This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

An Off-Policy Trust Region Policy Optimization Method With Monotonic Improvement Guarantee for Deep Reinforcement Learning

Wenjia Meng, Qian Zheng, Yue Shi, and Gang Pan, *Member, IEEE*

***Abstract*—In deep reinforcement learning, off-policy data help reduce on-policy interaction with the environment, and the trust region policy optimization (TRPO) method is efficient to stabilize the policy optimization procedure. In this article, we propose an off-policy TRPO method, off-policy TRPO, which exploits both on- and off-policy data and guarantees the monotonic improvement of policies. A surrogate objective function is developed to use both on- and off-policy data and keep the monotonic improvement of policies. We then optimize this surrogate objective function by approximately solving a constrained optimization problem under arbitrary parameterization and finite samples. We conduct experiments on representative continuous control tasks from OpenAI Gym and MuJoCo. The results show that the proposed off-policy TRPO achieves better performance in the majority of continuous control tasks compared with other trust region policy-based methods using off-policy data.**

***Index Terms*—Deep reinforcement learning, off-policy data, policy-based method, trust region.**

I. Introduction

MODEL-FREE deep reinforcement learning has achieved great successes in scaling reinforcement learning to complex sequential decision-making problems [1]–[3]. The model-free deep reinforcement learning aims to optimize an agent's policy through trial and error interaction with a black-box environment [4], [5]. According to the way how to learn the policy, previous works can be roughly divided into two categories [6]–[8]: value-based methods [9]–[12], [13]–[15] and policy-based methods [16]–[19], [20], [21]. Generally, value-based methods are less effective in continuous control tasks due to the problem of the curse of dimensionality, while policy-based methods address this problem by directly learning policy distribution over action space [22].

Though policy-based methods are likely to be more effective on continuous control tasks [22]–[24], they suffer from instability due to their simple policy update strategy [5], [25], [26]. Trust region policy-based methods [18], [25]–[27], [28]–[30] are proposed to alleviate such instability through bounding policy updates to a trust region. However, early works (e.g., [18], [25]–[27], [28]) cannot fully exploit off-policy data [31], [32] and, thus, require a large amount of on-policy interaction with the environment, resulting in significant performance degradation for many real-world applications.

Recently, several trust region policy-based works are proposed to take off-policy data into account as the utilization of these data can reduce the on-policy interaction with the environment [33]. Previous works use off-policy data to train value function (e.g., Q-Prop [33]) or both value function and policy function (e.g., interpolated policy gradient (IPG) [34], actor critic with experience replay (ACER) [16], and Trust-PCL [5]). However, these methods cannot guarantee monotonic improvement, which is considered to be critical for the training of complex and powerful policies [25].

To this end, we develop an off-policy trust region policy optimization (TRPO) method, off-policy TRPO, which can guarantee the monotonic improvement of policies. Specifically, the proposed off-policy TRPO guarantees monotonic policy improvement by monotonically optimizing a surrogate objective function using both on- and off-policy data. This surrogate objective function in our method is inspired by the idea of the surrogate objective function using only on-policy data in TRPO [25]. Our contributions are twofold.

1) We develop a novel surrogate objective function using both on- and off-policy data. The monotonic improvement of this surrogate objective function can guarantee the monotonic improvement of policies.
2) We propose a practical optimization method (called off-policy TRPO) to optimize our surrogate objective function under parameterized policies and finite samples, which achieves superior performance compared with other trust region policy-based methods using off-policy data.

II. Notation and Background

In this article, we consider a standard reinforcement learning setup with the Markov decision process (MDP) [35]. The MDP comprises of a state space S, an action space A, a stationary transition dynamics distribution $P : S \times A \times S \to \mathbb{R}$, an initial state distribution $\rho_0 : S \to \mathbb{R}$, a reward function $r : S \times A \to \mathbb{R}$, and a discount factor $\gamma \in (0, 1)$. In MDP, an agent acts in

Manuscript received January 23, 2020; revised August 10, 2020; accepted November 24, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1002503, in part by the Natural Science Foundation of China under Grant 61925603 and Grant U1909202, and in part by the Ten Thousand Talent Program of Zhejiang Province under Grant 2018R52039. *(Corresponding author: Gang Pan.)*

Wenjia Meng and Yue Shi are with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: mengwenjia@zju.edu.cn; shiyue@zju.edu.cn).

Qian Zheng is with the ROSE Lab, Nanyang Technological University, Singapore 637553 (e-mail: csqianzheng@gmail.com).

Gang Pan is with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310027, China (e-mail: gpan@zju.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2020.3044196.

Digital Object Identifier 10.1109/TNNLS.2020.3044196

2162-237X © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

a stochastic environment E by sequentially choosing actions over a sequence of timesteps. At each timestep t, the agent encounters a state s_t and performs an action a_t according to policy $\pi : S \times A \rightarrow [0, 1]$. The environment then returns a scalar reward $r(s_t, a_t)$ and a new state according to dynamics $P(s_{t+1} \mid s_t, a_t)$.

According to policy π, the agent interacts with the MDP to give a trajectory of states, actions, and rewards: $s_0, a_0, r_0, \ldots, s_t, a_t, r_t, \ldots$ The return R_t is the cumulative discounted reward from timestep t, $R_t = \sum_{k=t}^{\infty} \gamma^{k-t} r(s_k, a_k)$ [36]. The state value function and the action value function are defined as the expected return, $V_\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, \ldots}[R_t]$ and $Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots}[R_t]$ [36]. The formulations of these two functions and their relationship [4] are

$$V_\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, \ldots \sim \pi}\left[\sum_{k=t}^{\infty} \gamma^{k-t} r(s_k, a_k)\right] \tag{1}$$

$$Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots \sim \pi}\left[\sum_{k=t}^{\infty} \gamma^{k-t} r(s_k, a_k)\right] \tag{2}$$

$$V_\pi(s_t) = \mathbb{E}_{a_t \sim \pi(\cdot|s_t)}[Q_\pi(s_t, a_t)]. \tag{3}$$

The standard definition for the advantage function then can be represented

$$A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t) \tag{4}$$

which provides a relative measure of value with each action since $\mathbb{E}_{a_t \sim \pi}[A_\pi(s_t, a_t)] = 0$. Given policy π, the discounted visitation frequency $\rho_\pi(s)$ [25] can be defined as a discounted weighting of states encountered starting at s_0 and then following π [37]:

$$\begin{aligned}\rho_\pi(s) &= P(s_0 = s) + \gamma P(s_1 = s) + \gamma^2 P(s_2 = s) + \cdots \\ &= \sum_{t=0}^{\infty} \gamma^t P(s_t = s | s_0, \pi).\end{aligned} \tag{5}$$

In the following, we first introduce the goal of reinforcement learning, i.e., maximizing policy performance. Next, we briefly introduce the TRPO method that can use on-policy data to improve policy performance with the monotonic improvement guarantee.

