On the Estimation Bias in Double Q-Learning

Anonymous Authors[1]

Abstract

Double Q-learning is a classical method for reducing overestimation bias, which is caused by taking maximum estimated values in the Bellman operation. Its variants in the deep Q-learning paradigm have shown great promise in producing reliable value prediction and improving learning performance. However, as shown by prior work, double Q-learning is not fully unbiased and suffers from underestimation bias. In this paper, we show that such underestimation bias may lead to multiple non-optimal fixed points under an approximated Bellman operator. To address the concerns of converging to non-optimal stationary solutions, we propose a simple but effective approach as a partial fix for the underestimation bias in double Q-learning. This approach leverages an approximate dynamic programming to bound the target value. We extensively evaluate our proposed method in the Atari benchmark tasks and demonstrate its significant improvement over baseline algorithms.

1. Introduction

Value-based reinforcement learning with neural networks as function approximators has become a widely-used paradigm and shown great promise in solving complicated decision-making problems in various real-world applications, including robotics control (Lillicrap et al., 2016), molecular structure design (Zhou et al., 2019), and recommendation systems (Chen et al., 2018). Towards understanding the foundation of these successes, investigating algorithmic properties of deep-learning-based value function approximation has been seen a growth of attention in recent years (Van Hasselt et al., 2018; Fu et al., 2019; Achiam et al., 2019; Dong et al., 2020). One of the phenomena of interest is that Q-learning (Watkins, 1989) is known to suffer from overestimation issues, since it takes a maximum operator over estimated action-values. Comparing with underestimated values, overestimation errors are more likely to be propagated through greedy action selections, which leads to an overestimation bias in value prediction (Thrun & Schwartz, 1993). This overoptimistic behavior of decision making has also been investigated in the literature of management science (Smith & Winkler, 2006) and economics (Thaler, 1988).

From a statistical perspective, the value estimation error may come from many sources, such as the stochasticity of the environment and the imperfection of sample-based estimations. However, for deep Q-learning algorithms, even if most benchmark environments are nearly deterministic (Brockman et al., 2016) and millions of samples are collected, the overestimation phenomenon is still dramatic (Hasselt et al., 2016). One cause of this problematic issue is the difficulty of optimization. Although a deep neural network may have a sufficient expressiveness power to represent an accurate value function, the back-end optimization is hard to solve. As a result of computational considerations, stochastic gradient descent is almost the default choice for deep Q-learning algorithms. The high variance of gradient estimation by such stochastic methods would lead to an unavoidable approximation error in value prediction. This kind of approximation error cannot be eliminated by simply increasing sample size and network capacity, which is a major source of overestimation bias.

Double Q-learning is a classical method to reduce the risk of overestimation, which is a specific variant of the double estimator (Stone, 1974) in the Q-learning paradigm. It uses a second value function to construct an independent action-value evaluation as a cross validation. With proper assumptions, double Q-learning was proved to underestimate rather than overestimate the maximum expected values (Van Hasselt, 2010). In continuous control domains, obtaining two independent value estimators is usually intractable in large-scale tasks, which makes double Q-learning still suffer from a minor overestimation in some situations. To address these concerns, Fujimoto et al. (2018) proposed a variant named clipped double Q-learning, which takes the minimum over two value estimations. This approach implicitly penalizes regions with high uncertainty (Fujimoto et al., 2019) and thus significantly repress the incentive of overestimation. This technique has become the default implementation of most advanced approaches (Haarnoja et al.,

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

2018; Kalashnikov et al., 2018) for continuous control.

In this paper, we first review an analytical model adopted by prior work (Thrun & Schwartz, 1993; Lan et al., 2020) and reveal a fact that, due to the perturbation of approximation error, both double Q-learning and clipped double Q-learning have multiple approximate fixed points in this model. This result raises a concern that double Q-learning may easily get stuck in some local stationary regions and become inefficient in searching for the optimal policy. Motivated by this finding, we propose a novel value estimator, named *doubly bounded estimator*, that utilizes an abstracted dynamic programming as a lower bound estimation to rule out the potential non-optimal fixed points. The proposed method is easy to be combined with other existing techniques such as clipped double Q-learning. We extensively evaluate our approach on a variety of Atari benchmark tasks, and demonstrate significant improvement over baseline algorithms in terms of sample efficiency and convergence performance.

2. Background

Markov Decision Process (MDP; Bellman, 1957) is a classical framework to formalize an agent-environment interaction system which can be defined as a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$. We use $\mathcal{S}$ and $\mathcal{A}$ to denote the state and action space, respectively. $P(s'|s,a)$ and $R(s,a)$ denote the transition and reward functions, which are initially unknown to the agent. γ is the discount factor. The goal of reinforcement learning is to construct a policy $\pi : \mathcal{S} \to \mathcal{A}$ maximizing discounted cumulative rewards $V^\pi(s) =$

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \middle| s_0 = s, s_{t+1} \sim P(\cdot|s_t, \pi(s_t))\right].$$

Another quantity of interest in policy learning can be defined through the Bellman equation $Q^\pi(s,a) = R(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}\left[V^\pi(s')\right]$. The optimal value function Q^* corresponds to the unique solution of the Bellman optimality equation, $\forall (s,a) \in \mathcal{S} \times \mathcal{A}$,

$$Q^*(s,a) = R(s,a) + \gamma \mathop{\mathbb{E}}_{s' \sim P(\cdot|s,a)}\left[\max_{a' \in \mathcal{A}} Q^*(s',a')\right].$$

Q-learning algorithms are based on the Bellman operator $\mathcal{T}$ stated as follows:

$$(\mathcal{T}Q)(s,a) = R(s,a) + \gamma \mathop{\mathbb{E}}_{s' \sim P(\cdot|s,a)}\left[\max_{a' \in \mathcal{A}} Q(s',a')\right]. \tag{1}$$

