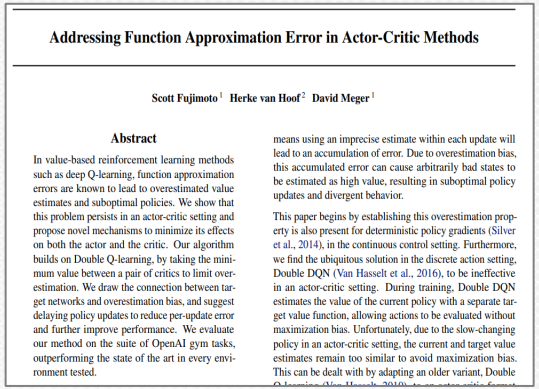
Twin Delayed DDPG (TD3)

: Addressing Function Approximation Error in Actor-Critic Methods

Sooyoung Lee

joo1.number02@gmail.com



Addressing Function Approximation Error in Actor-Critic Methods

Scott Fujimoto [1] Herke van Hoof [2] David Meger [1]

Abstract

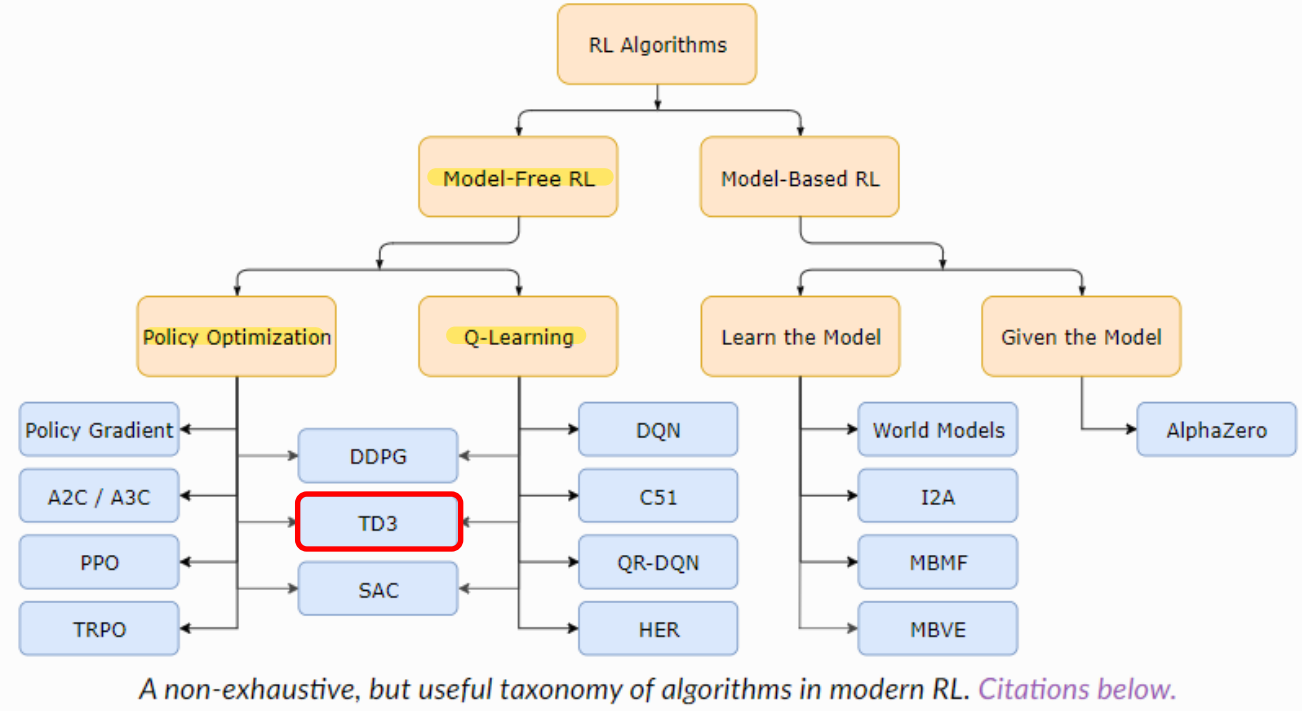
In value-based reinforcement learning methods such as deep Q-learning, function approximation errors are known to lead to overestimated value estimates and suboptimal policies. We show that this problem persists in an actor-critic setting and propose novel mechanisms to minimize its effects on both the actor and the critic. Our algorithm builds on Double Q-learning, by taking the minimum value between a pair of critics to limit overestimation. We draw the connection between target networks and overestimation bias, and suggest delaying policy updates to reduce per-update error and further improve performance. We evaluate our method on the suite of OpenAI gym tasks, outperforming the state of the art in every environment tested.

means using an imprecise estimate within each update will lead to an accumulation of error. Due to overestimation bias, this accumulated error can cause arbitrarily bad states to be estimated as high value, resulting in suboptimal policy updates and divergent behavior.