A. Maximizing Policy Performance

The goal of reinforcement learning is to maximize expected return from the start state, denoted by policy performance objective $\eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots}[R_0]$ [37], [38]. This objective is further represented as

$$\eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$$

where

$$s_0 \sim \rho_0, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t, a_t). \tag{6}$$

The expected return of another policy $\tilde{\pi}$ can be expressed in terms of the advantage over π, accumulated over timesteps (detailed in [39])

$$\begin{aligned}\eta(\tilde{\pi}) &= \eta(\pi) + \mathbb{E}_{s_0, a_0, \ldots \sim \tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right] \\ &= \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s, a).\end{aligned} \tag{7}$$

Note that any policy update $\pi \rightarrow \tilde{\pi}$ that satisfies $\sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s, a) \geq 0$ can guarantee the improvement of policy performance η. The difficulty of such a policy improvement guarantee lies in the dependence of $\rho_{\tilde{\pi}}(s)$ on $\tilde{\pi}$. In order to reduce such dependence, a local approximation to $\eta(\tilde{\pi})$ is proposed by replacing visitation frequency $\rho_{\tilde{\pi}}$ with ρ_π [25]

$$L_\pi(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s, a). \tag{8}$$

A sufficiently small step $\pi \rightarrow \tilde{\pi}$ that improves L_π will also improve η [25]. However, such an improvement of η is limited by this sufficiently small step.

B. Trust Region Policy Optimization

TRPO [25] is proposed to avoid such limitation, which is inspired by the method of conservative policy iteration [39]. TRPO proposes a surrogate objective function based on a lower bound for policy performance η and proves that maximizing the proposed function can guarantee the improvement of policy performance. Specifically, the lower bound provided in TRPO is as follows:

$$\eta(\pi_{\text{new}}) \geq L_{\pi_{\text{old}}}(\pi_{\text{new}}) - \frac{4\epsilon\gamma}{(1-\gamma)^2} D_{\text{KL}}^{\max}(\pi_{\text{old}}, \pi_{\text{new}}) \tag{9}$$

where $\epsilon = \max_{s,a} |A_{\pi_{\text{old}}}(s, a)|$, π_{old} and π_{new} separately represent current policy and new policy, and the distance formulation $D_{\text{KL}}^{\max}(\pi_{\text{old}}, \pi_{\text{new}})$ is defined by the Kullback–Leibler (KL) divergence [40]: $D_{\text{KL}}^{\max}(\pi_{\text{old}}, \pi_{\text{new}}) = \max_s D_{\text{KL}}(\pi_{\text{old}}(\cdot|s) \parallel \pi_{\text{new}}(\cdot|s))$. Equation (9) shows that the improvement of the surrogate objective function in the right-hand side can guarantee the improvement of policy performance η [25]. For this surrogate objective function, its visitation frequency $\rho_{\pi_{\text{old}}}$ in $L_{\pi_{\text{old}}}(\pi_{\text{new}})$ depends on specific behavior policy, i.e., current policy π_{old}.

III. Surrogate Objective Function

TRPO is difficult to use off-policy data because its surrogate objective function depends on a specific behavior policy, i.e., current policy. In order to maintain the advantage of the monotonic policy improvement guarantee and make full use of off-policy data, we develop a new surrogate objective function that depends on general behavior policy, which can represent both current policy and previous policy. Specifically, we first propose a local approximation to policy performance η and derive its corresponding lower bound of policy performance. Next, with this lower bound, we derive the desired surrogate objective function that can guarantee the monotonic improvement of policy performance.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A. Lower Bound of Policy Performance η

We propose a local approximation to $\eta(\tilde{\pi})$ by replacing visitation frequency $\rho_{\tilde{\pi}}$ by ρ_μ

$$L_{\pi,\mu}(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_\mu(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a) \tag{10}$$

where the general behavior policy μ can represent both current policy and previous policy, and $\rho_\mu(s)$ denotes a discounted weighting of states encountered starting at s_0 and then following μ: $\rho_\mu(s) = \sum_{t=0}^{\infty} \gamma^t P(s_t = s|s_0, \mu)$ [37]. This approximation is inspired by the approximation in (8). The key difference between approximation here and that of (8) is that our $L_{\pi,\mu}$ uses the visitation frequency ρ_μ rather than ρ_π, which implies that both on- and off-policy data can be used in this approximation.

With the proposed approximation, we next describe how to derive the lower bound of policy performance η. In the derivation, we adopt the total variation (TV) divergence [25] as policy distance measure, which is defined as $D_{\mathrm{TV}}(p\|q) = 1/2 \sum_i |p_i - q_i|$ for probability distribution p, q. The specific formulation used in the derived bound is

$$D_{\mathrm{TV}}^{\max}(p,q) = \max_s D_{\mathrm{TV}}(p(\cdot|s) \parallel q(\cdot|s)) \tag{11}$$

which is consistent with that in [25]. The original formulation of the lower bound is described in the following theorem.

Theorem 1: Let $\alpha_\pi = D_{\mathrm{TV}}^{\max}(\pi_{\mathrm{old}}, \pi_{\mathrm{new}})$, $\alpha_\mu = D_{\mathrm{TV}}^{\max}(\mu, \pi_{\mathrm{new}})$. Then, the following bound holds:

$$\eta(\pi_{\mathrm{new}}) \geq L_{\pi_{\mathrm{old}},\mu}(\pi_{\mathrm{new}}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha_\pi \alpha_\mu$$

where

$$\epsilon = \max_{s,a} |A_{\pi_{\mathrm{old}}}(s,a)|. \tag{12}$$

The proof can be found in Appendix A.