By iterating this operator, value iteration is proved to converge to the optimal value function Q^*. To extend Q-learning methods to real-world applications, function approximation is indispensable to deal with a high-dimensional state space. Deep Q-learning (Mnih et al., 2015) considers a sample-based objective function $L(\theta; \theta_t)$ and deploys an iterative optimization framework $\theta_{t+1} \leftarrow \arg\min_{\theta \in \Theta} L(\theta; \theta_t)$ where $L(\theta; \theta_t) =$

$$\mathop{\mathbb{E}}_{(s,a,r,s')}\left[\left(r + \gamma \max_{a' \in \mathcal{A}} Q_{\theta_t}(s',a') - Q_\theta(s,a)\right)^2\right], \tag{2}$$

in which Θ denotes the parameter space of the value network, and $\theta_0 \in \Theta$ is initialized by some predetermined method. (s,a,r,s') is sampled from a data distribution $\mathcal{D}$ which is changing during exploration. With infinite samples and a sufficiently rich function class, the update rule stated in Eq. (2) is asymptotically equivalent to applying the Bellman operator $\mathcal{T}$, but the underlying optimization is usually inefficient in practice. In deep Q-learning, Eq. (2) is optimized by mini-batch gradient descent and thus its value estimation suffers from unavoidable approximation errors.

3. On the Effects of Underestimation Bias in Double Q-Learning

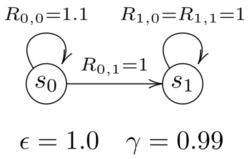
In this section, we will first revisit a common analytical model used by previous work for studying estimation bias (Thrun & Schwartz, 1993; Lan et al., 2020), in which double Q-learning is known to have underestimation bias. Based on this analytical model, we show that its underestimation bias could make double Q-learning have multiple fixed-point solutions under an approximate Bellman operator. This result suggests that double Q-learning may have extra non-optimal stationary solutions under the effects of the approximation error.

3.1. Modeling Approximation Error in Q-Learning

Following the model adopted by Thrun & Schwartz (1993) and Lan et al. (2020), we formalize the underlying approximation error of target value regression as a set of random noises $e^{(t)}(s,a)$ impacting on the Bellman operation $Q^{(t+1)}(s,a) = (\widetilde{\mathcal{T}}Q^{(t)})(s,a)$, in which $\widetilde{\mathcal{T}}$ denotes a stochastic operator with noisy outputs:

$$(\widetilde{\mathcal{T}}Q^{(t)})(s,a) = (\mathcal{T}Q^{(t)})(s,a) + e^{(t)}(s,a), \tag{3}$$

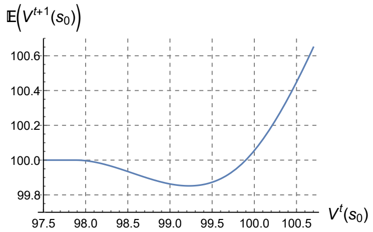
where t denotes the iteration number, and $\mathcal{T}$ denotes the ground truth Bellman operator using full information of the MDP (see Eq. (1)). The main purpose of introducing the explicit noise term $e^{(t)}(s,a)$ is to emphasize that the approximation error discussed here is different from the term "*sampling error*". In an information-theoretic perspective, the sampling error can be reduced asymptotically as the sample size increases. However, there is a barrier of optimization difficulty to establish a precise estimation in practice, which leads to an unavoidable approximation error in optimization. By integrating the noise term $e^{(t)}(s,a)$ into the Bellman operation, this analytical model enables us



(a) A simple construction

$V(s_0)$	$V(s_1)$	$\tilde{\pi}(a_0 \mid s_0)$
100.162	100.0	62.2%
101.159	100.0	92.9%
110.0	100.0	100.0%

(b) Numerical solutions of fixed points



(c) Visualizing non-monotonicity

Figure 1. (a) A simple infinite-horizon MDP where double Q-learning stated as Eq. (4) has multiple approximate fixed points. $R_{i,j}$ is a shorthand of $R(s_i, a_j)$. (b) The numerical solutions of the fixed points produced by double Q-learning in the MDP presented above. $\tilde{\pi}$ denotes the expected policy generated by the corresponding fixed point under the perturbation of noise $e(s,a)$. A formal description of $\tilde{\pi}$ refers to Definition 2 in Appendix A.3. (c) The relation between the input state-value $V^{(t)}(s_0)$ and the expected output state-value $\mathbb{E}[V^{(t+1)}(s_0)]$ generated by double Q-learning in the constructed MDP, in which we assume $V^{(t)}(s_1) = 100$.

to investigate how Q-learning algorithms interact with the inherent approximation error during optimization.

In this model, double Q-learning (Van Hasselt, 2010) can be modeled by two estimator instances $\{Q_i^{(t)}\}_{i\in\{1,2\}}$ with separated noise variables $\{e_i^{(t)}\}_{i\in\{1,2\}}$. For simplification, we introduce a policy function $\pi^{(t)}(s) = \arg\max_a Q_1^{(t)}(s,a)$ to override the state value function as follows: $\forall i \in \{1,2\}$,

$$Q_i^{(t+1)}(s,a) = R(s,a) + \gamma \mathbb{E}_{s'}\left[V^{(t)}(s')\right] + e_i^{(t)}(s,a),$$
$$V^{(t)}(s) = Q_2^{(t)}\left(s,\ \arg\max\nolimits_{a\in\mathcal{A}} Q_1^{(t)}(s,a)\right). \qquad (4)$$

A minor difference of Eq. (4) from the definition of double Q-learning given by Van Hasselt (2010) is using a unified target value $V^{(t)}(s')$ for both two estimators. This simplification does not affect the derived implications, and is also implemented by advanced variants of double Q-learning (Fujimoto et al., 2018; Lan et al., 2020).

