) + \frac{b}{1-\gamma} = \arg\max_{a\in\mathcal{A}} Q^*(s,a), \tag{3}$$

Remark 2. When $|\mathcal{A}| = \infty$, scaling the reward function with linear transformation does not change the optimal policy induced by deterministic policy gradient [21], given proper learning rate $\eta' = \eta/k$:

$$\nabla_\theta J(\mu_\theta) = \mathbb{E}_{s_t}[\nabla_a Q^*(s_t,a_t)|_{a_t=\mu_\theta(s_t)} \nabla_\theta \mu_\theta(s_t)] = \mathbb{E}_{s_t}[\nabla_a Q^*_{k,b}(s_t,a_t)|_{a_t=\mu_\theta(s_t)} \nabla_\theta \mu_\theta(s_t)]/k, \tag{4}$$

Remark 1 and Remark 2 declare the fact that constant reward shifting does not affect the optimal policy induced by the optimal Q-value function. However, Figure 1 presents "counter-intuitive" results that in a demonstrative exploration task of Grid World. In this task, an agent is initialized at the upper left corner of the map need to explore the map without reward unless it reaches the goal point located at the lower right corner, where a positive reward of $+1$ will be assigned. We report learning curves with regard to episodic success rate in reaching the goal point and the learned Q-values. In this toy example, we find a negative reward shifting remarkably boosts the learning efficiency of Q-learning, and surpasses the count-based method that is classic for exploration.

Moreover, Remark 1 is empirically verified with such a toy example, i.e., reward shifting does not change the optimal Q-value as well as its induced policy, but on the contrary, accelerating the learning of (near)-optimal policy.

In the following of this work, we study the effect of varying constant bias b and fix the scaling factor $k = 1$ to avoid trivial discussions on the choices of learning rate — there is no surprise that changing an appropriate learning rate can be empirically important. We focus on revealing the importance of selecting the universal bias term b in the reward function through the lens of initialization priors in function approximation, and show such a bias is generally helpful for both online and offline settings. In the online settings it improves learning efficiency, and in the offline settings it promotes conservative exploitation.

4 Shifted Priors for Q-value Estimation

4.1 Reward Shift Equals to Different Initialization

We start by formally introducing notions and the key idea of this work: reward shifting equals different initialization. We use Figure 2 to illustrate how reward shifting affects function approximation, hence changing the learning dynamics for value-based algorithms. The original optimal Q-value function is denoted as Q^*, and plotted in the figures as **Black** curves. We then denote the shifted optimal Q-value function as $Q^*_{b^+}$, which is the Q-value function with the shifted reward $r' = r + b^+$. We use **Red** curves in figures to denote those shifted Q-value functions. In this section we provide

