Conservative Policy Gradient in Multi-critic Setting

Bao Xi[1,2], Rui Wang[2], Shuo Wang[1,2,3], Tao Lu[2], and Yinghao Cai[2]
1. *University of Chinese Academy of Sciences,* Beijing, China;
2. *State Key Laboratory of Management and Control for Complex System, Institute of Automation,* Beijing, China;
3. *Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences,* Shanghai, China.
Email: {$xibao2016, rwang5212, shuo.wang, tao.lu, yinghao.cai$}@ia.ac.cn

***Abstract*—Twin Delayed Deep Deterministic policy gradient algorithm (TD3) addressed the overestimation bias problem by adopting a clipped Double Q-Learning method. As the two Q networks are different, the updation of a policy to maximize one Q function might minimize another, which is called inconsistency in this paper. Therefore, we propose an algorithm based on TD3, conservative policy gradient (CPG), that optimizes the policy with respect to the lower bound of the two Q functions to deal with the inconsistancy. In Q function learning, one-step estimate is usually used in target value estimation. However, due to the constantly changing target networks, there will be fluctuations in the estimation. As the target Q function is changing slowly, we combine the one-step estimate with zero-step estimate to avoid sharp changes. The experimental results illustrate that CPG outperforms TD3 and some other reinforcement learning methods on multiple MuJoCo benchmarks.**

***Index Terms*—inconsistancy, stablility, Q learning, policy gradient**

I. Introduction

In recent years, deep reinforcement learning (DRL) has achieved great successes in a wide range of challenging problems, such as video games [1], [2] and robotics [3]. There are two main categories of DRL methods: off-policy [4] and off-policy methods [5]. In the training process, the off-policy methods reuse previously collected transitions while the on-policy methods can only use the data collected by the current policy. Thus the sample efficiency of the off-policy methods are higher.

Deep Q learning (DQN) [6] is one of the off-policy methods, which shows success in the majority of Atari games. DQN is also a value function based method, and the max operator in value target estimate results in overestimation bias. To tackle with the continuous action space problems, deep deterministic policy gradient (DDPG) [7], [8] is proposed, and it is a Q learning based method. The policy is updated to maximize the current Q function. The Q funtion and policy are also called critic and actor respectively in DDPG. Since the policy ouputs the action directly, DDPG becomes a popular method in domains with continous action space. In DDPG, the overestimation bias of the critic can pass to the policy through policy gradient. Many algorithms have been proposed to address the overestimation bias in Q learning. Double DQN [9] addresses the overesitmation problem by using two independent Q functions. However, as the actor is updated slowly, the target critics are too similar with the current critics to avoid overestimation bias. Zheng Z et al. [10] adopts a multi-head network to describe the critic and estimates the target action value by the weighted sum of the critic heads. However, it is hard to find appropriate weights for different tasks or environments. Fujimoto et al. extended DDPG to Twin Delayed Deep Deterministic policy gradient algorithm (TD3) [11], which estimates the target Q value using the minimum of two target Q value, called clipped double Q learning.

Although the value target estimate in TD3 cannot introduce additional overestimation, the two critics are not accurate everywhere in the state-action space. In addition, due to different initialization, the network landscapes of the two critics are slightly different. Therefore, the updation of the actor with respect to one critic may be inconsistant with another. Specifically, the updation may maximize a critic but minimize another. Inspired by the EM algorithm [12] that maximize the lower bound of the likelyhood, we update the actor respect to the lower bound of the critics to deal with the inconsistancy problem.

The value target estimate is depend on the target actor and target critic. As the target actor is constantly updating, the next action used in the one-step value target estimate is also changing. Therefore, the updation of target actor may cause a big change in target value estimate, which would cause instablility in critic training. In this work, we utilize a weighted sum of the zero-step and one-step estimate of the target value to avoid sharp fluctuations. Since our method optimizes the lower bound of the critics and uses the 0-step estimate to stablize the training, it is a conservative method. Therefore, we call it Conservative Policy Gradient (CPG).

In this paper we evaluate our method on variety of OpenAI Gym's MuJoCo benchmark tasks [13]. The results of the experiment show that our method can obtain better performance than TD3 and DDPG.

II. Background

In this section, we introduce the notations in reinforcement learning problems and give a brief review about deep deterministic policy gradient (DDPG) and twin delayed deep deterministic policy gradient (TD3).

This work was supported in part by the National Natural Science Foundation of China under Grant 61773378, U1713222, and U1806204; in part by Equipment Pre-research Field Fund under Grant 61403120407, and in part by the Foundation for Innovative Research Groups of the National Natural Science Foundation of China under Grant 61421004.