Note that, as shown in Eq. (4), the target value can be constructed only using the state-value function. Based on this observation, we can define the stationary point of target values as the fixed point of a stochastic Bellman operator.

Definition 1 (Approximate Fixed Points)**.** *Let $\widetilde{\mathcal{T}}$ denote a stochastic Bellman operator, such as what are stated in Eq. (3) and Eq. (4). A state-value function V is regarded as an approximate fixed point under a stochastic Bellman operator $\widetilde{\mathcal{T}}$ if it satisfies $\mathbb{E}[\widetilde{\mathcal{T}}V] = V$, where $\widetilde{\mathcal{T}}V$ denotes the output state-value function while applying the Bellman operator $\widetilde{\mathcal{T}}$ on V.*

The fixed point defined above is named "*approximate*" since they are not truly static, but is invariant under the stochastic Bellman operator in expectation. In Appendix A.2, we will prove the existence of such fixed points as the following statement.

Proposition 1. *Assume the probability density functions of the noise terms $\{e(s,a)\}$ are continuous. The stochastic Bellman operators defined by Eq. (3) and Eq. (4) must have approximate fixed points in arbitrary MDPs.*

3.2. Existence of Multiple Approximate Fixed Points in Double Q-Learning Algorithms

Given the definition of the approximate fixed point, a natural question is whether such kind of fixed points are unique or not. Recall that the optimal value function Q^* is the unique solution of the Bellman optimality equation, which is the foundation of Q-learning algorithms. However, in this section, we will show that, under the effects of the approximation error, the approximate fixed points of double Q-learning may not be unique.

An Illustrative Example. Figure 1a presents a simple MDP in which double Q-learning stated as Eq. (4) has multiple approximate fixed points. For simplicity, this MDP is set to be fully deterministic and contains only two states s_0 and s_1. All actions in state s_1 lead to a self-loop and produce a unit reward signal. On state s_0, the result of executing action a_0 is a self-loop with a slightly larger reward signal than choosing action a_1 which leads to state s_1. The only challenge for decision making in this MDP is to distinguish the outcomes of executing action a_0 and a_1 on state s_0. To make the example more accessible, we assume the approximation errors $\{e^{(t)}(s,a)\}_{t,s,a}$ are a set of independent random variables following a uniform distribution *Uniform*$(-\epsilon, \epsilon)$. This simplification is also adopted by Thrun & Schwartz (1993) and Lan et al. (2020) in case studies. Here, we select the magnitude of noise as $\epsilon = 1.0$ and the discount factor as $\gamma = 0.99$ to balance the scale of involved amounts.

Considering to solve the equation $\mathbb{E}[\widetilde{\mathcal{T}}V] = V$ according to the definition of the approximate fixed point (see Definition 1), the numerical solutions of such fixed points are presented in Table 1b. There are three different fixed point solutions. The first thing to notice is that the optimal fixed point V^* is retained in this MDP (see the last row of Table 1b), since the noise magnitude $\epsilon = 1.0$ is much smaller than the optimality gap $Q^*(s_0, a_0) - Q^*(s_0, a_1) = 10$. The other two fixed points are non-optimal and very close to $Q(s_0, a_0) \approx Q(s_0, a_1) = 100$. Intuitively, under the perturbation of approximation error, the agent cannot identify the correct maximum-value action for policy improvement in these situations, which is the cause of such non-optimal fixed points. To formalize the implications, we would present a sufficient condition for the existence of multiple extra fixed points.

Mathematical Condition. Note that the definition of the stochastic Bellman operator is a model of an imprecise target value regression. From this perspective, the input of a stochastic Bellman operator can be defined as a set of ground truth target values $\{(\mathcal{T}Q^{(t)})(s,a)\}_{s,a}$. Based on this notation, a sufficient condition for the existence of multiple fixed points is stated as follows.

Proposition 2. *Let $f_s\left(\{(\mathcal{T}Q)(s,a)\}_{a\in\mathcal{A}}\right) = \mathbb{E}[(\widetilde{\mathcal{T}}V)(s)]$ denote the expected output value of the stochastic Bellman operator $\widetilde{\mathcal{T}}$ on state s, and assume $f_s(\cdot)$ is differentiable. If a stochastic Bellman operator $\widetilde{\mathcal{T}}$ satisfies Eq. (5), there exists an MDP such that $\widetilde{\mathcal{T}}$ has multiple fixed points.*

$$\exists i,\ \exists X \in \mathbb{R}^{|\mathcal{A}|}, \quad \frac{\partial}{\partial x_i} f_s(X) > 1, \tag{5}$$

where $X = \{x_i\}_{i=1}^{|\mathcal{A}|}$ denotes the input of the function f_s.

The proof of Proposition 2 is deferred to Appendix A.4. This proposition suggests that, in order to determine whether a given stochastic Bellman operator $\widetilde{\mathcal{T}}$ may have multiple fixed points, we need to check whether its expected output values could change dramatically with a slight alter of the input values.

Considering the constructed MDP as an example, Figure 1c visualizes the relation between the input state-value $V^{(t)}(s_0)$ and the expected output state-value $\mathbb{E}[V^{(t+1)}(s_0)]$ while assuming $V^{(t)}(s_1) = 100$ has converged to its stationary point. The minima point of the output value is located at the situation where $V^{(t)}(s_0)$ is slightly smaller than $V^{(t)}(s_1)$, since the expected policy derived by $\widetilde{\mathcal{T}}V^{(t)}$ will have a remarkable probability to choose sub-optimal actions. This local minima suffers from the most dramatic underestimation among the whole curve, and the underestimation will eventually vanish as the value of $V^{(t)}(s_0)$ increases. During this process, a large magnitude of the first-order derivative could be found to meet the condition stated in Eq. (5).