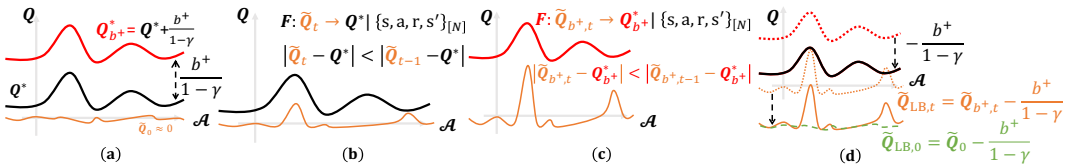

Figure 2: Illustrative figure for conservative exploitation, with a positive constant bias added to the reward function. We use **Black** lines to denote the original value function, and use **Red** lines to denote the shifted value function with the constant reward shift. **Orange** lines are used to denote the function approximation during different learning stages and **Green** line shows the equivalent **pessimistic initialization**. **(a)** shifting the reward function with a positive bias term b^+ leads to an uniformly increased Q-value function, namely $Q^*_{b^+} = Q^* + \frac{b^+}{1-\gamma}$, during learning, a neural network estimator $\tilde{Q}$ initialized with $\tilde{Q}_0 \approx 0$ is optimized to approximate the Q-value functions (e.g., through TD or MC estimation). **(b)** for any value-based RL algorithm, the value optimization step can be regarded as a function F that minimizes the difference between the estimated Q-value function $\tilde{Q}_t$ and the optimal one Q^*, given the interaction experience with the environment (e.g., a replay buffer $\mathcal{B}$ for off-policy methods). Note $\tilde{Q}_t$ approximating Q^* better than $\tilde{Q}_{t-1}$ holds in expectation as long as the RL algorithm is designed to approximate the optimal value function. **(c)** similarly, the optimizer given the same interactive experience (e.g., replay buffer $\mathcal{B}$) will learn to minimize the difference between Q-value function $\tilde{Q}_{b^+,t}$ and the optimal one $Q^*_{b^+}$, after re-labeling the rewards in the buffer by $r' = r + b^+$. **(d)** according to Remark 2, the optimization conducted in (c) is equivalent to (b) with the neural network Q-value estimator initialized as $\tilde{Q}_{\text{LB},0} \approx \tilde{Q}_0 - \frac{b^+}{1-\gamma}$, rather than $\tilde{Q}_0 \approx 0$. i.e., by shifting the reward with proper positive value b^+, we are able to initialize the Q-value network that lower-bounds the optimal Q-value.

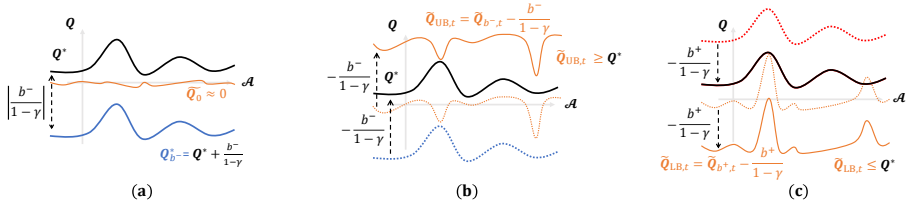

Figure 3: Illustrative figure for curiosity-driven exploration with a negative shifted reward. **(a)** While adding a negative constant value b^- on the reward function leads to negatively shifted optimal Q-value function $Q^*_{b^-}$. **(b)** Minimizing the difference between a Q-value approximator and the optimal Q-value will enable calculating an upper-bound estimation for Q^* which can be used for **optimistic exploration**. **(c)** A positive constant shift added to the reward function can be used for **conservative policy update**.

analysis with a positive bias $b^+ > 0$ to illustrate how positively shifted value function affects function approximation, and coordinates the normally applied near-zero initialized function approximators (e.g., neural networks. **Orange** curves in the figure) to inspire conservative behaviors. The discussion of negative biases is elaborated in the next section.

To summarize, shifting the reward function with a positive constant is equivalent to initializing the value function network with a smaller value — as the Q-value of unseen state-action pairs during training are much lower than their shifted optimal value, those actions will not be selected in argmax-style policy updates, leading to conservative learning behaviors that benefit policy learning in offline settings [23].

4.2 Conservative Exploitation

According to the key insight we presented in Section 4.1, conservative Q-value estimation can emerge with a positively shifted reward. As such a value estimation empirically lower bounds the optimal Q-value function. As shown in Figure 2(a), a positive constant b^+ added to the reward function will lead to a uniformly (positively) shifted optimal Q-value function, and the gap between the primal Q-value function and the new one is $\frac{b^+}{1-\gamma}$. Optimizing the Q-value function with logged data (e.g., a fixed replay buffer) will minimize the difference between the predicted value and the optimal value with observed data. For the unobserved data point (in the state-action space $\mathcal{S} \times \mathcal{A}$), the

near-zero initialization guarantees the prediction is lower than the optimal Q-value, thus conservative exploitation can be conducted with such a value function.