A. Notations

The reinforcement learning problems are usually formalized as a Markov decision process (MDP), characterized by a tuple $\mathbb{M} = \{\mathcal{S}, \mathcal{A}, P, r, \gamma\}$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action space of the MDP, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ and $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ are the environment dynamics and reward function, and $\gamma \in (0, 1]$ is a discount factor.

An RL agent interact with the environment according to a policy $\pi : \mathcal{S} \times \mathcal{A} \mapsto (0, 1]$. During interaction, the agent receives rewards from the environment as feedbacks. The discounted sum of the rewards of an episode is defined as:

$$R_0 = \sum_{t=0}^{T} \gamma^t r(s_t, a_t) \tag{1}$$

where T is the length of the episode. The goal of RL is to find a policy which maximize the expectation of return:

$$J_\pi = \mathbb{E}_{s_t \sim \rho^\pi, a_t \sim \pi} \left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t) \right] \tag{2}$$

where ρ^π denotes the state visitation distribution of policy π.

An action-value function is often used in many reinforcement learning algorithms, which is used to represent the long-term return from a given state after taking an action:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{s_i \sim \rho^\pi, a_i \sim \pi} \left[r(s_t, a_t) + \sum_{i=t+1}^{T} \gamma^{i-t} r(s_i, a_i) \right] \tag{3}$$

The action-value function is also called Q-function in many literatures. The Q-function can be writen in a recursive format by Bellman Equation:

$$Q^\pi(s_t, a_t) = r(s_t, a_t) + \mathbb{E}_{a_{t+1} \sim \pi} \left[Q^\pi(s_{t+1}, a_{t+1}) \right] \tag{4}$$

B. DDPG

DDPG is a popular off-policy actor-critic method in continuous action space domain, which is consisted of a Q-function (critic) Q_θ, parameterized by θ, and a deterministic policy (actor) μ_ϕ, parameterized by ϕ. Besides, DDPG uses slowly updated critic and actor target networks to stabilize the training, which are represented by $Q_{\theta'}$ and $\mu_{\phi'}$ respectively.

The critic is learned using temporal difference (TD) learning, an update rule based on the Bellman equation. The loss function is given by the expectation of TD error:

$$\mathcal{L}(\theta) = \mathbb{E}_{a_i \sim \beta, s_i \sim \rho^\beta} \left[(y_i - Q_\theta(s_i, a_i))^2 \right] \tag{5}$$

where β and ρ^β are the behavior policy and the corresponding state visitation distribution, and the target Q value y_i is evaluated by:

$$y_i = r(s_i, a_i) + Q_{\theta'}(s_{i+1}, \mu_{\phi'}(s_{i+1})) \tag{6}$$

As (6) utilizes the next state, it is called 1-step estimate.

The actor is updated to increase the value of $Q_\theta(s_t, \mu_\phi(s_t))$ by appling the chain rule:

$$\nabla_\phi J(\phi) \approx \mathbb{E}_{s \sim \rho^\beta, a \sim \beta} \left[\nabla_a Q_\theta(s, a)|_{a=\mu(s)} \nabla_\phi \mu_\phi(s) \right] \tag{7}$$

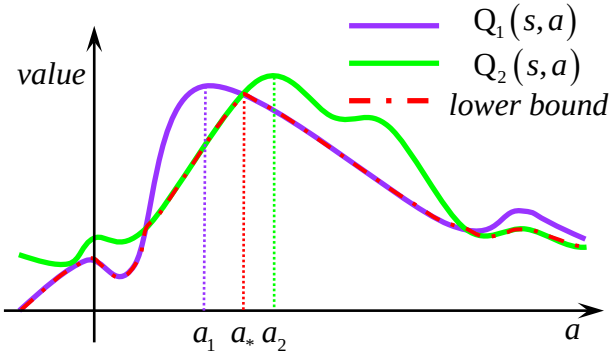

Fig. 1. An illustration of the inconsistancy of policy updation in two critic setting. As the target in Q learning is the same for two critics, Q_1 and Q_2 are roughly equal but slightly different in detail.If the actor is updated respect to one of the critic, the optimal action is a_1 or a_2. a_* is the result of updating with respect to the lower bound of the critics, which performs fairly good on both critics.

DDPG alternates between running a behavior policy to collect interaction experiences and updating the network parameters. The behavior policy is obtained by adding the policy an noise to allow action exploration.

$$\beta(s_t) = \mu_\phi(s_t) + \mathcal{N} \tag{8}$$

where $\mathcal{N}$ is an exploration noise in action space.