In Appendix A.5, we show that clipped double Q-learning, a popular variant of double Q-learning, has multiple fixed points in an MDP slightly modified from Figure 1a.

3.3. Diagnosing Non-Optimal Fixed Points

In this section, we first characterize the properties of the extra non-optimal fixed points of double Q-learning in the analytical model. And then, we discuss its connections to the literature of stochastic optimization, which motivates our proposed algorithm in section 4.

Underestimated Solutions. The first notable thing is that, the non-optimal fixed points of double Q-learning would not overestimate the true maximum values. Formally, every fixed-point solution could be characterized as the ground truth value function of some stochastic policy as the following proposition.

Proposition 3 (Fixed-Point Characterization)**.** *Assume the noise terms e_1 and e_2 are independently generated in the double estimator stated in Eq. (4). Every approximate fixed point V is equal to the ground truth value function $V^{\tilde{\pi}}$ with respect to a stochastic policy $\tilde{\pi}$.*

The proof of Proposition 3 is deferred to Appendix A.3. In addition, the corresponding stochastic policy $\tilde{\pi}$ can be interpreted as $\tilde{\pi}(a|s) =$

$$\mathbb{P}\left[a = \arg\max_{a'\in\mathcal{A}}\left(\underbrace{R(s,a') + \gamma\mathbb{E}_{s'}\left[V(s')\right]}_{(\mathcal{T}Q)(s,a')} + e(s,a')\right)\right],$$

which is the expected policy generated by the corresponding fixed point along with the random noise $e(s,a')$. This stochastic policy, named as *induced policy*, can provide a snapshot to infer how the agent behaves and evolves around these approximate fixed points. To deliver intuitions, we provide an analogical explanation in the context of optimization as the following arguments.

Analogy with Saddle Points. Taking the third column of Table 1b as an example, due to the existence of the approximation error, the induced policy $\tilde{\pi}$ suffers from a remarkable uncertainty in determining the best action on state s_0. Around such non-optimal fixed points, the greedy action selection may be disrupted by approximation error and deviate from the correct direction for policy improvement. These approximate fixed points are not necessary to be strongly stationary solutions but may seriously hurt the learning efficiency. If we imagine each iteration of target updating as a step of "*gradient update*" for Bellman error minimization, the non-optimal fixed points would refer to the concept of *saddle points* in the context of optimization. As stochastic gradient may be trapped in saddle points, Bellman operation with approximation error may get stuck around non-optimal approximate fixed points.

Escaping from Saddle Points. In the literature of non-convex optimization, the most famous approach to escaping saddle points is *perturbed gradient descent* (Ge et al., 2015; Jin et al., 2017). Recall that, although gradient directions are ambiguous around saddle points, they are not strongly convergent solutions. Some specific perturbation mechanisms with certain properties could help to make the optimizer to escape non-optimal saddle points. Although these methods cannot be directly applied to double Q-learning since the Bellman operation is not an exact gradient descent, it motivates us to construct a specific perturbation for Bellman operations. In section 4, we would introduce a perturbed target updating mechanism that uses an external value estimation to rule out non-optimal fixed points of double Q-learning.

4. Doubly Bounded Q-Learning through Abstracted Dynamic Programming

As discussed in the last section, the underestimation bias of double Q-learning may lead to multiple non-optimal fixed points in the analytical model. A major source of such underestimation is the inherent approximation error caused by the difficulty of optimization. Motivated by the literature of escaping saddle points, we introduce a novel method, named *Doubly Bounded Q-learning*, which integrates two different value estimators to reduce the negative effects of underestimation.

4.1. Algorithmic Framework

As discussed in section 3.3, the geometry property of non-optimal approximate fixed points of double Q-learning is similar to that of saddle points in the context of non-convex optimization. The theory of escaping saddle points suggests that, a well-shaped perturbation mechanism could help to remove non-optimal saddle points from the landscape of optimization (Ge et al., 2015; Jin et al., 2017). To realize this brief idea in the specific context of iterative Bellman error minimization, we propose to integrate a second value estimator using different learning paradigm as an external auxiliary signal to rule out non-optimal approximate fixed points of double Q-learning. To give an overview, we first revisit two value estimation paradigms as follows:

1. **Bootstrapping Estimator:** As the default implementation of most temporal-difference learning algorithms, the target value y^{Boots} of a transition sample (s_t, a_t, r_t, s_{t+1}) is computed through bootstrapping the latest value function back-up $V_{\theta_{\text{target}}}$ parameterized by θ_{target} on the successor state s_{t+1} as follows:

   $$y^{\text{Boots}}_{\theta_{\text{target}}}(s_t, a_t) = r_t + \gamma V_{\theta_{\text{target}}}(s_{t+1}),$$

   where the computations of $V_{\theta_{\text{target}}}$ differ in different algorithms (e.g., different variants of double Q-learning).

2. **Dynamic Programming Estimator:** Another approach to estimating state-action values is applying dynamic programming in an abstracted MDP (Li et al., 2006) constructed from the collected dataset. By utilizing a state aggregation function $\phi(s)$, we could discretize a complex environment to a manageable tabular MDP. The reward and transition functions of the abstracted MDP are estimated through the collected samples in the dataset. An alternative target value y^{DP} is computed as:

   $$y^{\text{DP}}(s_t, a_t) = r_t + \gamma V^*_{\text{DP}}(\phi(s_{t+1})), \tag{6}$$

   where V^*_{DP} corresponds to the optimal value function of the abstracted MDP. A practical implementation refers to section 4.3.