Based on the upper bound of α_μ in Appendix B, i.e., $\alpha_\mu \leq D_{\mathrm{TV}}^{\max}(\mu, \pi_{\mathrm{old}}) + D_{\mathrm{TV}}^{\max}(\pi_{\mathrm{old}}, \pi_{\mathrm{new}})$, (12) can be rewritten as

$$\begin{aligned}\eta(\pi_{\mathrm{new}}) &\geq L_{\pi_{\mathrm{old}},\mu}(\pi_{\mathrm{new}}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha_\pi \alpha_\mu \\ &\geq L_{\pi_{\mathrm{old}},\mu}(\pi_{\mathrm{new}}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha_\pi \\ &\quad \times \left(D_{\mathrm{TV}}^{\max}(\mu, \pi_{\mathrm{old}}) + D_{\mathrm{TV}}^{\max}(\pi_{\mathrm{old}}, \pi_{\mathrm{new}})\right)\end{aligned}$$

where

$$\epsilon = \max_{s,a} |A_{\pi_{\mathrm{old}}}(s,a)|. \tag{13}$$

The lower bound of $\eta(\pi_{\mathrm{new}})$ then follows:

$$\begin{aligned}&\eta(\pi_{\mathrm{new}}) \\ &\geq L_{\pi_{\mathrm{old}},\mu}(\pi_{\mathrm{new}}) - CD_{\mathrm{TV}}^{\max}(\pi_{\mathrm{old}}, \pi_{\mathrm{new}}) \\ &\quad \left[D_{\mathrm{TV}}^{\max}(\mu, \pi_{\mathrm{old}}) + D_{\mathrm{TV}}^{\max}(\pi_{\mathrm{old}}, \pi_{\mathrm{new}})\right] \\ &= L_{\pi_{\mathrm{old}},\mu}(\pi_{\mathrm{new}}) - C[\max_s D_{\mathrm{TV}}(\mu, \pi_{\mathrm{old}}) \max_s D_{\mathrm{TV}}(\pi_{\mathrm{old}}, \pi_{\mathrm{new}}) \\ &\qquad + \max_s D_{\mathrm{TV}}(\pi_{\mathrm{old}}(\cdot|s), \pi_{\mathrm{new}}(\cdot|s))^2]\end{aligned}$$

where

$$C = \frac{4\epsilon\gamma}{(1-\gamma)^2}. \tag{14}$$

B. Derivation of Surrogate Objective Function

The derivation of the surrogate objective function is based on the lower bound in (14). Specifically, we take advantage of the relationship between TV divergence and KL divergence $D_{\mathrm{TV}}(p \parallel q)^2 \leq D_{\mathrm{KL}}(p \parallel q)$ [41] and obtain the following lower bound from (14):

$$\eta(\tilde{\pi}) \geq L_{\pi,\mu}(\tilde{\pi}) - CD_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\mu,\pi) D_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\pi,\tilde{\pi}) - CD_{\mathrm{KL}}^{\max}(\pi,\tilde{\pi}) \tag{15}$$

where $D_{\mathrm{KL}}^{\max}(\pi,\tilde{\pi}) = \max_s D_{\mathrm{KL}}(\pi(\cdot|s) \parallel \tilde{\pi}(\cdot|s))$, $D_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\pi,\tilde{\pi}) = \max_s (D_{\mathrm{KL}}(\pi(\cdot|s) \parallel \tilde{\pi}(\cdot|s)))^{1/2}$, and $D_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\mu,\pi) = \max_s (D_{\mathrm{KL}}(\mu(\cdot|s) \parallel \pi(\cdot|s)))^{1/2}$.

It should be noted that a policy update that improves the right-hand side of (15) can guarantee the improvement of the true performance η. For simplicity, we denote the right-hand side of (15) as $M_\pi(\tilde{\pi})$

$$M_\pi(\tilde{\pi}) = L_{\pi,\mu}(\tilde{\pi}) - CD_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\mu,\pi) D_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\pi,\tilde{\pi}) - CD_{\mathrm{KL}}^{\max}(\pi,\tilde{\pi}). \tag{16}$$

Note that, when $\tilde{\pi} = \pi$, $M_\pi(\tilde{\pi})$ in (16) can be represented as

$$\begin{aligned}&M_\pi(\pi) \\ &= L_{\pi,\mu}(\pi) - CD_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\mu,\pi) D_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\pi,\pi) - CD_{\mathrm{KL}}^{\max}(\pi,\pi) \\ &= L_{\pi,\mu}(\pi) \quad \text{by } D_{\mathrm{KL}}(\pi(\cdot|s) \parallel \pi(\cdot|s)) = 0 \\ &= \eta(\pi) + \sum_s \rho_\mu(s) \sum_a \pi(a|s) A_\pi(s,a) \quad \text{by Equation (10)} \\ &= \eta(\pi) \\ &\quad + \sum_s \rho_\mu(s) \sum_a \pi(a|s)(Q_\pi(s,a) - V_\pi(s)) \text{by Equation (4)} \\ &= \eta(\pi) + \sum_s \rho_\mu(s)(V_\pi(s) - V_\pi(s)) \quad \text{by Equation (3)} \\ &= \eta(\pi).\end{aligned} \tag{17}$$

According to (15) and (17), the lower bound of $\eta(\tilde{\pi}) - \eta(\pi)$ is derived as follows:

$$\begin{aligned}\eta(\tilde{\pi}) &\geq M_\pi(\tilde{\pi}) \text{ by Equation (15)} \\ \eta(\pi) &= M_\pi(\pi) \text{ by Equation (17).}\end{aligned}$$

Thus

$$\eta(\tilde{\pi}) - \eta(\pi) \geq M_\pi(\tilde{\pi}) - M_\pi(\pi). \tag{18}$$

That is, maximizing the function $M_\pi(\tilde{\pi})$ by choosing appropriate policy $\tilde{\pi}$ can guarantee that the true objective η is nondecreasing. Therefore, $M_\pi(\tilde{\pi})$ is our desired surrogate objective function that can make use of both on- and off-policy data and guarantee the monotonic improvement of policies.