C. TD3

Overestimation bias is a property of Q-learning which is caused by maximization of a noisy value estimate. TD3 introduces a clipped variant of Double Q-learning to reduce overestimation bias. Specifically, TD3 adopts two critics, Q_{θ_1} and Q_{θ_2}, to get a less optimistic estimate of an action value by taking the minimum between two estimates. For a transition (s, a, r, s'), the target Q value y is given by:

$$y = r(s, a) + \min_{i=1,2} Q_{\theta_i'}(s', \mu_{\phi'}(s')) \tag{9}$$

The target Q value is used to train the two critics as in DDPG. (9) is also an one-step estimate of the target value.

There is only one actor in TD3, which is optimized with respect to Q_{θ_1}. The actor is updated at a lower frequency than the critic to ensure the TD-error remains small, which results in higher quality policy updates in practice.

III. Proposed Method

In actor-critic methods the actor is updated with respect to the critic. Although TD3 addressed the overestimation bias in Q-learning, the two critics still cannot be accurate everywhere in the state-action space.

In this section, we first analyze that the target Q value estimate in TD3 might be fluctuate violently and give a more stable estimate method. Then, we show that the updation of an actor with respect a critic will lead to inconsistent between the critics and then propose a method to take both critics into consideration by updating the actor with respect to the lower bound of the critics.

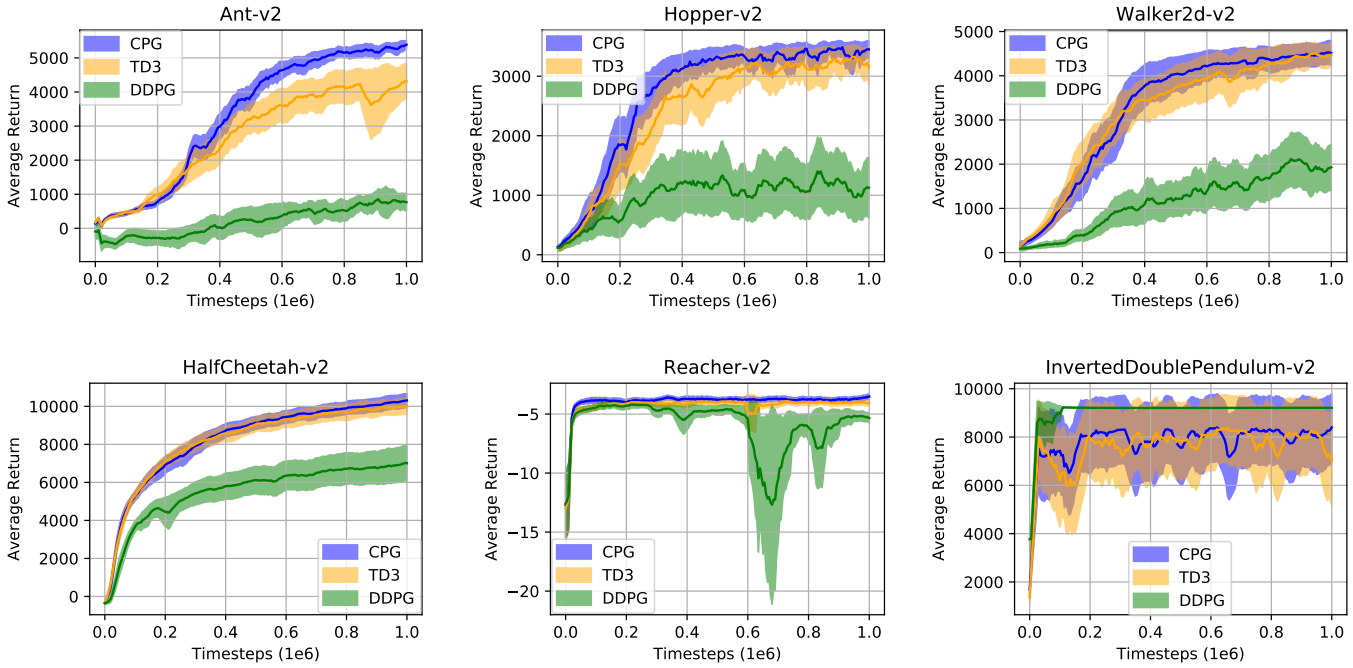

Fig. 2. The cumulative rewards for CPG, TD3 and DDPG method. The bold lines are averages of 10 independent trails with different random seeds on the six simulated environments, and the shaded area denotes 0.5 standard deviation of the rewards.