This paper begins by establishing this overestimation property is also present for deterministic policy gradients (Silver et al., 2014), in the continuous control setting. Furthermore, we find the ubiquitous solution in the discrete action setting, Double DQN (Van Hasselt et al., 2016), to be ineffective in an actor-critic setting. During training, Double DQN estimates the value of the current policy with a separate target value function, allowing actions to be evaluated without maximization bias. Unfortunately, due to the slow-changing policy in an actor-critic setting, the current and target value estimates remain too similar to avoid maximization bias. This can be dealt with by adapting an older variant, Double

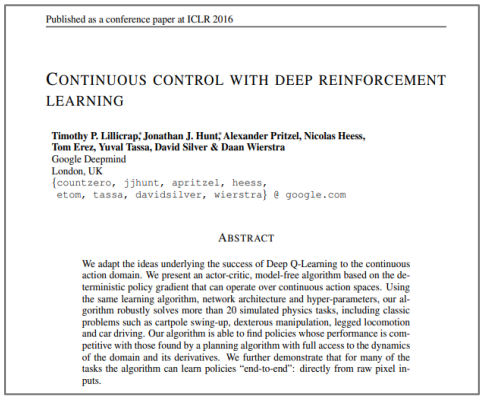
ICML 2018

Twin Delayed DDPG (TD3)



https://spinningup.openai.com/en/latest/index.html#

Twin Delayed DDPG (TD3)



Published as a conference paper at ICLR 2016

CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING

Timothy P. Lillicrap,* Jonathan J. Hunt,* Alexander Pritzel, Nicolas Heess,
Tom Erez, Yuval Tassa, David Silver & Daan Wierstra
Google Deepmind
London, UK
{countzero, jjhunt, apritzel, heess,
etom, tassa, davidsilver, wierstra} @ google.com

ABSTRACT

We adapt the ideas underlying the success of Deep Q-Learning to the continuous action domain. We present an actor-critic, model-free algorithm based on the deterministic policy gradient that can operate over continuous action spaces. Using the same learning algorithm, network architecture and hyper-parameters, our algorithm robustly solves more than 20 simulated physics tasks, including classic problems such as cartpole swing-up, dexterous manipulation, legged locomotion and car driving. Our algorithm is able to find policies whose performance is competitive with those found by a planning algorithm with full access to the dynamics of the domain and its derivatives. We further demonstrate that for many of the tasks the algorithm can learn policies "end-to-end": directly from raw pixel inputs.

DDPG
(Deep Deterministic Policy Gradient)

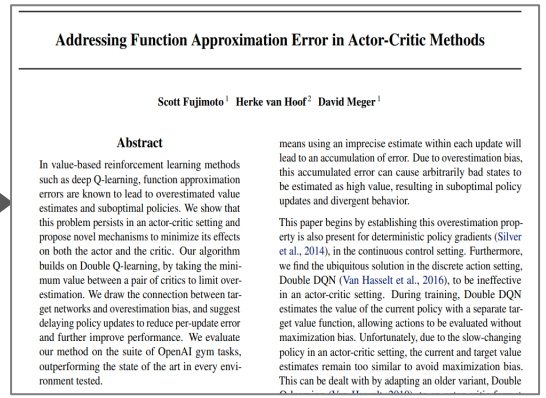
ICLR 2016

① Clipped Double-Q Learning

② "Delayed" Policy Updates

③ Target Policy Smoothing

α



Addressing Function Approximation Error in Actor-Critic Methods

Scott Fujimoto [1] Herke van Hoof [2] David Meger [1]

Abstract

In value-based reinforcement learning methods such as deep Q-learning, function approximation errors are known to lead to overestimated value estimates and suboptimal policies. We show that this problem persists in an actor-critic setting and propose novel mechanisms to minimize its effects on both the actor and the critic. Our algorithm builds on Double Q-learning, by taking the minimum value between a pair of critics to limit overestimation. We draw the connection between target networks and overestimation bias, and suggest delaying policy updates to reduce per-update error and further improve performance. We evaluate our method on the suite of OpenAI gym tasks, outperforming the state of the art in every environment tested.

means using an imprecise estimate within each update will lead to an accumulation of error. Due to overestimation bias, this accumulated error can cause arbitrarily bad states to be estimated as high value, resulting in suboptimal policy updates and divergent behavior.

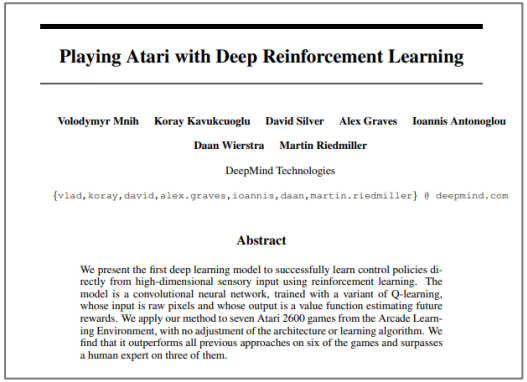
This paper begins by establishing this overestimation property is also present for deterministic policy gradients (Silver et al., 2014), in the continuous control setting. Furthermore, we find the ubiquitous solution in the discrete action setting, Double DQN (Van Hasselt et al., 2016), to be ineffective in an actor-critic setting. During training, Double DQN estimates the value of the current policy with a separate target value function, allowing actions to be evaluated without maximization bias. Unfortunately, due to the slow-changing policy in an actor-critic setting, the current and target value estimates remain too similar to avoid maximization bias. This can be dealt with by adapting an older variant, Double

TD3
(Twin Delayed DDPG)

ICML 2018

- TD3는 DDPG 성능 향상을 위한 방법론을 추가한 버전
- Actor-Critic Setting에서도 Overestimation Bias가 발생함을 증명하고, 해결법을 제시