In order to further illustrate the monotonic improvement guarantee, we then describe a policy iteration scheme that optimizes the proposed surrogate objective function at each iteration. Specifically, we construct the surrogate objective function $M_{\pi_i}(\pi)$ and maximize this function at each iteration i, i.e.,

$$\pi_{i+1} = \arg\max_\pi \left[L_{\pi_i,\mu}(\pi) - CD_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\mu,\pi_i) D_{\mathrm{KL}}^{\max,\mathrm{sqrt}}(\pi_i,\pi) - CD_{\mathrm{KL}}^{\max}(\pi_i,\pi)\right]. \tag{19}$$

This policy iteration scheme can generate a monotonically improved policy sequences $\eta(\pi_k) \leq \eta(\pi_{k+1})$, $k \in \mathbb{N}$.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

IV. Off-Policy TRPO

The optimization of the proposed surrogate objective function in Section III is independent of the policy parameterization. It assumes that the policy can be evaluated in all states. However, in practice, it depends on the policy parameterization and only exploits finite samples [25], [28]. To tackle this challenge, we approximate the above optimization problem to a constrained optimization problem under parameterized policies and finite samples.

A. Derivation of the Constrained Optimization Problem

The derivation of the constrained optimization problem is based on parameterized policies and finite samples, which is widely adopted in practical methods [25], [28]. Different from that in [25] and [28], our derivation is to optimize the surrogate objective function proposed in Section III.

1) Optimization Problem Under Parameterized Policies: We consider parameterized policies by representing policies with $\pi_\theta(a|s)$, where θ denotes the parameter vector. With the parameterized policies, the lower bound of policy performance η in (15) becomes

$$\eta(\theta) \geq L_{\theta_{\text{old}},\mu}(\theta) - \mathrm{C}D_{\text{KL}}^{\text{max, sqrt}}(\mu,\theta_{\text{old}})D_{\text{KL}}^{\text{max, sqrt}}(\theta_{\text{old}},\theta) - \mathrm{C}D_{\text{KL}}^{\text{max}}(\theta_{\text{old}},\theta) \quad (20)$$

where $\eta(\theta) := \eta(\pi_\theta)$, $L_{\theta_{\text{old}},\mu}(\theta) := L_{\pi_{\theta_{\text{old}}},\mu}(\pi_\theta)$, $D_{\text{KL}}^{\text{max}}(\theta_{\text{old}},\theta) := D_{\text{KL}}^{\text{max}}(\pi_{\theta_{\text{old}}},\pi_\theta)$, $D_{\text{KL}}^{\text{max, sqrt}}(\theta_{\text{old}},\theta) := D_{\text{KL}}^{\text{max, sqrt}}(\pi_{\theta_{\text{old}}},\pi_\theta)$, $D_{\text{KL}}^{\text{max, sqrt}}(\mu,\theta_{\text{old}}) := D_{\text{KL}}^{\text{max, sqrt}}(\mu,\pi_{\theta_{\text{old}}})$, and θ_{old} denotes current policy parameters. Thus, the optimization problem guaranteeing the improvement of the true objective η can be represented as

$$\underset{\theta}{\text{maximize}}\big[L_{\theta_{\text{old}},\mu}(\theta) - \mathrm{C}D_{\text{KL}}^{\text{max, sqrt}}(\mu,\theta_{\text{old}})D_{\text{KL}}^{\text{max, sqrt}}(\theta_{\text{old}},\theta) - \mathrm{C}D_{\text{KL}}^{\text{max}}(\theta_{\text{old}},\theta)\big]. \quad (21)$$

However, the above optimization process is limited by a small step size due to the penalty coefficient C. Similar to the solution in [25] and [28], we adopt a trust region constraint to take larger steps

$$\begin{aligned}&\max_\theta\ L_{\theta_{\text{old}},\mu}(\theta)\\ &\text{s.t. } D_{\text{KL}}^{\text{max, sqrt}}(\mu,\theta_{\text{old}})D_{\text{KL}}^{\text{max, sqrt}}(\theta_{\text{old}},\theta) + D_{\text{KL}}^{\text{max}}(\theta_{\text{old}},\theta) \leq \delta \end{aligned} \quad (22)$$

where δ indicates the bound for the trust region constraint.

The constraints bounded at each point in the state space make the above problem intractable [25]. We employ the constraint of the average KL divergence: $\overline{D}_{\text{KL}}^{\rho_\mu}(\theta_{\text{old}},\theta)$, $\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\mu,\theta_{\text{old}})$, and $\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\theta_{\text{old}},\theta)$, which are defined in Appendix C. Therefore, the optimization problem with trust region constraint in (22) can be approximated by

$$\begin{aligned}&\max_\theta\ L_{\theta_{\text{old}},\mu}(\theta)\\ &\text{s.t. } \overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\mu,\theta_{\text{old}})\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\theta_{\text{old}},\theta) + \overline{D}_{\text{KL}}^{\rho_\mu}(\theta_{\text{old}},\theta) \leq \delta. \end{aligned} \quad (23)$$

2) Optimization Problem under Finite Samples: Based on the derived optimization problem under parameterized policies in (23), we next take finite samples into account. In order to use finite samples to estimate the objective in the constrained optimization problem, we expand $L_{\theta_{\text{old}},\mu}(\theta)$ in (23)

$$L_{\theta_{\text{old}},\mu}(\theta) = \eta(\theta_{\text{old}}) + \sum_s \rho_\mu(s)\sum_a \pi_\theta(a|s)A_{\theta_{\text{old}}}(s,a). \quad (24)$$