4.3 (Curiosity-Driven) Optimistic Exploration

On the other hand, if we shift the reward function to the negative side, it is equivalent to optimistic initialization. Figure 3 (a-b) illustrate how adding a negative bias leads to curiosity-driven exploration. With sufficiently small b^- (so that $|b^-|$ larger than the maximal value of any (s, a)-pair), such an upper bound of Q^* can be used to conduct curiosity-driven exploration. Intuitively, initializing a value network that predicts value larger or equal than the true value will lead to curiosity-driven exploration, as any visited state will be assigned a relatively smaller value and the argmax-style policy will tend to choose under-explored actions.

Based on the analysis above that (1) a positive constant shift added to the reward function can be used for conservative policy update and (2) a negative constant shift added to the reward function can be used for curiosity-driven exploration, we are ready to access both the upper bound, i.e., the optimistic estimation with b^-, and the lower bound, i.e., the conservative estimation with b^+ of the optimal value function. Formally, we use

$$\tilde{Q}_{\mathrm{LB},t}(s,a) = \tilde{Q}_{b^+,t}(s,a) - \frac{b^+}{1-\gamma} \tag{5}$$

to denote the empirical lower bound of value estimation (cf. Figure 2(d), orange line), and

$$\tilde{Q}_{\mathrm{UB},t}(s,a) = \tilde{Q}_{b^-,t}(s,a) - \frac{b^-}{1-\gamma} \tag{6}$$

to denote the empirical upper bound of value estimation (cf. Figure 3(b)). In both notions, t denotes the optimization step. e.g., when $t = 0$, with a near-zero initialization $\tilde{Q}_0 \approx 0$, $\tilde{Q}_{\mathrm{LB},0}(s,a) = -\frac{b^+}{1-\gamma}$ is able to lower bound the unknown optimal Q-value given sufficiently large b^+. (cf. Figure 2(d), green line). Similarly, Q-value is upper-bounded by $\tilde{Q}_{\mathrm{UB},0}(s,a)$ with a sufficiently small b^-.

Following those notions, we introduce our sample-efficient algorithms for both continuous control and discrete action space respectively. We propose a practical algorithm for sample-efficient continuous control in Section 4.3.1, and focus on a special class of curiosity-driven exploration method, the RND [15], in Section 4.3.2.

4.3.1 Trading-off Exploration and Exploitation with Reward Shift

According to the principle of optimism on the face of uncertainty (OFU), an exploration bonus that manifests the uncertainty of the Q-value function can be introduced into the value estimation: the basic idea starts from integrating optimistic exploration with conservative exploitation, i.e.,

$$\begin{aligned}\hat{Q}(s,a) &= \tilde{Q}_{\mathrm{LB},t}(s,a) + \beta[\tilde{Q}_{\mathrm{UB},t}(s,a) - \tilde{Q}_{\mathrm{LB},t}(s,a)] \\ &= (1-\beta)(\tilde{Q}_{b^+,t}(s,a) - \frac{b^+}{1-\gamma}) + \beta(\tilde{Q}_{b^-,t}(s,a) - \frac{b^-}{1-\gamma}) \\ &= (1-\beta)\tilde{Q}_{b^+,t}(s,a) + \beta\tilde{Q}_{b^-,t}(s,a) - \frac{(1-\beta)b^+ + \beta b^-}{1-\gamma},\end{aligned} \tag{7}$$

where the second term with coefficient β denotes exploration bonus that is composed of uncertainty.

For those under-explored state-action pairs, i.e., extremely out-of-distribution samples for our neural network, both $\tilde{Q}_{b^+,t}(s,a)$ and $\tilde{Q}_{b^-,t}(s,a)$ will give near-zero predictions as a consequence of initialization (detailed implications are provided in Appendix B). Hence, the explorative bonus becomes $-\frac{(1-\beta)b^+ + \beta b^-}{1-\gamma}$. Note this is equivalent to applying another constant reward shift with value of $c_r = (1-\beta)b^+ + \beta b^-$.