Deep Deterministic Policy Gradient(DDPG)



Playing Atari with Deep Reinforcement Learning

Volodymyr Mnih Koray Kavukcuoglu David Silver Alex Graves Ioannis Antonoglou
Daan Wierstra Martin Riedmiller

DeepMind Technologies

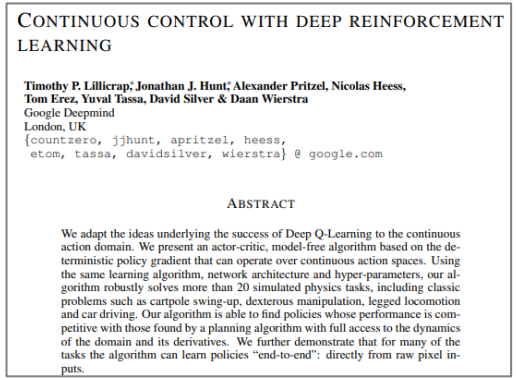
{vlad,koray,david,alex.graves,ioannis,daan,martin.riedmiller} @ deepmind.com

Abstract

We present the first deep learning model to successfully learn control policies directly from high-dimensional sensory input using reinforcement learning. The model is a convolutional neural network, trained with a variant of Q-learning, whose input is raw pixels and whose output is a value function estimating future rewards. We apply our method to seven Atari 2600 games from the Arcade Learning Environment, with no adjustment of the architecture or learning algorithm. We find that it outperforms all previous approaches on six of the games and surpasses a human expert on three of them.

DQN

2013



CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING

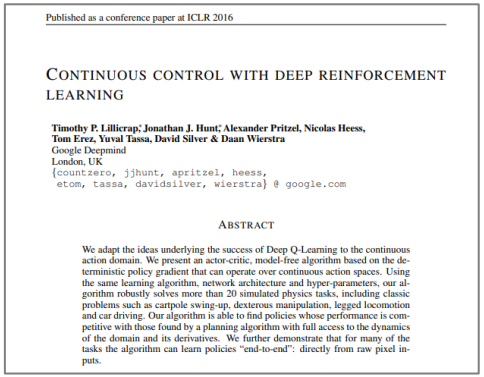
Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess,
Tom Erez, Yuval Tassa, David Silver & Daan Wierstra
Google Deepmind
London, UK
{countzero, jjhunt, apritzel, heess,
etom, tassa, davidsilver, wierstra} @ google.com

ABSTRACT

We adapt the ideas underlying the success of Deep Q-Learning to the continuous action domain. We present an actor-critic, model-free algorithm based on the deterministic policy gradient that can operate over continuous action spaces. Using the same learning algorithm, network architecture and hyper-parameters, our algorithm robustly solves more than 20 simulated physics tasks, including classic problems such as cartpole swing-up, dexterous manipulation, legged locomotion and car driving. Our algorithm is able to find policies whose performance is competitive with those found by a planning algorithm with full access to the dynamics of the domain and its derivatives. We further demonstrate that for many of the tasks the algorithm can learn policies "end-to-end": directly from raw pixel inputs.

DPG
(Deterministic
Policy Gradient)

ICML 2014



Published as a conference paper at ICLR 2016

CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess,
Tom Erez, Yuval Tassa, David Silver & Daan Wierstra
Google Deepmind
London, UK
{countzero, jjhunt, apritzel, heess,
etom, tassa, davidsilver, wierstra} @ google.com

ABSTRACT

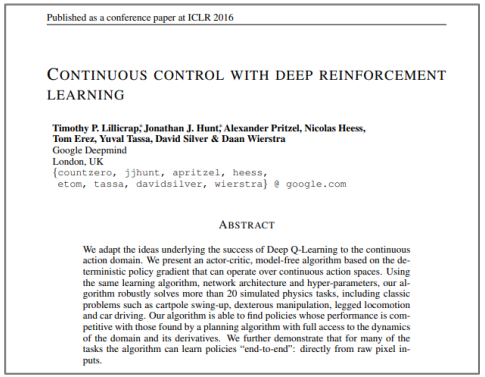
We adapt the ideas underlying the success of Deep Q-Learning to the continuous action domain. We present an actor-critic, model-free algorithm based on the deterministic policy gradient that can operate over continuous action spaces. Using the same learning algorithm, network architecture and hyper-parameters, our algorithm robustly solves more than 20 simulated physics tasks, including classic problems such as cartpole swing-up, dexterous manipulation, legged locomotion and car driving. Our algorithm is able to find policies whose performance is competitive with those found by a planning algorithm with full access to the dynamics of the domain and its derivatives. We further demonstrate that for many of the tasks the algorithm can learn policies "end-to-end": directly from raw pixel inputs.