$\sum_s \rho_\mu(s)[\cdots]$ in Equation (24) can be replaced by the expectation $(1/1-\gamma)\mathbb{E}_{s\sim\rho_\mu}[\cdots]$ [25]. Subsequently, the summation of actions (sum over actions) can be replaced by importance sampling estimator, and the contribution of a single s to the objective function becomes

$$\sum_a \pi_\theta(a|s)A_{\theta_{\text{old}}}(s,a) = \mathbb{E}_{a\sim\mu}\left[\frac{\pi_\theta(a|s)}{\mu(a|s)}A_{\theta_{\text{old}}}(s,a)\right]. \quad (25)$$

Thus, our optimization problem with trust region constraint in (23) can be represented in terms of expectations

$$\begin{aligned}&\max_\theta\ \mathbb{E}_{s\sim\rho_\mu,a\sim\mu}\left[\frac{\pi_\theta(a|s)}{\mu(a|s)}A_{\theta_{\text{old}}}(s,a)\right]\\ &\text{s.t. } \overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\mu,\theta_{\text{old}})\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\theta_{\text{old}},\theta) + \overline{D}_{\text{KL}}^{\rho_\mu}(\theta_{\text{old}},\theta) \leq \delta. \end{aligned} \quad (26)$$

Specifically, the expectations in (26) can be approximated by averaging finite samples, which can be either on-policy or off-policy.

B. Solving the Constrained Optimization Problem

The solution to the constrained optimization problem under parameterized policies and finite samples in (26) contains two steps that are similar to those in [25]. Specifically, we first compute a search direction s for the policy update. Then, we derive a reasonable step size κ, which enables policy updates to improve objective function and satisfies the trust region constraint. With the search direction s and step size κ, we update policy parameters by $\theta = \theta_{\text{old}} + \kappa s$ to optimize the constrained optimization problem in (26).

Adopting a linear approximation to the objective and a quadratic approximation to the constraint [25], the constrained optimization problem in (26) becomes

$$\begin{aligned}&\max_\theta[\nabla_\theta O_{\theta_{\text{old}},\mu}(\theta)\mid_{\theta=\theta_{\text{old}}}\cdot(\theta-\theta_{\text{old}})]\\ &\text{s.t. } \frac{1}{2}(\theta-\theta_{\text{old}})^T F(\theta_{\text{old}})(\theta-\theta_{\text{old}}) \leq \delta \end{aligned} \quad (27)$$

where

$$F(\theta_{\text{old}})_{ij} = \frac{\partial}{\partial\theta_i}\frac{\partial}{\partial\theta_j}\big[\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\mu,\theta_{\text{old}})\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\theta_{\text{old}},\theta) + \overline{D}_{\text{KL}}^{\rho_\mu}(\theta_{\text{old}},\theta)\big]|_{\theta=\theta_{\text{old}}}$$

$$O_{\theta_{\text{old}},\mu}(\theta) = \mathbb{E}_{s\sim\rho_\mu,a\sim\mu}\left[\frac{\pi_\theta(a|s)}{\mu(a|s)}A_{\theta_{\text{old}}}(s,a)\right].$$

The search direction can be obtained by approximately solving the equation $F(\theta_{\text{old}})s = \nabla_\theta O_{\theta_{\text{old}},\mu}(\theta)\mid_{\theta=\theta_{\text{old}}}$, where s represents the search direction. It is difficult to obtain this search direction due to its prohibitive cost on forming full matrix $F(\theta_{\text{old}})^{-1}$. In order to address this issue, we adopt

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Algorithm 1 Off-Policy TRPO

```
Require: Environment $E$, trust region bound $\delta$, discount factor
  $\gamma$, trace-decay parameter $\lambda$, collected steps $P$, replay times
  $B$, replay buffer $RB$
  Initialize policy network parameter $\theta$
  Initialize state value network parameter $w$
  Initialize empty replay buffer $RB$
  repeat
    // Collect
    Sample $P$ steps $s_{0:P} \sim \pi_\theta$ on environment $E$
    Add $\mathcal{S}_{0:P} = \{s_{0:P}, a_{0:P-1}, r_{0:P-1}, \{\mu(\cdot|s_j)|_{\mu=\pi_\theta}\}_{j=0}^{P-1}\}$ to
  replay buffer $RB$
    // Update using on-policy data
    Exploit on-policy data $\mathcal{S}_{0:P}$ collected in previous step
    Call UPDATE(data $\mathcal{S}_{0:P}$) in Algorithm 2
    // Update using off-policy data in RB
    for $k = 1$ to $B$ do
      Sample off-policy data $\mathcal{S}_{0:P}$ from $RB$
      Call UPDATE(data $\mathcal{S}_{0:P}$) in Algorithm 2
    end for
  until $\pi_\theta$ converges
```

the conjugate gradient algorithm [25], [42] to approximately obtain search direction $s \approx F(\theta_{\text{old}})^{-1}\nabla_\theta O_{\theta_{\text{old}},\mu}(\theta) \ |_{\theta=\theta_{\text{old}}}$ without forming full matrix $F(\theta_{\text{old}})^{-1}$.

Given search direction s, according to the trust region constraint in (27), the maximal step length β satisfies

$$\delta \approx \frac{1}{2}(\theta_{\text{old}}-\theta)^T F(\theta_{\text{old}})(\theta_{\text{old}}-\theta) = \frac{1}{2}(\beta s)^T F(\theta_{\text{old}})(\beta s). \quad (28)$$

Next, we derive the maximal step length according to (28): $\beta = (2\delta/(s^T F(\theta_{\text{old}})s))^{1/2}|_{\theta=\theta_{\text{old}}}$. We then obtain the reasonable step size κ by performing the line search that starts with the maximal step length β and shrinks exponentially until the objective improves.

Based on the derived search direction $s = F(\theta_{\text{old}})^{-1}\nabla_\theta O_{\theta_{\text{old}},\mu}(\theta) \ |_{\theta=\theta_{\text{old}}}$ and step size κ, we can obtain the policy update scheme

$$\theta = \theta_{\text{old}} + \kappa F(\theta_{\text{old}})^{-1}\nabla_\theta O_{\theta_{\text{old}},\mu}(\theta) \ |_{\theta=\theta_{\text{old}}} \quad (29)$$

which can improve the objective function with trust region constraint in (26) and guarantee monotonic improvement of policies.