A. Stablelizing Q updation

As is show in (9) the target action value y is depend on the current target policy μ'. In the training process, the policy and target policy are constantly updating. In the k th iteration, for a transition (s, a, r, s'), the target value is given by:

$$y_k = r(s,a) + \min_{i=1,2} Q_{\theta'_{k-1,i}}(s', a_{k-1}) \tag{10}$$

where $a_{k-1} = \mu_{\phi'_{k-1}}(s') + \mathcal{N}$ is the ouput of the noisy target policy. Similarly, the target value in the $k-1$ th iteration is:

$$y_{k-1} = r(s,a) + \min_{i=1,2} Q_{\theta'_{k-2,i}}(s', a_{k-2}) \tag{11}$$

As the target actor is changing, the action a_{k-1} and a_{k-2} might be quite different. In addtion, the target critics are also changing. Therefore, the updating in target networks may lead to a great difference between the action value y_k and y_{k-1}, which would cause instablility in Q fucntion learning and actor updating.

To stablelizing the Q learning, we combine the 0-step and 1-step action value estimate to evaluate the target action value, which is given by:

$$y = \alpha \min_{i=1,2} Q_{\theta'_i}(s,a) + (1-\alpha)(r(s,a) + \min_{i=1,2} Q_{\theta'_i}(s', \mu_{\phi'}(s'))) \tag{12}$$

where $Q_{\theta'_i}(s,a)$ is the 0-step estimate of state-action pair (s,a).

estimate (12) avoids sudden changes in target action value, which can benefits the stability in Q learning. While this target value may casue slow updation, this is perferable to instablility.

B. Clipped Double Q Policy Gradient

In the k th iteration of training, the parameters of the actor $\mu_\phi(s)$, are updated along the gradient direction to maximize $Q_{\theta_1}(s, \mu_\phi(s))$:

$$\phi_k = \phi_{k-1} + \xi \mathbb{E}_{s\sim\rho^\beta}\left[\nabla_a Q_{\theta_1}(s,a)|_{a=\mu(s)} \nabla_\phi \mu_\phi(s)\right] \tag{13}$$

where α is a small updation stepsize. There exists ϵ sufficiently small such that if $\alpha \leq \epsilon$ the following inequality holds.

$$Q_{\theta_1}(s, \mu_{\phi_k}(s)) \geq Q_{\theta_1}(s, \mu_{\phi_{k-1}}(s)) \tag{14}$$

As the gradient in Eq.13 is respect to critic Q_{θ_1}, the parameters ϕ_k and ϕ_{k-1} cannot ensure the next inequality holds.

$$Q_{\theta_2}(s, \mu_{\phi_k}(s)) \geq Q_{\theta_2}(s, \mu_{\phi_{k-1}}(s)) \tag{15}$$

However, a reasonable policy updation should make both $Q_{\theta_1}(s, \mu_{\phi_k}(s))$ and $Q_{\theta_2}(s, \mu_{\phi_k}(s))$ higher. Therefore, the updation in (13) is inconsistancy. The inconsistent is a result of training the actor by just one critic, which is a common problem in multi-critic methods as is illustrated in Fig. 1. If an actor is trained with respect to Q_1 or Q_2, the optimal action is a_1 or a_2. As the critic are not accurate every where in the state-action space, a_* that maximizes the lower bound of the critics is more reasonable.

Inspired by the Expectation Maximum (EM) algothm that optimize the lower bound of the likelyhood, we update the actor with respect to the lower bound of the Q functions, $Q_{lb}(s,a)$, to take both critics into account:

$$Q_{lb}(s,a) = \min_{i=1,2} Q_{\theta_i}(s,a) \tag{16}$$

Q_{lb} is then used to optimize the actor:

$$\phi_k = \phi_{k-1} + \xi \mathbb{E}_{s\sim\rho^\beta}\left[\nabla_a Q_{lb}(s,a)|_{a=\mu(s)} \nabla_\phi \mu_\phi(s)\right] \quad (17)$$

As Q_{lb} is the lower bound of Q_{θ_1} and Q_{θ_2}, this updation makes the worst performance of the actor better than before.

IV. Simulated Experiments

In this section, we first introduce the experimental settings and evaluate the proposed algorithm on popular benchmarks for continuous robotic environments in MuJoCo simulator from OpenAI Gym. Several experimental results are given with comparison to DDPG and TD3.

A. Environmental setup

We built the code of CPG based on the implementation of TD3 provided by the Fujimoto et al. [11]. The DDPG for comparison in this work is also the implementation given by Fujimoto, which is better than the original DDPG. Unless specified otherwise, these three methods share the same network architectures and hyperparamters that are used in Fujimoto's implementation.