DDPG
(Deep Deterministic
Policy Gradient)

ICLR 2016

- Deterministic PG에 Deep Learning을 적용
- DQN에서 사용하는 심층강화학습을 안정화 시키는 방법들을 사용(Replay Buffer, Target network 등)

Twin Delayed DDPG (TD3)



Published as a conference paper at ICLR 2016

CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING

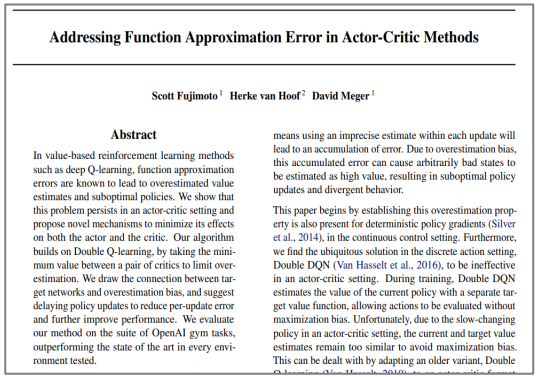
Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess,
Tom Erez, Yuval Tassa, David Silver & Daan Wierstra
Google Deepmind
London, UK
{countzero, jjhunt, apritzel, heess,
etom, tassa, davidsilver, wierstra} @ google.com

ABSTRACT

We adapt the ideas underlying the success of Deep Q-Learning to the continuous action domain. We present an actor-critic, model-free algorithm based on the deterministic policy gradient that can operate over continuous action spaces. Using the same learning algorithm, network architecture and hyper-parameters, our algorithm robustly solves more than 20 simulated physics tasks, including classic problems such as cartpole swing-up, dexterous manipulation, legged locomotion and car driving. Our algorithm is able to find policies whose performance is competitive with those found by a planning algorithm with full access to the dynamics of the domain and its derivatives. We further demonstrate that for many of the tasks the algorithm can learn policies "end-to-end": directly from raw pixel inputs.

DDPG
(Deep Deterministic Policy Gradient)

ICLR 2016



Addressing Function Approximation Error in Actor-Critic Methods

Scott Fujimoto [1] Herke van Hoof [2] David Meger [1]

Abstract

In value-based reinforcement learning methods such as deep Q-learning, function approximation errors are known to lead to overestimated value estimates and suboptimal policies. We show that this problem persists in an actor-critic setting and propose novel mechanisms to minimize its effects on both the actor and the critic. Our algorithm builds on Double Q-learning, by taking the minimum value between a pair of critics to limit overestimation. We draw the connection between target networks and overestimation bias, and suggest delaying policy updates to reduce per-update error and further improve performance. We evaluate our method on the suite of OpenAI gym tasks, outperforming the state of the art in every environment tested.

means using an imprecise estimate within each update will lead to an accumulation of error. Due to overestimation bias, this accumulated error can cause arbitrarily bad states to be estimated as high value, resulting in suboptimal policy updates and divergent behavior.

This paper begins by establishing this overestimation property is also present for deterministic policy gradients (Silver et al., 2014), in the continuous control setting. Furthermore, we find the ubiquitous solution in the discrete action setting, Double DQN (Van Hasselt et al., 2016), to be ineffective in an actor-critic setting. During training, Double DQN estimates the value of the current policy with a separate target value function, allowing actions to be evaluated without maximization bias. Unfortunately, due to the slow-changing policy in an actor-critic setting, the current and target value estimates remain too similar to avoid maximization bias. This can be dealt with by adapting an older variant, Double

TD3
(Twin Delayed DDPG)

ICML 2018

① **Actor-Critic**

② **Continuous action spaces**

③ **Deterministic Policy**

④ **Off-policy algorithm**

1. Introduction

- In reinforcement learning problems with discrete action spaces, the issue of value overestimation as a result of function approximation errors is well-studied.
- In this paper, we show overestimation bias and the accumulation of error in temporal difference methods are present in an actor-critic setting.
- This inaccuracy is further exaggerated by the nature of temporal difference learning (Sutton, 1988). Using an imprecise estimate within each update will lead to an accumulation of error.

1. Introduction

- This paper contains a number of components that address variance reduction.
- First, we show that target networks, a common approach in deep Q-learning methods, are critical for variance reduction by reducing the accumulation of errors.
- Second, to address the coupling of value and policy, we propose delaying policy updates until the value estimate has converged.
- Finally, we introduce a novel regularization strategy, where a SARSA-style update bootstraps similar action estimates to further reduce variance.

3. Background

Actor

$$R_t = \sum_{i=t}^{T} \gamma^{i-t} r(s_i, a_i)$$ Definition of Return

$$J(\phi) = \mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi} [R_0]$$ Objective Function of π_ϕ

$$Q^\pi(s, a) = \mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi} [R_t | s, a]$$ Expected Return

$$\nabla_\phi J(\phi) = \mathbb{E}_{s \sim p_\pi} \left[\nabla_a Q^\pi(s, a)|_{a=\pi(s)} \nabla_\phi \pi_\phi(s) \right]$$ Deterministic policy gradient algorithm (Silver et al., 2014)

Critic

$$Q^\pi(s, a) = r + \gamma \mathbb{E}_{s', a'} \left[Q^\pi(s', a') \right], \quad a' \sim \pi(s')$$ Bellman equation

$$Q_\theta(s, a) \qquad Q_{\theta'}(s, a)$$

$$y = r + \gamma Q_{\theta'}(s', a'), \quad a' \sim \pi_{\phi'}(s')$$ Target y

$$\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$$ Updated periodically

4. Overestimation Bias

- While in the discrete action setting overestimation bias is an obvious artifact from the analytical maximization, the presence and effects of overestimation bias is less clear in an actor-critic setting where the policy is updated via gradient descent.
- We begin by proving that the value estimate in deterministic policy gradients will be an overestimation under some basic assumptions in Section 4.1 and then propose a clipped variant of Double Q-learning in an actor-critic setting to reduce overestimation bias in Section 4.2.
  → Section 4.1 proving overestimation
  → Section 4.2 clipped variant of Double Q-learning