C. Off-Policy TRPO Algorithm

In this section, we will develop a practical method (off-policy TRPO) based on the above policy update scheme, which monotonically improves policy performance. The whole algorithm is outlined in Algorithm 1. The update function is summarized in Algorithm 2.

As described in Algorithm 1, the iteration for data collecting and policy update repeats until policy function converges. Specifically, at each iteration, we first collect data by interacting with the black-box environment and insert these collected data to replay buffer that is a commonly used technique for experience replay [1], [16]. We then use the on-policy data

Algorithm 2 Update Function

```
function UPDATE(data $\mathcal{S}_{0:P}$)
    Estimate $\{V(s_j)\}_{j=0}^{P}$ using state value network
    Estimate $\{Q(s_j, a_j)\}_{j=0}^{P-1}$ by $RQ_j = r_j +$
  $\gamma\lambda\overline{\rho}_{j+1}[RQ_{j+1} - V(s_{j+1})] + \gamma V(s_{j+1})$
    Obtain advantage values $\{A(s_j, a_j)\}_{j=0}^{P-1} = \{Q(s_j, a_j) -$
  $V(s_j)\}_{j=0}^{P-1}$
    Construct the optimization problem with trust region
  constraint in Equation (26) in Section IV-A
    Update policy parameter $\theta$ according to Equation (29) in
  Section IV-B
    Update parameter $w$ of state value network $V$ by mini-
  mizing a mean squared error loss $\frac{1}{P}\sum_{j=0}^{P-1}(RQ_j - V(s_j))^2$
end function
```

collected in the previous step to perform the update function in Algorithm 2. Finally, we sample data from the replay buffer and take advantage of these sampled off-policy data to perform the update function in Algorithm 2.

As described in Algorithm 2, we make use of the data inputted to the function to update the policy network and the state value network. Specifically, we first use these data to estimate state value $V(s_j)$, where j indicates the tag number of a sample among these data. We then estimate state action value $Q(s_j, a_j)$ using an approximated variant of retrace estimator [16], [43]: $RQ_j = r_j + \gamma\lambda\overline{\rho}_{j+1}[RQ_{j+1} - V(s_{j+1})] + \gamma V(s_{j+1})$, where $\overline{\rho}_{j+1} = \min\{1, \rho_{j+1}\}$ with $\rho_{j+1} = (\pi(a_{j+1}|s_{j+1})/\mu(a_{j+1}|s_{j+1}))$ and λ represents the trace-decay parameter. Different from the original retrace estimator [16], [43], we use $V(s_{j+1})$ to replace $Q(s_{j+1}, a_{j+1})$ to eliminate the need for the action value network and simplify our method. With the estimated state value $V(s_j)$ and state action value $Q(s_j, a_j)$ described above, we can obtain the advantage value $A(s_j, a_j) = Q(s_j, a_j) - V(s_j)$. We next construct the constrained optimization problem with trust region in (26) and update policy parameter by (29). Subsequently, we adopt the approximated retrace estimator as a target for learning the state value network and update it by minimizing the mean squared error loss.

V. Experiments

To validate the proposed off-policy TRPO, we conduct experiments using nine representative continuous control tasks. We first describe the experimental setup that includes the detailed configuration in our method and the continuous control tasks used in the experiments. We then analyze an important hyperparameter, i.e., trust region bound, for the proposed off-policy TRPO. We next compare the overall performance of our method with other state-of-the-art methods. Finally, we study its effectiveness in terms of monotonic improvement guarantee, using off-policy data, and computational efficiency.

A. Setup

Neural network architectures for both policy and state value functions are the fully connected neural network with two

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

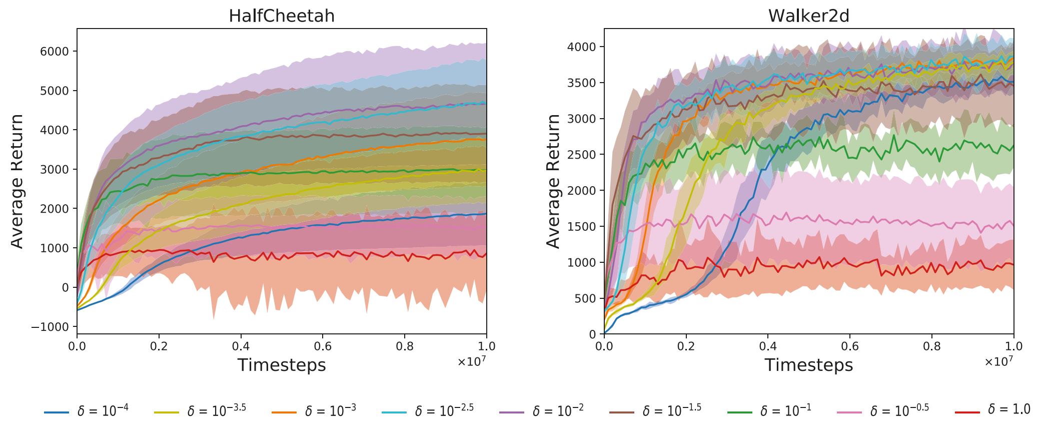


Fig. 1. Results of the proposed off-policy TRPO regarding different δ values on HalfCheetah and Walker2d tasks. The shaded region indicates the standard deviation over ten random seeds. The X-axis represents the timesteps in the environment. The Y-axis represents the average return.

hidden layers of dimension 64 and with tanh activations [5]. The output of the policy network specifies a Gaussian distribution to estimate policy [25]. The state value network produces a single scalar value to estimate state value [5]. For experimental parameters, we use the discount factor $\gamma = 0.995$ [5] and set trace-decay parameter $\lambda = 0.97$ [33]. The number of collected steps P is 5000 [33], and the value of replay times B is 4 [16]. The size of the replay buffer is 50 000 [16]. Note that the Adam optimizer [45] with a learning rate of 0.001 is used for the minimization of the loss for the state value network. The average return shown in the experimental results is obtained by averaging over ten seeded training runs [46]. Note that the returns for our method, TRPO [25], Q-Prop [33], IPG [34], ACER [16], and Trust-PCL [5] shown in the experiments are based on random sampling actions (rather than greedy actions [5]) due to that optimal policy are often stochastic, selecting different actions with specific probabilities [37]. Notice that the experiments are performed on a server using four Titan X GPU, 48 CPU cores, and 128 GB of memory.