The advantages and bottlenecks of these two types of value estimators lie in different aspects of error controlling. The generalizability of function approximators is the major strength of the *bootstrapping estimator*, but on the other hand, the hardness of the back-end optimization would cause considerable approximation error and lead to the issues discussed in section 3. The tabular representation of the *dynamic programming estimator* would not suffer from systematic approximation error during optimization, but its performance relies on the accuracy of state aggregation and the sampling error in transition estimation.

Doubly Bounded Estimator. To establish a trade-off between the considerations in the above two value estimators, we propose to construct an integrated estimator, named *doubly bounded estimator*, which takes the maximum values over two different basis estimation methods:

$$y^{\text{DB}}_{\theta_{\text{target}}}(s_t, a_t) = \max\left\{ y^{\text{Boots}}_{\theta_{\text{target}}}(s_t, a_t),\ y^{\text{DP}}(s_t, a_t) \right\}. \tag{7}$$

The derived targets values would be used in training the parameterized value function Q_θ by minimizing

$$L(\theta; \theta_{\text{target}}) = \mathop{\mathbb{E}}_{(s_t, a_t)\sim D} \left(Q_\theta(s_t, a_t) - y^{\text{DB}}_{\theta_{\text{target}}}(s_t, a_t) \right)^2,$$

where D denotes the experience buffer. Note that, this estimator maintains two value functions using different data structures. Q_θ is the major value function which is used to generate the behavior policy for both exploration and evaluation. V_{DP} is an auxiliary value function computed by the abstracted dynamic programming, which is stored in a discrete table. The only functionality of V_{DP} is computing the auxiliary target value y^{DP} used in Eq. (7) during training.

Remark. The name "*doubly bounded*" refers to the following intuitive motivation: Assume both basis estimators, y^{Boots} and y^{DP}, are implemented by their conservative variants and do not tend to overestimate values. The doubly bounded target value $y^{\text{DB}}(s_t, a_t)$ would become a good estimation if either of basis estimator provides an accurate value prediction on the given state-action pair (s_t, a_t). The outcomes of abstracted dynamic programming could help the bootstrapping estimator to escape the non-optimal fixed points of double Q-learning. The function approximator used by the bootstrapping estimator could extend the generalizability of discretization-based state aggregation. The learning procedure could make progress if either of estimators can identify the correct direction for policy improvement.

4.2. Underlying Bias-Variance Trade-Off

Formally, we analyze the effects of our proposed method on the underlying bias-variance trade-off as follows.

Provable Benefits on Variance Reduction. The algorithmic structure of the proposed *doubly bounded estimator* could be formalized as a stochastic Bellman operator $\widetilde{\mathcal{T}}^{\text{DB}}$:

$$(\widetilde{\mathcal{T}}^{\text{DB}} V)(s) = \max\left\{ (\widetilde{\mathcal{T}}^{\text{Boots}} V)(s),\ V^{\text{DP}}(s) \right\}, \quad (8)$$

where $\widetilde{\mathcal{T}}^{\text{Boots}}$ is the stochastic Bellman operator corresponding to the back-end bootstrapping estimator (e.g., Eq. (4)). V^{DP} is an arbitrary deterministic value estimator such as using abstracted dynamic programming. The strength of the *doubly bounded estimator* can be characterized as the following proposition.

Proposition 4. *Given an arbitrary stochastic operator $\widetilde{\mathcal{T}}^{Boots}$ and a deterministic estimator V^{DP}, we have*

$$\forall V,\ \forall \in \mathcal{S}, \quad \mathit{Var}[(\widetilde{\mathcal{T}}^{DB} V)(s)] \le \mathit{Var}[(\widetilde{\mathcal{T}}^{Boots} V)(s)],$$

where $(\widetilde{\mathcal{T}}^{DB} V)(s)$ is defined as Eq. (8).

The proof of Proposition 4 is deferred to Appendix A.6. The intuition behind this statement is that, with a deterministic lower bound cut-off, the variance of the outcome target values would be reduced, which may contribute to improve the stability of training.

Trade-Off between Different Biases. In general, the proposed *doubly bounded estimator* does not have a rigorous guarantee for bias reduction, since the behavior of abstracted dynamic programming depends on the properties of the tested environments. In the most unfavorable case, if the dynamic programming component carries a large magnitude of error, the lower bounded objective would propagate high-value errors to increase the risk of overestimation. To address these concerns, we propose to implement a conservative approximate dynamic programming as discussed in section 4.3. The experimental analysis in section 5.1 demonstrate that, the error carried by abstracted dynamic programming is acceptable, and our proposed method definitely works well in most benchmark tasks.

4.3. Practical Implementation

The abstracted dynamic programming module is implemented through the following components. A detailed description is deferred to Appendix B.

State Aggregation. We consider a simple discretization to construct the state abstraction function $\phi(\cdot)$ used in Eq. (6). We first follow the standard Atari pre-processing proposed by Mnih et al. (2015) to rescale each RGB frame to an 84×84 luminance map, and the observation is constructed as a stack of 4 recent luminance maps. We round the each pixel to 256 possible integer intensities and use a standard static hashing (Karp & Rabin, 1987) to set up the table for storing V_{DP}.

Conservative Action Pruning. To obtain a conservative value estimation, we follow the suggestions given by Fujimoto et al. (2019) and Liu et al. (2020) to prune the unseen state-action pairs in the abstracted MDP. The reward and transition functions of remaining state-action pairs are estimated through the average of collected samples.