Proposition 4.1. *Assuming we have access to an unbiased estimator for the optimal value function Q^*, e.g., with Monte-Carlo estimation $\hat{Q}^* = \mathbb{E}\sum_t \gamma^t r$, and the optimization is based on minimizing the MSE between the unbiased estimator and the function approximator, i.e., $\epsilon_t^2 = (\tilde{Q}_t - \hat{Q}^*)^2$, $\tilde{Q}_t = \tilde{Q}_{t-1} - 2\eta(\tilde{Q}_{t-1} - \hat{Q}^*)$, then the linear combination in Equation (7) is equivalent to a linear combination of the constants with value of $c_r = (1-\beta)b^+ + \beta b^-$.*

The proof can be found in the Appendix A. According to Proposition 4.1, a grid search for trading-off between the three hyper-parameters: the exploration bias b^-, the exploitation bias b^+ and the coefficient β in Equation (7) is trivial as they only lead to a linear combination as $c_r = (1-\beta)b^+ + \beta b^-$, indicating that

Corollary 4.2. *Changing the reward shifting constant b is sufficient to trade off between exploration and exploitation.*

The corollary indicates that, in principle, a meta-learner can be trained to monitor the learning process and select a proper reward shifting constant automatically [52, 53]. For the ease of exposition, in this work we choose to focus on the simplest yet effective uniform sampling strategy from multiple shift constants, which has been shown as a strong baseline of those meta-learner approaches [54, 55], and leave more complicated meta-learner-based approaches for future investigation.

Specifically, we use multiple Q-networks to learn with transition tuples (s, a, r, s') sampled from the identical buffer that collects the policy's historcal interactions with the environment. In propagating Q-values through the temporal difference loss, the primal recorded reward value r is replaced with a new reward with *different* shifted constant bias in those Q-networks. We then uniformly sample one of those learned Q-networks for the optimization of policy network (with e.g., DPG [21]). It is worth noting that our approach only requires post-hoc revision of the primal reward function, rather than interacting with the environment multiple times to collect samples for each value network. We dub the proposed method Random Reward Shift (RRS), and provide the pseudo-code in Algorithm 1 of Appendix C.2.

4.3.2 Tailored Curiosity-Driven Exploration for Value-Based Method

In previous works, the curiosity-driven exploration methods are always work with policy-based methods. In this section, we cast the key insight introduced above to RND [15] — one of the leading algorithms for exploration — to answer why is its vanilla design not suitable for value-based algorithms like Q-learning, and propose a variant of RND that is tailored for DQN.

The vanilla RND use two randomly initialized networks ϕ_1 and ϕ_2 to generate the intrinsic reward $r_{\text{int}} = |\phi_1(s,a) - \phi_2(s,a)| \geq 0$ for exploration. During learning, ϕ_2 is a fixed network and the parameters of ϕ_1 is optimized to approximate the output of ϕ_2 for frequently-visited states. We follow Burda et al. [15] to bound the intrinsic reward below 1 and use $r_{\text{int},t}$ to denote the intrinsic reward after t step of optimization. Specifically, $r_{\text{int},0}(s,a) = 1, \forall(s,a)$ — an universally positive bonus is added to the primal reward function at beginning.

According to our analysis in previous sections, such a positive reward shift is equivalent to pessimistic initialization and will lead to conservative behaviors in Q-value estimation. Therefore, the exploration behaviors at the beginning of learning will be hindered, rather than boosted. To overcome such a conservative behavior induced by the pessimistic initialization, we proposed to use $r^-_{\text{int}}(s,a) = |\phi_1(s,a) - \phi_2(s,a)|^2 - I, \forall s, a$, where I is a positive constant that assures $r^-_{\text{int}} \leq 0$ is negatively initialized for optimistic exploratory behaviors.