Environments for validation consist of nine representative continuous control tasks, i.e., two OpenAI Gym tasks [47] (MountainCarContinuous and Pendulum) and seven MuJoCo tasks [48] (InvertedPendulum, Swimmer, Hopper, HalfCheetah, Walker2d, Ant, and Humanoid). The complexity of these tasks lies in the dimensions of their state space and action space. Such dimensions for these nine tasks are summarized in Table I. Table I indicates that the tasks, i.e., Swimmer, Hopper, HalfCheetah, Walker2d, Ant, and Humanoid, are more complex than the other tasks, i.e., MountainCarContinuous, Pendulum, and InvertedPendulum.

TABLE I
LIST OF ENVIRONMENTS. $||S||$ AND $||A||$ SEPARATELY REPRESENT THE DIMENSIONS OF STATE SPACE AND ACTION SPACE

Environment	$\|\|S\|\|$	$\|\|A\|\|$
MountainCarContinuous	2	1
Pendulum	3	1
InvertedPendulum	4	1
Swimmer	8	2
Hopper	11	3
HalfCheetah	17	6
Walker2d	17	6
Ant	111	8
Humanoid	376	17

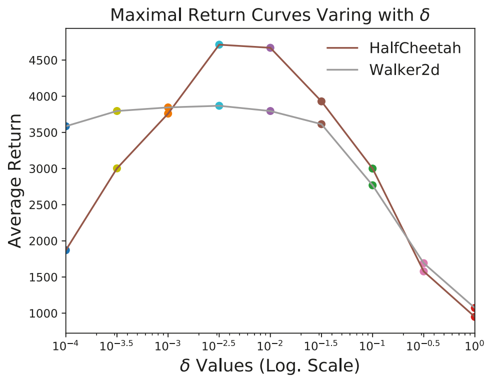


Fig. 2. Maximal return curves of the proposed off-policy TRPO on HalfCheetah and Walker2d tasks. Here, we focus on the proposed off-policy TRPO with $\delta = 10^{-4}, 10^{-3.5}, 10^{-3}, 10^{-2.5}, 10^{-2}, 10^{-1.5}, 10^{-1}, 10^{-0.5}$, and 1.0, and the above curves are based on these methods' maximal return values during training. The X-axis represents the δ values (logarithmic scale). The Y-axis represents the average return.

B. Results on Different Trust Region Bounds

In our method, the trust region bound, i.e., δ, is an important hyperparameter since it is a tradeoff between the utilization of off-policy data and the stability of our method. In order to validate its effect and choose a reasonable δ value, we compare our off-policy TRPO methods with several different δ values (i.e., 10^{-4}, $10^{-3.5}, 10^{-3}, 10^{-2.5}, 10^{-2}, 10^{-1.5}, 10^{-1}, 10^{-0.5}$, and 1.0). Among these proportional δ values, $10^{-3}, 10^{-2}, 10^{-1}$, and 1.0 are chosen due to that these four values are commonly used in representative trust region methods [16], [25], [33], [34], [5] and δ values (i.e., $10^{-4}, 10^{-3.5}, 10^{-2.5}, 10^{-1.5}$, and $10^{-0.5}$) are chosen for a comprehensive comparison. We show the performance comparison of our methods with different δ values during training in Fig. 1. We also compare their maximal returns during training in Fig. 2 to clearly show how their maximal returns vary with δ values.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