Computation Acceleration. Note that the size of the abstracted MDP is growing as the exploration. Regarding computational considerations, we adopt the idea of *prioritized sweeping* (Moore & Atkeson, 1993) to accelerate the computation of tabular dynamic programming. In addition to periodically applying the complete Bellman operator, we perform extra updates on the most recent visited states, which would reduce the total number of operations to obtain an acceptable estimation.

Connecting to Parameterized Estimator. Finally, the results of abstracted dynamic programming would be delivered to the deep Q-learning as Eq. (7). Note that the constructed doubly bounded target value $y^{\text{DB}}_{\theta_{\text{target}}}$ is only used to update the parameterized value function Q_θ and would not affect the computation in the abstracted MDP.

5. Experiments

Our experiment environments are based on the standard Atari benchmark tasks supported by OpenAI Gym (Brockman et al., 2016). All baselines and our approaches are implemented using the same set of hyper-parameters suggested by Castro et al. (2018). A detailed description of experiment settings is deferred to Appendix B.

5.1. Performance Comparison on Atari Benchmark

To demonstrate the superiority of our proposed method, *Doubly Bounded Q-Learning through Abstracted Dynamic Programming* (DB-ADP), we investigated six variants of deep Q-networks as baseline algorithms, including DQN (Mnih et al., 2015), double DQN (DDQN; Hasselt et al., 2016), dueling DDQN (Wang et al., 2016), averaged DQN

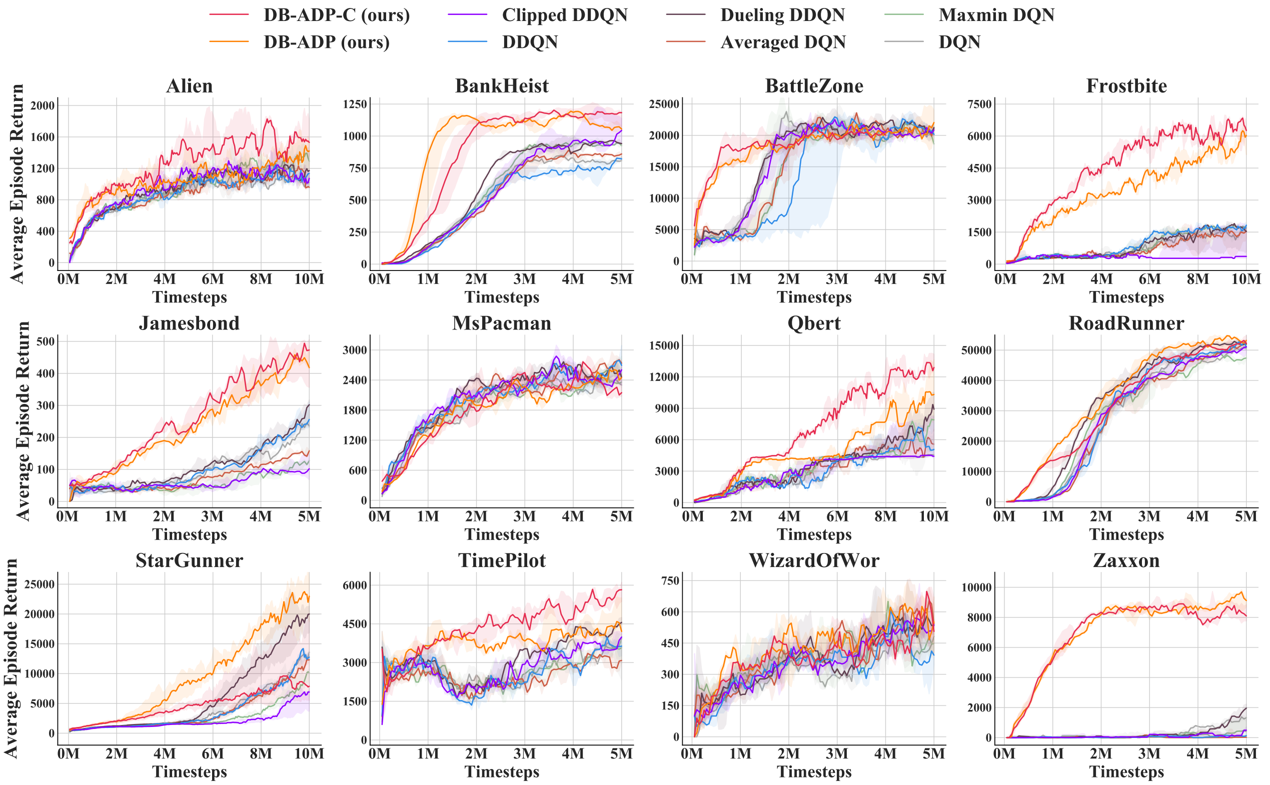


Figure 2. Learning curves on a suite of Atari benchmark tasks. DB-ADP and DB-ADP-C refer to our proposed approach built upon double Q-learning and clipped double Q-learning, respectively. All curves presented in this paper are plotted according to the median performance over 5 runs with random network initialization. To make the comparison more clear, the curves are smoothed by averaging 10 most recent evaluation points.

(Anschel et al., 2017), maxmin DQN (Lan et al., 2020), and clipped double DQN adapted from Fujimoto et al. (2018). Our proposed doubly bounded target estimation y^{DB} is built upon two types of bootstrapping estimators that have clear incentive of underestimation, i.e., double Q-learning and clipped double Q-learning. We denote these two variants as DB-ADP-C and DB-ADP according to our proposed method with or without using clipped double Q-learning.

Overall Comparison. As shown in Figure 2, the proposed doubly bounded estimator has great promise in bootstrapping the performance of double Q-learning algorithms. The improvement can be observed both in terms of sample efficiency and final performance.