4.1. Overestimation Bias in Actor-Critic

- In this section we assume the policy is updated using the deterministic policy gradient, and show that the update induces overestimation in the value estimate.
- ϕtrue the parameters from the hypothetical actor update with respect to the true underlying value function Qπ (s, a) (which is not known during learning)

$$\phi_{\text{approx}} = \phi + \frac{\alpha}{Z_1}\mathbb{E}_{s\sim p_\pi}\left[\nabla_\phi \pi_\phi(s)\nabla_a Q_\theta(s,a)|_{a=\pi_\phi(s)}\right]$$

$$\phi_{\text{true}} = \phi + \frac{\alpha}{Z_2}\mathbb{E}_{s\sim p_\pi}\left[\nabla_\phi \pi_\phi(s)\nabla_a Q^\pi(s,a)|_{a=\pi_\phi(s)}\right]$$

$$\mathbb{E}\left[Q_\theta(s, \pi_{\text{approx}}(s))\right] \geq \mathbb{E}\left[Q^\pi(s, \pi_{\text{approx}}(s))\right]$$

As the gradient direction is a local maximizer, there exists ϵ_1 sufficiently small such that if $\alpha \leq \epsilon_1$ then the *approximate* value of π_{approx} will be bounded below by the *approximate* value of π_{true}:

$$\mathbb{E}\left[Q_\theta(s, \pi_{\text{approx}}(s))\right] \geq \mathbb{E}\left[Q_\theta(s, \pi_{\text{true}}(s))\right]. \quad (5)$$

Conversely, there exists ϵ_2 sufficiently small such that if $\alpha \leq \epsilon_2$ then the *true* value of π_{approx} will be bounded above by the *true* value of π_{true}:

$$\mathbb{E}\left[Q^\pi(s, \pi_{\text{true}}(s))\right] \geq \mathbb{E}\left[Q^\pi(s, \pi_{\text{approx}}(s))\right]. \quad (6)$$

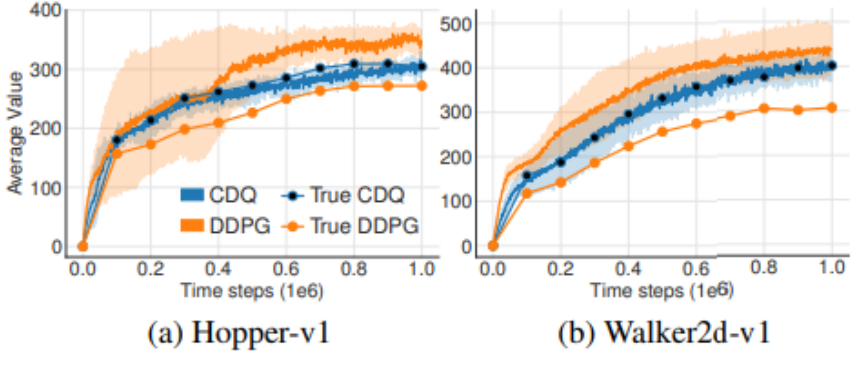
If in expectation the value estimate is at least as large as the *true* value with respect to ϕ_{true}, $\mathbb{E}\left[Q_\theta(s, \pi_{\text{true}}(s))\right] \geq \mathbb{E}\left[Q^\pi(s, \pi_{\text{true}}(s))\right]$, then Equations (5) and (6) imply that if $\alpha < \min(\epsilon_1, \epsilon_2)$, then the value estimate will be overestimated:

$$\mathbb{E}\left[Q_\theta(s, \pi_{\text{approx}}(s))\right] \geq \mathbb{E}\left[Q^\pi(s, \pi_{\text{approx}}(s))\right]. \quad (7)$$

4.1. Overestimation Bias in Actor-Critic

Does this theoretical overestimation occur in practice for state-of-the-art methods?

- We graph the average value estimate over 10000 states and compare it to an estimate of the true value.
- A very clear overestimation bias occurs from the learning procedure, which contrasts with the novel method, Clipped Double Q-learning, which greatly reduces overestimation by the critic

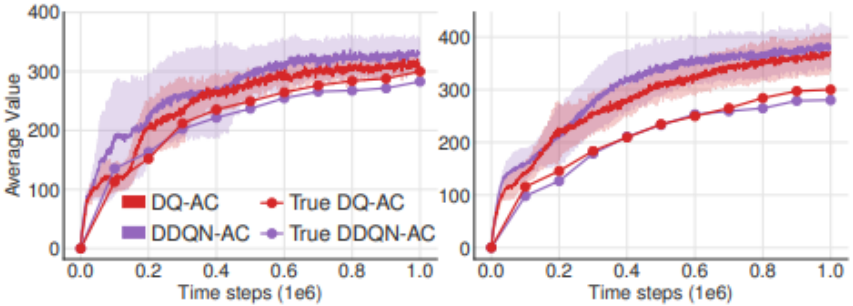


(a) Hopper-v1 (b) Walker2d-v1

Figure 1. Measuring overestimation bias in the value estimates of DDPG and our proposed method, Clipped Double Q-learning (CDQ), on MuJoCo environments over 1 million time steps.