In CPG, the standard deviation of the exploration Gaussian noise is initialized to 0.3 and decreased to 0.1 with respect to decay rate 0.95 and decay period 10 thausand time steps. We utilized two different value for weight α in equation (12). We set α to 0.5 in simulated environment Ant-$v2$ and $HalfCheetah$-$v2$, and 0.2 in the rest environments. The network initialization scheme is also different from that used in TD3. The differences lie in initializing the last layer of the two critic networks. The bias and weights of the last layer of critic Q_1 are sampled from a Gaussian distribution $\mathcal{N}(0, 0.1)$ and those of critic Q_2 are sampled from a Gaussian distribution $\mathcal{N}(0, 0.001)$. In the first 100 thausand time steps, the actor is updated with respect to Q_1. Then the actor is trained with respect to the lower bound of the critics.

B. Experiment Results

Each task is trained for 1 million time steps with evaluations every 5 thausand time steps, where each evaluation records the averaged return over 10 episodes with no exploration noise. We measure the performance by the averaged return and evaluate the stability by the standard deviation over the evaluations with 10 different random seeds. Figure 2 represent the mean return by lines and std return by shaded areas.

In CPG, the policy updation respect to the lower bound of critics can prevent the overestimation bias passing to the actor and take both critics into account. Therefore, it can find a more reasonable policy than updating respect to a single critic. From Figure 2, one can see that CPG obtains higher averaged return than DDPG and TD3 on Ant-$v2$, $Hopper$-$v2$, $Walker$-$v2$. As these three environments are unstable, the performance of DDPG is the worst. CPG outperforms TD3 slightly on $HalfCheetah$-$v2$, $Reacher$-$v2$. Due to the 0-step value estimate is used in target value estimation, the variance of returns of CPG is obviously lower than that of DDPG and TD3 as is shown in Figure 2. The experimental results verify the effectiveness of CPG.

V. Conclusion

In this paper, we propose Conservative Policy Gradient (CPG), which updates the actor with respect to the lower bound of the critics to deal with the inconsistancy problem in multi-critic settings. We also adopt a combination of zero-step return and one-step return to estimate the target value to stablize the training. We test the CPG on the on Gym's MuJoCo benchmarks and the experimental results show that CPG can obtain high averaged return and low variance on most tasks, which confirm the effectiveness of CPG.

References

[1] Silver D, Huang A, Maddison C J, et al. "Mastering the game of Go with deep neural networks and tree search[J]," nature, vol. 529(7587): 484, 2016.

[2] Lee D, Tang H, Zhang J O, et al., "Modular Architecture for StarCraft II with Deep Reinforcement Learning," Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference. 2018.

[3] Haarnoja T, Pong V, Zhou A, et al. "Composable deep reinforcement learning for robotic manipulation," 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 6244-6251, 2018.

[4] van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J., "Deep reinforcement learning and the deadly triad," CoRR, abs/1812.02648, 2018.

[5] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O., "Proximal policy optimization algorithms," CoRR, abs/1707.06347, 2017.

[6] Mnih V, Kavukcuoglu K, Silver D, et al. "Playing atari with deep reinforcement learning[J]," arXiv preprint arXiv:1312.5602, 2013.

[7] Silver D, Lever G, Heess N, et al. "Deterministic policy gradient algorithms[C]," International Conference on International Conference on Machine Learning. JMLR.org, 2014.

[8] Lillicrap T P , Hunt J J , Pritzel A , et al. "Continuous control with deep reinforcement learning[J]." Computer Science, 2015.

[9] Hado V. Hasselt, "Double Q-learning," Advances in Neural Information Processing Systems, pp. 2613-2621, 2010.

[10] Zheng Z, Yuan C, Lin Z, et al. "Self-Adaptive Double Bootstrapped DDPG," Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence Organization, pp.3198–3204, 2018.

[11] Fujimoto, Scott, Hoof, and Herke, et al. "Addressing Function Approximation Error in Actor-Critic Methods," International Conference on Machine Learning, pp. 1582–1591, 2018.

[12] Dempster A P, Laird N M, Rubin D B., "Maximum likelihood from incomplete data via the EM algorithm," Journal of the Royal Statistical Society: Series B (Methodological), vol.39(1), pp. 1–22, 1977.

[13] Emanuel Todorov, Tom Erez, and Yuval Tassa. "Mujoco: A physics engine for model-based control." In International Conference on Intelligent Robots and Systems, pp. 5026-5033, 2012.

[14] Kingma D , Ba J . Adam: A Method for Stochastic Optimization[J]. Computer Science, 2014.