The Role of Clipped Double Q-Learning. Another notable observation is that, although clipped double Q-learning can hardly improve the performance upon Double DQN, it can significantly improve the performance through our proposed approach in most environments (i.e., DB-ADP-C vs. DB-ADP in Figure 2). This improvement should be credit to the conservative property of clipped double Q-learning (Fujimoto et al., 2019) that may reduce the propagation of the errors carried by abstracted dynamic programming.

Table 1. Evaluating the standard deviation of target values produced by doubly bounded estimators and bootstrapping estimators. The presented amounts are normalized according to the value scale of corresponding runs. "†" refers to the ablation studies.

TASK NAME	DB-ADP-C	DB-ADP-C†	CDDQN
ALIEN	**0.006**	0.008	0.011
BANKHEIST	**0.008**	0.010	**0.008**
QBERT	**0.007**	0.010	0.011
TASK NAME	DB-ADP	DB-ADP†	DDQN
ALIEN	**0.006**	0.007	0.013
BANKHEIST	**0.008**	0.010	0.015
QBERT	0.008	0.010	0.014

5.2. Variance Reduction on Target Values

To support the theoretical claims in Proposition 4, we conduct an experiment to demonstrate the ability of doubly bounded estimator on variance reduction. Note that, the derived target values after one epoch of training depend on the selection of training batches. From this perspective, we evaluate the standard deviation of the target values with respect to using different training batches. We first collect an experience buffer using the corresponding algorithm, and then we sample a batch of transitions from the buffer as the testing set. For each transition sample in the testing set, we compute the standard deviation of its corresponding target value with respect to network updating using different training batches. Finally, we average the standard deviation evaluation of each single transition as the evaluation of the given algorithm. A detailed description for the evaluation process is included in Appendix B.

Table 1 presents the evaluation results of standard deviations. Three columns of this table correspond to three branches of approaches:

- DB-ADP(-C): We use doubly bounded Q-learning to update the network using different training batches, and evaluate the standard deviation of the doubly bounded target y^{DB} as defined in Eq. (7).
- DB-ADP(-C)†: The $\dagger$-version corresponds to an ablation study, where we train the network using our proposed approach but evaluate the target values computed by bootstrapping estimators, i.e., using the target value formula of double DQN or clipped double DQN.
- (Clipped) DDQN: We also evaluate the corresponding metrics in baseline algorithms.

As shown in Table 1, the standard deviation of target values is significantly reduced by our approaches, which matches our theoretical analysis in Proposition 4. It demonstrates a strength of our approach in improving training stability.

5.3. Ablation Studies on Dynamic Programming

As an ablation study, we investigate an alternative non-parametric value estimator in the construction of doubly bounded target values. We consider the real trajectory return as a lower-bound estimation of state values. Similar ideas are implemented by Oh et al. (2018) and Fujita et al. (2020) in different contexts. As shown in Figure 3, this simplified lower bound can also help to improve the performance upon basis algorithms in some extent. This observation matches our discussion in section 3.3. The geometry properties of non-optimal fixed points are similar to that of saddle points, so that a weak perturbation such as trajectory return can also help to escape from these non-optimal stationary regions.

6. Related Work

Correcting the estimation bias in double Q-learning is an active topic which induces a series of approaches. Weighted double Q-learning (Zhang et al., 2017) considers an importance weight parameter to integrate the overestimated and underestimated estimators. Clipped double Q-learning (Fujimoto et al., 2018), which uses a minimum operator in target values, has become the default implementation of most advanced actor-critic algorithms (Haarnoja et al., 2018). Based on clipped double Q-learning, several methods have been investigated to reduce the its underestimation and achieve promising performance (Ciosek et al., 2019; Li & Hou, 2019). Most recent advances usually focus on using ensemble methods to further reduce the error magnitude (Lan et al., 2020; Kuznetsov et al., 2020; Chen et al., 2021). Besides the variants of double Q-learning, using the softmax operator in Bellman operations is also considered as an effective approach to reduce the effects of approximation error (Fox et al., 2016; Asadi & Littman, 2017; Song et al., 2019; Kim et al., 2019). The characteristic of our approach is the usage of an approximate dynamic programming, which uses the environment prior to break statistical barriers of non-optimal fixed points. Our analysis would provide a theoretical support for memory-based approaches, such as episodic control (Blundell et al., 2016; Pritzel et al., 2017; Lin et al., 2018; Zhu et al., 2020), which are usually specific to near-deterministic environments.

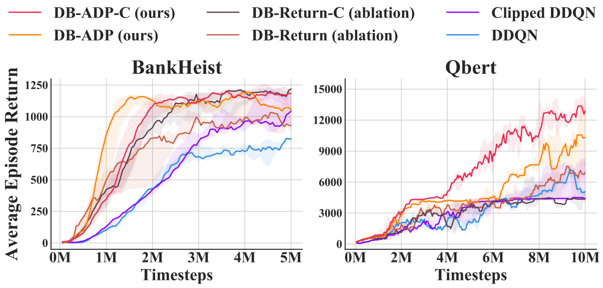


Figure 3. Learning curves of the ablation studies using trajectory return values to construct doubly bounded target values.

7. Conclusion

In this paper, we reveal an interesting fact that, under the effects of approximation error, double Q-learning may have multiple non-optimal fixed points. The main cause of such non-optimal fixed points is the underestimation bias of double Q-learning. Regarding this issue, we provide some analysis to characterize what kind of Bellman operators may suffer from the same problem, and how the agent may behave around these fixed points. To address the potential risk of converging to non-optimal solutions, we propose doubly bounded Q-learning to reduce the underestimation in double Q-learning. The main idea of this approach is to leverage an abstracted dynamic programming as a second value estimator to rule out non-optimal fixed points. The experiments show that the proposed method has shown great promise in improving both sample efficiency and convergence performance. It achieves a significant improvement over baselines algorithms on Atari benchmark environments.