4.2. Clipped Double Q-Learning for Actor-Critic

- The actor-critic Double DQN suffers from a similar overestimation as DDPG.
- While Double Q-learning is more effective, it does not entirely eliminate the overestimation.



(a) Hopper-v1 (b) Walker2d-v1

Figure 2. Measuring overestimation bias in the value estimates of actor critic variants of Double DQN (DDQN-AC) and Double Q-learning (DQ-AC) on MuJoCo environments over 1 million time steps.

4.2. Clipped Double Q-Learning for Actor-Critic

Algorithm 1 Double Q-learning

Initialize Q^A,Q^B,s
repeat
  Choose a, based on $Q^A(s,\cdot)$ and $Q^B(s,\cdot)$, observe r, s'
  Choose (e.g. random) either UPDATE(A) or UPDATE(B)
  if UPDATE(A) **then**
    Define $a^* = \arg\max_a Q^A(s',a)$
    $Q^A(s,a) \leftarrow Q^A(s,a) + \alpha(s,a)\left(r + \gamma Q^B(s',a^*) - Q^A(s,a)\right)$
  else if UPDATE(B) **then**
    Define $b^* = \arg\max_a Q^B(s',a)$
    $Q^B(s,a) \leftarrow Q^B(s,a) + \alpha(s,a)(r + \gamma Q^A(s',b^*) - Q^B(s,a))$
  end if
  $s \leftarrow s'$
until end

Double Q-learning
(Van Hasselt, 2010)

Algorithm 1 : Double Q-learning (Hasselt et al., 2015)

Initialize primary network Q_θ, target network $Q_{\theta'}$, replay buffer $\mathcal{D}$, $\tau << 1$
for each iteration **do**
  for each environment step **do**
    Observe state s_t and select $a_t \sim \pi(a_t, s_t)$
    Execute a_t and observe next state s_{t+1} and reward $r_t = R(s_t, a_t)$
    Store (s_t, a_t, r_t, s_{t+1}) in replay buffer $\mathcal{D}$
  for each update step **do**
    sample $e_t = (s_t, a_t, r_t, s_{t+1}) \sim \mathcal{D}$
    Compute target Q value:
      $Q^*(s_t, a_t) \approx r_t + \gamma\, Q_\theta(s_{t+1}, argmax_{a'} Q_{\theta'}(s_{t+1}, a'))$
    Perform gradient descent step on $(Q^*(s_t, a_t) - Q_\theta(s_t, a_t))^2$
    Update target network parameters:
      $\theta' \leftarrow \tau * \theta + (1 - \tau) * \theta'$

Double DQN
(Van Hasselt et al., 2015)

: Q' for action selection and Q for action evaluation

4.2. Clipped Double Q-Learning for Actor-Critic

- To address this problem, we propose to simply upper-bound the less biased value estimate $Q_{\theta 2}$ by the biased estimate $Q_{\theta 1}$.
- This results in taking the minimum between the two estimates, to give the target update of our Clipped Double Q-learning algorithm:

$$y_1 = r + \gamma \min_{i=1,2} Q_{\theta'_i}(s', \pi_{\phi_1}(s'))$$

5. Addressing Variance

- In this section we emphasize the importance of minimizing error at each update,
  → Propose modifications to the learning procedure of actor-critic for variance reduction.

5.1. Accumulating Error

- Due to the temporal difference update, where an estimate of the value function is built from an estimate of a subsequent state, there is a build up of error.
  → Each update leaves some amount of residual TD-error δ(s, a)
- While it is reasonable to expect small error for an individual update, these estimation errors can accumulate, resulting in the potential for large overestimation bias and suboptimal policy updates.

$$Q_\theta(s,a) = r + \gamma \mathbb{E}[Q_\theta(s',a')] - \delta(s,a).$$

$$\begin{aligned} Q_\theta(s_t,a_t) &= r_t + \gamma \mathbb{E}[Q_\theta(s_{t+1},a_{t+1})] - \delta_t \\ &= r_t + \gamma \mathbb{E}\left[r_{t+1} + \gamma \mathbb{E}\left[Q_\theta(s_{t+2},a_{t+2}) - \delta_{t+1}\right]\right] - \delta_t \\ &= \mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi}\left[\sum_{i=t}^{T} \gamma^{i-t}(r_i - \delta_i)\right]. \end{aligned} \tag{12}$$

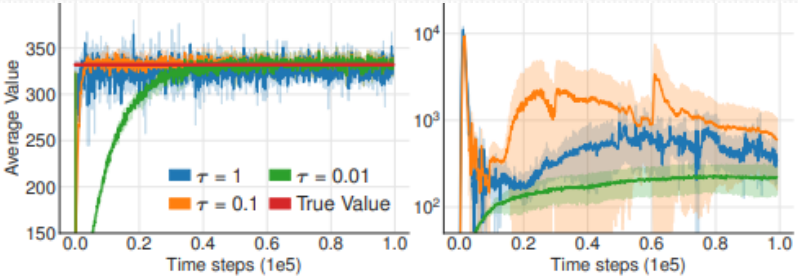
5.2. Target Networks and Delayed Policy Updates

- In this section we examine the relationship between target networks and function approximation error, and show the use of a stable target reduces the growth of error.
- Target networks are a well-known tool to achieve stability in deep reinforcement learning.
- Without a fixed target, each update may leave residual error which will begin to accumulate.