The Chicken-and-Egg Problem To further understand the difference between the curiosity-driven method in ways of intrinsic reward and our reward shifting-based optimistic initialization, let us consider when do those curiosities work in each case: for intrinsic reward methods, an agent must **first experience** a new (s, a) pair before receiving a high intrinsic reward — this is extremely hard with an arg-max style policy. On the other hand, with optimistic initialization, the rarely-visited (s, a) pairs will naturally have higher Q-values **before experiencing** it — as the frequently-visited pairs have updated their values with a negatively shifted reward. From such a perspective, reward shifting not only works by itself motivates exploratory behaviors, but can also be seamlessly plugged into intrinsic reward methods to **encourage the first visitation** of new states.

5 Experiments

5.1 Offline RL with Conservative Q-value Estimation

We start our experiments by demonstrating the effectiveness of reward shifting in offline RL benchmarks. As discussed in Section 4.2, a positive reward shift is equivalent to pessimistic initialization that benefits conservative exploitation. In general, our proposed method can be plugged in to any

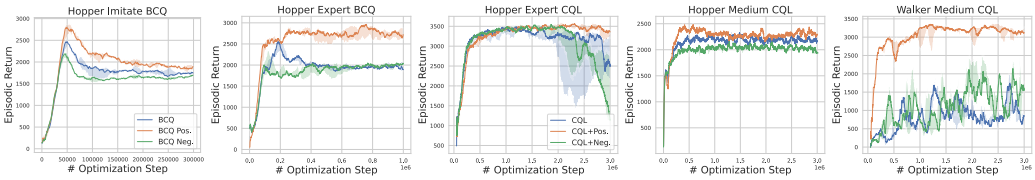

Figure 4: Results on offline RL settings. We verify our key insight that a positive reward shift equals to conservative exploitation thus helps offline value estimation, while a negative reward shift leads to worse performance. Results are from 10 runs with shaded areas indicating the 25%-75% quantiles.

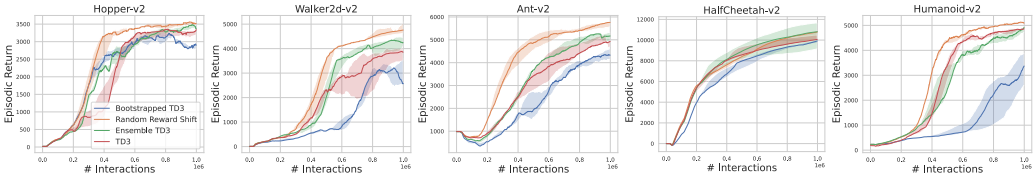

Figure 5: Results on continuous control tasks, Random Reward Shift (RRS) outperforms its value-based baselines in most environments. Results are from 10 runs with shaded areas indicating the 25%-75% quantiles.

off-the-shelf offline RL algorithm, we choose to verify the effectiveness and generality of such a conservative Q-value estimation based on BCQ [22] and CQL [23], i.e., both distribution-matching approach and conservative value estimation approaches in offline RL.

To verify our insight, we experiment with both positive reward shift (**Pos.**) and negative reward shift (**Neg.**), added on either BCQ or CQL. Figure 4 shows our experiment results. We experiment with both the dataset generated in [22] (Hopper Imitate) and the dataset used in [46] (others), and find in our experiments that learning with the CQL dataset is much more stable. The first two panels show results with BCQ as the backbone algorithm. We observe clear performance drop when training with BCQ as the policy overfits the given dataset in these settings, on the other hand, our proposed reward shift can alleviate such a problem and outperform BCQ w.r.t. both best achieved performance and convergence performance. The following three panels in Figure 4 use CQL as the backbone. In all experiments, using a positive reward shift leads to improved learning performance, while a negative reward shift leads to performance decay, as expected. Implementation details and more results can be found in Appendix C.1.