References

Achiam, J., Knight, E., and Abbeel, P. Towards characterizing divergence in deep q-learning. *arXiv preprint arXiv:1903.08894*, 2019.

Anschel, O., Baram, N., and Shimkin, N. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 176–185. PMLR, 2017.

Asadi, K. and Littman, M. L. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, pp. 243–252, 2017.

Bellman, R. Dynamic programming. *Princeton University Press*, 89:92, 1957.

Blundell, C., Uria, B., Pritzel, A., Li, Y., Ruderman, A., Leibo, J. Z., Rae, J., Wierstra, D., and Hassabis, D. Model-free episodic control. *arXiv preprint arXiv:1606.04460*, 2016.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym, 2016.

Brouwer, L. E. J. Über abbildung von mannigfaltigkeiten. *Mathematische annalen*, 71(1):97–115, 1911.

Castro, P. S., Moitra, S., Gelada, C., Kumar, S., and Bellemare, M. G. Dopamine: A research framework for deep reinforcement learning. *arXiv preprint arXiv:1812.06110*, 2018.

Chen, S.-Y., Yu, Y., Da, Q., Tan, J., Huang, H.-K., and Tang, H.-H. Stabilizing reinforcement learning in dynamic environment with application to online recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1187–1196, 2018.

Chen, X., Wang, C., Zhou, Z., and Ross, K. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021.

Ciosek, K., Vuong, Q., Loftin, R., and Hofmann, K. Better exploration with optimistic actor critic. In *Advances in Neural Information Processing Systems*, pp. 1785–1796, 2019.

Dong, K., Luo, Y., Yu, T., Finn, C., and Ma, T. On the expressivity of neural networks for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 2627–2637. PMLR, 2020.

Fox, R., Pakman, A., and Tishby, N. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pp. 202–211, 2016.

Fu, J., Kumar, A., Soh, M., and Levine, S. Diagnosing bottlenecks in deep q-learning algorithms. In *International Conference on Machine Learning*, pp. 2021–2030, 2019.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596, 2018.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062, 2019.

Fujita, Y., Uenishi, K., Ummadisingu, A., Nagarajan, P., Masuda, S., and Castro, M. Y. Distributed reinforcement learning of targeted grasping with active vision for mobile manipulators. *arXiv preprint arXiv:2007.08082*, 2020.

Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pp. 797–842. PMLR, 2015.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1861–1870, 2018.

Hasselt, H. v., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 2094–2100, 2016.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. In *International Conference on Machine Learning*, pp. 1724–1732. PMLR, 2017.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*. PMLR, 2018.

Karp, R. M. and Rabin, M. O. Efficient randomized pattern-matching algorithms. *IBM journal of research and development*, 31(2):249–260, 1987.

Kim, S., Asadi, K., Littman, M., and Konidaris, G. Deepmellow: removing the need for a target network in deep q-learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 2733–2739, 2019.

Kuznetsov, A., Shvechikov, P., Grishin, A., and Vetrov, D. Controlling overestimation bias with truncated mixture of continuous distributional quantile critics. In *International Conference on Machine Learning*, 2020.

Lan, Q., Pan, Y., Fyshe, A., and White, M. Maxmin q-learning: Controlling the estimation bias of q-learning. In *International Conference on Learning Representations*, 2020.

Li, L., Walsh, T. J., and Littman, M. L. Towards a unified theory of state abstraction for mdps. *ISAIM*, 4:5, 2006.

Li, Z. and Hou, X. Mixing update q-value for deep reinforcement learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–6. IEEE, 2019.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Lin, Z., Zhao, T., Yang, G., and Zhang, L. Episodic memory deep q-networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 2433–2439, 2018.

Liu, Y., Swaminathan, A., Agarwal, A., and Brunskill, E. Provably good batch reinforcement learning without great exploration. In *Advances in Neural Information Processing Systems*, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Moore, A. W. and Atkeson, C. G. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine learning*, 13(1):103–130, 1993.

Oh, J., Guo, Y., Singh, S., and Lee, H. Self-imitation learning. In *International Conference on Machine Learning*, pp. 3878–3887, 2018.

Pritzel, A., Uria, B., Srinivasan, S., Badia, A. P., Vinyals, O., Hassabis, D., Wierstra, D., and Blundell, C. Neural episodic control. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2827–2836, 2017.

Smith, J. E. and Winkler, R. L. The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322, 2006.

Song, Z., Parr, R., and Carin, L. Revisiting the softmax bellman operator: New benefits and new perspective. In *International Conference on Machine Learning*, pp. 5916–5925, 2019.

Stone, M. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133, 1974.

Thaler, R. H. The winner's curse. *The Journal of Economic Perspectives*, 2(1):191–202, 1988.

Thrun, S. and Schwartz, A. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School Hillsdale, NJ. Lawrence Erlbaum*, 1993.

Van Hasselt, H. Double q-learning. In *Advances in neural information processing systems*, pp. 2613–2621, 2010.

Van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003, 2016.

Watkins, C. *Learning from delayed rewards*. PhD thesis, King's College, Cambridge, 1989.

Zhang, Z., Pan, Z., and Kochenderfer, M. J. Weighted double q-learning. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 3455–3461, 2017.

Zhou, Z., Kearnes, S., Li, L., Zare, R. N., and Riley, P. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):1–10, 2019.

Zhu, G., Lin, Z., Yang, G., and Zhang, C. Episodic reinforcement learning with associative memory. In *International Conference on Learning Representations*, 2020.