5.2. Target Networks and Delayed Policy Updates

- We examine the learning behavior with and without target networks on both the critic and actor.
- In (a) we compare the behavior with a fixed policy and in (b) we examine the value estimates with a policy that continues to learn, trained with the current value estimate. The target networks use a slow-moving update rate, parametrized by τ .

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta'$$



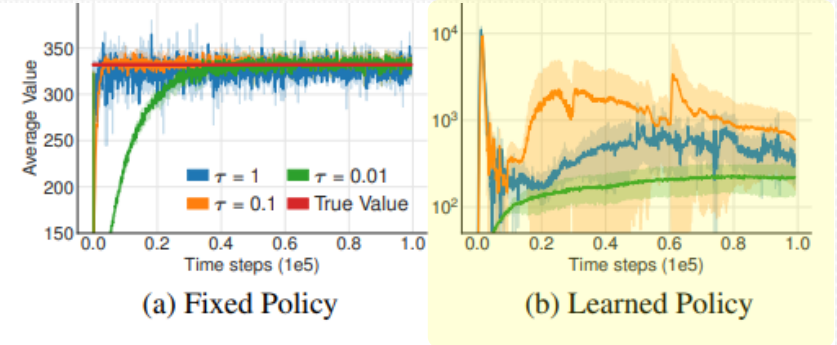
(a) Fixed Policy (b) Learned Policy

Figure 3. Average estimated value of a randomly selected state on Hopper-v1 without target networks, ($\tau = 1$), and with slow-updating target networks, ($\tau = 0.1, 0.01$), with a fixed and a learned policy.

5.2. Target Networks and Delayed Policy Updates

- When the policy is trained with the current value estimate, the use of fast-updating target networks results in highly divergent behavior.
- Figure 3, as well as Section 4, suggest failure can occur due to the interplay between the actor and critic updates. Value estimates diverge through overestimation when the policy is poor, and the policy will become poor if the value estimate itself is inaccurate.

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta'$$



(a) Fixed Policy (b) Learned Policy

Figure 3. Average estimated value of a randomly selected state on Hopper-v1 without target networks, ($\tau = 1$), and with slow-updating target networks, ($\tau = 0.1, 0.01$), with a fixed and a learned policy.

5.2. Target Networks and Delayed Policy Updates

- The less frequent policy updates that do occur will use a value estimate with lower variance, and in principle, should result in higher quality policy updates

5.3. Target Policy Smoothing Regularization

- When updating the critic, a learning target using a deterministic policy is highly susceptible to inaccuracies induced by function approximation error, increasing the variance of the target.
- → This induced variance can be reduced through regularization.
- Our approach enforces the notion that similar actions should have similar value.
- → We propose that fitting the value of a small area around the target action.

$$y = r + \mathbb{E}_{\epsilon}\left[Q_{\theta'}(s', \pi_{\phi'}(s') + \epsilon)\right]$$

- In practice, we can approximate this expectation over actions by adding a small amount of random noise to the target policy and averaging over mini-batches.

$$y = r + \gamma Q_{\theta'}(s', \pi_{\phi'}(s') + \epsilon),$$
$$\epsilon \sim \text{clip}(\mathcal{N}(0, \sigma), -c, c),$$

- where the added noise is clipped to keep the target close to the original action

Algorithm. TD3

Algorithm 1 TD3

Initialize critic networks Q_{θ_1}, Q_{θ_2}, and actor network π_ϕ
with random parameters θ_1, θ_2, ϕ
Initialize target networks $\theta'_1 \leftarrow \theta_1$, $\theta'_2 \leftarrow \theta_2$, $\phi' \leftarrow \phi$
Initialize replay buffer $\mathcal{B}$
for $t = 1$ **to** T **do**
  Select action with exploration noise $a \sim \pi_\phi(s) + \epsilon$,
  $\epsilon \sim \mathcal{N}(0, \sigma)$ and observe reward r and new state s'
  Store transition tuple (s, a, r, s') in $\mathcal{B}$

  Sample mini-batch of N transitions (s, a, r, s') from $\mathcal{B}$
  $\tilde{a} \leftarrow \pi_{\phi'}(s') + \epsilon, \quad \epsilon \sim \text{clip}(\mathcal{N}(0, \tilde{\sigma}), -c, c)$
  $y \leftarrow r + \gamma \min_{i=1,2} Q_{\theta'_i}(s', \tilde{a})$
  Update critics $\theta_i \leftarrow \text{argmin}_{\theta_i} N^{-1} \sum (y - Q_{\theta_i}(s, a))^2$
  if t mod d **then**
    Update ϕ by the deterministic policy gradient:
    $\nabla_\phi J(\phi) = N^{-1} \sum \nabla_a Q_{\theta_1}(s, a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$
    Update target networks:
    $\theta'_i \leftarrow \tau\theta_i + (1 - \tau)\theta'_i$
    $\phi' \leftarrow \tau\phi + (1 - \tau)\phi'$
  end if
end for

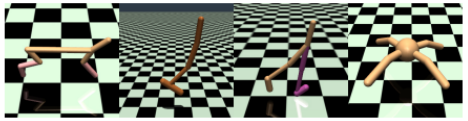
OU process

Target policy smoothing

Clipped double Q-learning

Delayed policy updates

6. Experiments



(a) (b) (c) (d)

Figure 4. Example MuJoCo environments (a) HalfCheetah-v1, (b) Hopper-v1, (c) Walker2d-v1, (d) Ant-v1.