5.2 Online RL with Randomized Priors

We then conduct experiments on the online RL settings. We demonstrate our proposed method in the MuJoCo locomotion benchmarks. As our implementation is based on TD3, we use TD3-based variants as our baselines: The **TD3** is trained with default settings according to Fujimoto et al. [14]. We also include **Ensemble TD3** and **Bootstrapped TD3** as baselines due to they are similar to our work in using multiple Q-networks in value estimation. We follow Osband et al. [44] but extend it to the continuous control settings. In continuous control settings, the argmax operator is approximated by the policy network, multiple policy networks are needed to cooperate with the multiple bootstrapped Q-value networks. Otherwise, multiple Q-value networks are not independent of each other thus breaking the condition of bootstrapped value estimation. The Ensemble TD3 presents the baseline performance when multiple Q-networks are used for value estimation in TD3, which also works as an ablation of our method when all reward shift priors are set to be 0.

As has been illustrated in Sec. 4.3.1, learning with different reward shifting values is equivalent to learning with optimistic or conservative initialization. In our method of RRS we use 3 Q-networks with different priors. In our experiments, we found $\pm 0.5, 0$ work universally as a default setting. Though, future investigation on hyper-parameter may help to further improve the performance.

Results are shown in Figure 5. RRS outperforms the vanilla TD3 in all five environments and outperforms all baseline methods in most tasks. In the experiment of Bootstrapped TD3 and Ensemble TD3, we also use 3 Q-networks for a fair comparison. Note that there is a trade-off between computational complexity and sample efficiency, i.e., using more Q-networks may further improve the

performance at the cost of more computational expensive, as reported in [44]. More implementation details, pseudo code of RRS, and ablation studies can be found in Appendix C.2.

5.3 Optimistic Random Network Distillation

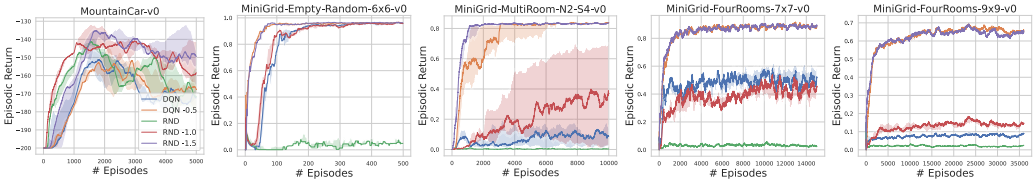

Figure 6: Value-based RND with shifted prior: Plugging the vanilla RND into DQN is not well-motivated according to our analysis in Section 4.3.2. The insight of equivalence between negative reward shifting and curiosity-driven exploration motivates us to negatively shift the intrinsic reward of RND, which drastically improves DQN-based RND. Results are from 10 runs with shaded areas indicating the 25%-75% quantiles.

To demonstrate the key insight and effectiveness of reward shifting in optimistic exploration, we benchmark with five discrete exploration tasks, including the classic MountainCar control and four MiniGrid navigation tasks [56], environment details can be found at Appendix C.3.

Figure 6 presents the results of 5 different methods for comparison: besides the as-is ①**DQN** and ②**RND** baselines, we use ③**DQN -0.5** to indicate DQN with a -0.5 reward shift, and ④**RND -1.0** ⑤ **RND -1.5** to indicate RND with -1.0, -1.5 reward shift, respectively. In the following, we use $\succ$ between methods to indicate the former outperforms the latter. Comparing the results:

1. reward shifting is equivalent to optimistic initialization, thus boosting exploration
③ $\succ$ ①: a negative reward shift is equivalent to optimistic initialization and helps exploration.

2. RND is effective for value-based exploration — as long as a negative intrinsic reward is used
① $\succ$ ②: vanilla RND with positive intrinsic reward is always worse than DQN: adding a positive intrinsic reward like RND is harmful for exploration as it is equivalent to a pessimistic initialization. ④ $\succ$① ; ⑤ $\succ$ ③: RND is effective for exploration (i.e., improve over DQN) as long as the intrinsic reward bonus is always negative.