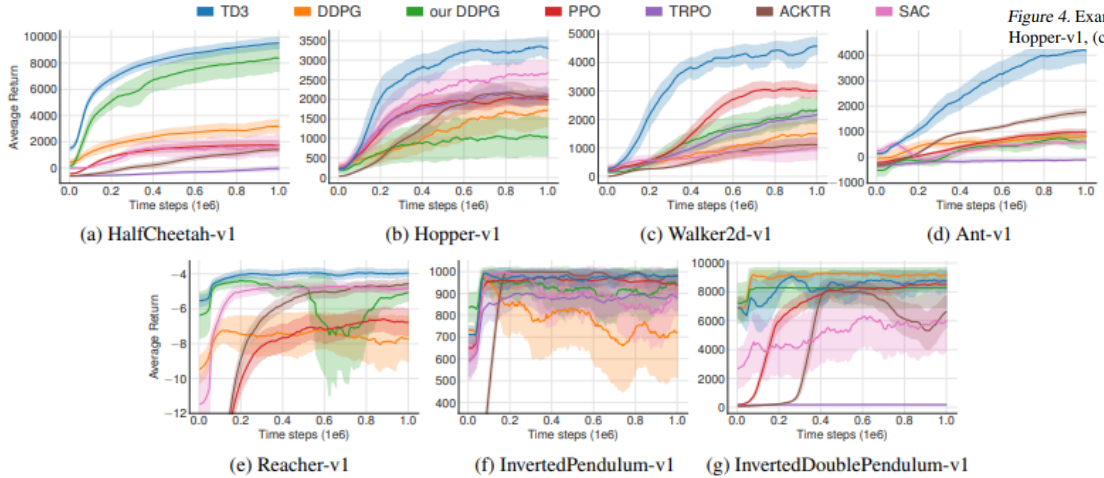


Figure 5. Learning curves for the OpenAI gym continuous control tasks. The shaded region represents half a standard deviation of the average evaluation over 10 trials. Curves are smoothed uniformly for visual clarity.

Table 1. Max Average Return over 10 trials of 1 million time steps. Maximum value for each task is bolded. ± corresponds to a single standard deviation over trials.

Environment	TD3	DDPG	Our DDPG	PPO	TRPO	ACKTR	SAC
HalfCheetah	**9636.95 ± 859.065**	3305.60	8577.29	1795.43	-15.57	1450.46	2347.19
Hopper	**3564.07 ± 114.74**	2020.46	1860.02	2164.70	2471.30	2428.39	2996.66
Walker2d	**4682.82 ± 539.64**	1843.85	3098.11	3317.69	2321.47	1216.70	1283.67
Ant	**4372.44 ± 1000.33**	1005.30	888.77	1083.20	-75.85	1821.94	655.35
Reacher	**-3.60 ± 0.56**	-6.51	**-4.01**	-6.18	-111.43	-4.26	-4.44
InvPendulum	**1000.00 ± 0.00**	**1000.00**	**1000.00**	**1000.00**	985.40	**1000.00**	**1000.00**
InvDoublePendulum	**9337.47 ± 14.96**	**9355.52**	8369.95	8977.94	205.85	9081.92	8487.15

6.2. Ablation Studies

- We perform ablation studies to understand the contribution of each individual component: Clipped Double Q-learning (CDQ), delayed policy updates (DP) and target policy smoothing (TPS).
- The significance of each component varies task to task.
- The full algorithm outperforms every other combination in most tasks.

Table 2. Average return over the last 10 evaluations over 10 trials of 1 million time steps, comparing ablation over delayed policy updates (DP), target policy smoothing (TPS), Clipped Double Q-learning (CDQ) and our architecture, hyper-parameters and exploration (AHE). Maximum value for each task is bolded.

Method	HCheetah	Hopper	Walker2d	Ant
TD3	9532.99	**3304.75**	**4565.24**	**4185.06**
DDPG	3162.50	1731.94	1520.90	816.35
AHE	8401.02	1061.77	2362.13	564.07
AHE + DP	7588.64	1465.11	2459.53	896.13
AHE + TPS	9023.40	907.56	2961.36	872.17
AHE + CDQ	6470.20	1134.14	3979.21	3818.71
TD3 - DP	9590.65	2407.42	**4695.50**	3754.26
TD3 - TPS	8987.69	2392.59	4033.67	**4155.24**
TD3 - CDQ	9792.80	1837.32	2579.39	849.75
DQ-AC	9433.87	1773.71	3100.45	2445.97
DDQN-AC	**10306.90**	2155.75	3116.81	1092.18

Reference

- https://arxiv.org/pdf/1802.09477.pdf
- https://spinningup.openai.com/en/latest/algorithms/td3.html
- https://towardsdatascience.com/double-deep-q-networks-905dd8325412