3. reward shifting for optimistic initialization is orthogonal to other exploration methods
⑤ $\succ$ ④: increasing the magnitude of the negatively shifted reward can further improve the exploration performance — reward shifting can either work in isolation or get combined with other exploration algorithms as they are working orthogonally. [2]

6 Conclusion

In this work, we studied how reward shifting affects policy learning in value-based deep reinforcement learning algorithms. Although constant reward shifting should not change the optimal policy induced by the optimal value function, in practice such a constant shift *does affect the function approximation*, and leads to different learning behaviors. Our detailed analysis manifests the fact that a constant reward shift is equivalent to using different initialization in the value function approximation. The proposed idea is then verified through a variety of application settings. Specifically, we show that a negative reward shift leads to curiosity-driven exploration, while a positive reward shift helps conservative exploitation. Importantly, our analysis manifests that changing reward shifting constant itself is sufficient in trading-off between exploration and exploitation. We empirically verify the effectiveness of the proposed method on a variety of experiments, including better exploitation in offline RL, sample-efficient learning in continuous control benchmarks, and enhanced curiosity-driven exploration in value-based discrete control.

While our experiments demonstrate the performance gain is quite robust to the shifting constant, we would like to point out the theoretical guidance for such a shifting constant is missing. Analysis from the perspective of optimization is an important topic, yet it is out of the scope of the current empirical study and left for future research.

[2] Code is available at https://anonymous.4open.science/r/RewardShifting-BCE7/

References

[1] Jette Randløv and Preben Alstrøm. Learning to drive a bicycle using reinforcement learning and shaping. In *ICML*, volume 98, pages 463–471. Citeseer, 1998.

[2] Adam Daniel Laud. *Theory and application of reward shaping in reinforcement learning*. University of Illinois at Urbana-Champaign, 2004.

[3] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[4] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[5] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[6] Mahmoud Elbarbari, Kyriakos Efthymiadis, Bram Vanderborght, and Ann Nowé. Ltlf-based reward shaping for reinforcement learning. In *Adaptive and Learning Agents Workshop 2021*, 2021.

[7] Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Conference on robot learning*, pages 482–495. PMLR, 2017.

[8] Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

[9] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

[10] Eric Wiewiora, Garrison W Cottrell, and Charles Elkan. Principled methods for advising reinforcement learning agents. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 792–799, 2003.

[11] Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. 1998.

[12] 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.

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[14] Scott Fujimoto, Herke Van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

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

[16] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

[17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[18] Karl W Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *International Conference on Machine Learning*, pages 2020–2027. PMLR, 2021.

[19] Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. *arXiv preprint arXiv:1611.01224*, 2016.

[20] 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.

[21] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

[22] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018.

[23] Homanga Bharadhwaj, Aviral Kumar, Nicholas Rhinehart, Sergey Levine, Florian Shkurti, and Animesh Garg. Conservative safety critics for exploration. *arXiv preprint arXiv:2010.14497*, 2020.

[24] Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 8617–8629, 2018.

[25] Tabish Rashid, Bei Peng, Wendelin Boehmer, and Shimon Whiteson. Optimistic exploration even with a pessimistic initialisation. *arXiv preprint arXiv:2002.12174*, 2020.

[26] Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better exploration with optimistic actor critic. In *Advances in Neural Information Processing Systems*, pages 1785–1796, 2019.

[27] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29:1471–1479, 2016.

[28] Georg Ostrovski, Marc G Bellemare, Aäron Oord, and Rémi Munos. Count-based exploration with neural density models. In *International conference on machine learning*, pages 2721–2730. PMLR, 2017.

[29] 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.

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

[31] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Variational information maximizing exploration. 2016.

[32] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 16–17, 2017.

[33] 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.

[34] Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. *arXiv preprint arXiv:1806.05635*, 2018.

[35] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.

[36] Hao Sun, Zhizhong Li, Xiaotong Liu, Bolei Zhou, and Dahua Lin. Policy continuation with hindsight inverse dynamics. In *Advances in Neural Information Processing Systems*, pages 10265–10275, 2019.

[37] 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.

[38] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

[39] 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.

[40] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

[41] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 263–272. JMLR. org, 2017.

[42] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pages 4863–4873, 2018.

[43] Chen Tessler, Guy Tennenholtz, and Shie Mannor. Distributional policy optimization: An alternative approach for continuous control. *arXiv preprint arXiv:1905.09855*, 2019.

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

[45] Chi Zhang, Sanmukh Rao Kuppannagari, and Viktor Prasanna. Brac+: Going deeper with behavior regularized offline reinforcement learning. 2020.

[46] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[47] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[48] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*, 2019.

[49] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pages 104–114. PMLR, 2020.

[50] Daniel Jarrett, Alihan Hüyük, and Mihaela Van Der Schaar. Inverse decision modeling: Learning interpretable representations of behavior. In *International Conference on Machine Learning*, pages 4755–4771. PMLR, 2021.

[51] Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

[52] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[53] Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *Conference on Robot Learning*, pages 835–853. PMLR, 2020.

[54] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *international conference on machine learning*, pages 1311–1320. PMLR, 2017.

[55] Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher–student curriculum learning. *IEEE transactions on neural networks and learning systems*, 31(9):3732–3740, 2019.

[56] Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid, 2018.

[57] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

[58] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pages 449–458. PMLR, 2017.

[59] Will Dabney, Mark Rowland, Marc Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[60] Clare Lyle, Marc G Bellemare, and Pablo Samuel Castro. A comparative analysis of expected and distributional reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4504–4511, 2019.

[61] Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

[62] Will Dabney, Zeb Kurth-Nelson, Naoshige Uchida, Clara Kwon Starkweather, Demis Hassabis, Rémi Munos, and Matthew Botvinick. A distributional code for value in dopamine-based reinforcement learning. *Nature*, 577(7792):671–675, 2020.

[63] Núria Armengol Urpí, Sebastian Curi, and Andreas Krause. Risk-averse offline reinforcement learning. *arXiv preprint arXiv:2102.05371*, 2021.

[64] Grégoire Delétang, Jordi Grau-Moya, Markus Kunesch, Tim Genewein, Rob Brekelmans, Shane Legg, and Pedro A Ortega. Model-free risk-sensitive reinforcement learning. *arXiv preprint arXiv:2111.02907*, 2021.

[65] Richard Y Chen, Szymon Sidor, Pieter Abbeel, and John Schulman. Ucb exploration via q-ensembles. *arXiv preprint arXiv:1706.01502*, 2017.

[66] 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.

[67] Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

[68] Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double q-learning: Learning fast without a model. *arXiv preprint arXiv:2101.05982*, 2021.

[69] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[70] William R Clements, Bastien Van Delft, Benoît-Marie Robaglia, Reda Bahi Slaoui, and Sébastien Toth. Estimating risk and uncertainty in deep reinforcement learning. *arXiv preprint arXiv:1905.09638*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We discussed in the Conclusion that this work is mainly limited in empirical studies.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] Our research studies the general problem of reward shifting in reinforcement learning, aiming at improve learning efficiency.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] Please refer to Appendix A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Implementation details can be found at Appendix C. Code is open-sourced at https://anonymous.4open.science/r/RewardShifting-BCE7/
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Implementation details can be found at Appendix C.1, C.2, and C.3.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] In our experiments, the results are averaged over 10 runs, and shaded areas indicate the 25% - 75% quantile values.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please refer to Appendix C.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] [22, 23]
   (b) Did you mention the license of the assets? [Yes] MIT license.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We follow BCQ and CQL to use the dataset they used in the papers.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] Not applicable.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] Not applicable.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] Not applicable.