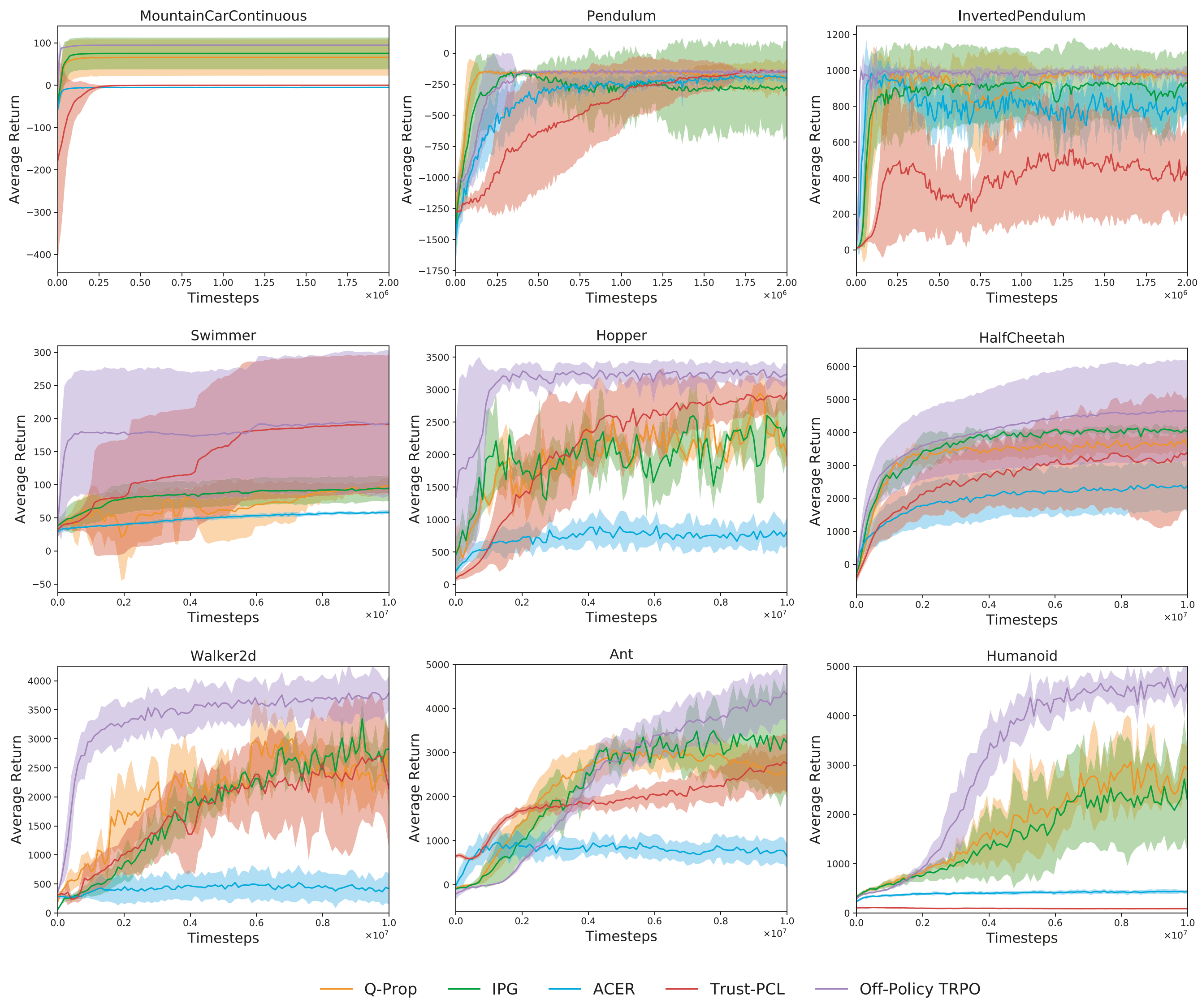


Fig. 3. Results of off-policy TRPO compared with other off-policy trust region methods, i.e., Q-Prop, IPG, ACER, and Trust-PCL, on nine representative continuous control tasks. The shaded region indicates the standard deviation over ten random seeds. Note that the return here is based on random sampling actions. The X-axis represents the timesteps in the environment. The Y-axis represents the average return.

We conduct experiments on two representative tasks, i.e., HalfCheetah and Walker2d, for their popularity [33], [34], [46] and complexity [48]. As shown in Figs. 1 and 2, our methods with small δ values (i.e., $10^{-4}, 10^{-3.5}, 10^{-3}$, and $10^{-2.5}$) achieve the same returns with more timesteps and achieve smaller or comparable returns compared with the method with $\delta = 10^{-2}$. Note that Fig. 2 shows that the return with small δ value nonlinearly increases and then stabilizes as δ value increases from 10^{-4} to 10^{-2}. The methods with small δ values are not as good as the method with $\delta = 10^{-2}$ due to that the small trust region bound limits the utilization of off-policy data. As shown in Figs. 1 and 2, our methods with large δ values (i.e., $10^{-1.5}, 10^{-1}, 10^{-0.5}$, and 1.0) achieve the same returns with more timesteps and achieve smaller returns compared with the method with $\delta = 10^{-2}$. Fig. 2 shows that the return with large δ value nonlinearly decreases as δ value increases from 10^{-2} to 1.0. The methods with large δ values are worse than the method with $\delta = 10^{-2}$ due to that the large trust region bound causes the instability of optimization. Results in Figs. 1 and 2 validate the effect of the trust region bound as a tradeoff and show that δ value 10^{-2} is a reasonable choice for trust region bound.

C. Comparison With the State-of-the-Art Methods

We compare our performance with four state-of-the-art trust region methods (Q-Prop [33], IPG[1] [34], ACER[2] [16], and Trust-PCL[3] [5]) that exploit both on- and off-policy data.

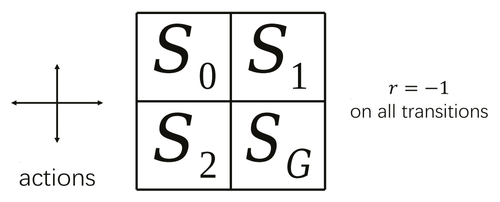
[1]The implementations for TRPO, Q-Prop, and IPG used in this article can be available at https://github.com/shaneshixiang/rllabplusplus, which is provided from [33] and [34].

[2]The implementation for ACER with continuous action space used in this article can be available at https://github.com/chainer/chainerrl, whose hyperparameter setting is consistent with that in [16].

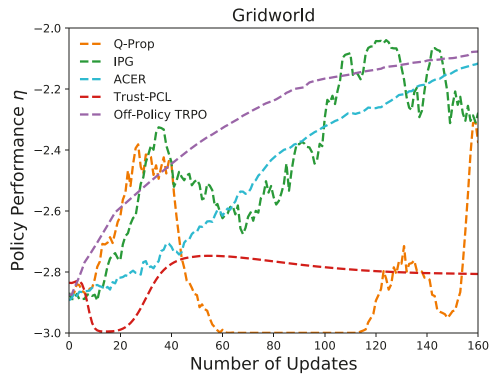
[3]The implementation for Trust-PCL used in this article can be available at https://github.com/tensorflow/models/tree/master/research/pcl_rl, which is provided from [5].

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



(a)



(b)

Fig. 4. (a) Illustration of gridworld environment. This 2×2 gridworld environment consists of a state space $S = \{S_0, S_1, S_2, S_G\}$ with a fixed starting state S_0 and a terminal state (a goal state) S_G, an action space $A = \{\text{up}, \text{down}, \text{right}, \text{left}\}$, which deterministically causes the corresponding state transitions, except that actions that would take the agent off the grid in fact leave the state unchanged [4]. The reward function in this environment is $r(s, a, s') = -1$ for all states s, s' and actions a. In order to further simplify this environment, we limit episodes terminate after three time steps or if the agent has reached the goal state [44]. (b) Comparison of policy performance η curves among the proposed off-policy TRPO and other off-policy trust region methods (Q-Prop, IPG, ACER, and Trust-PCL) on this gridworld task. The X-axis represents the number of update where data are collected using ten random seeds. The Y-axis represents the value of policy performance η, which is calculated according to the definition of η in (6): $\eta(\pi) = \mathbb{E}_{s_0,a_0,\ldots}[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t)] = \sum_i P(\{s_0^i, a_0^i, r_0^i, s_1^i, a_1^i, r_1^i, \ldots, s_{T-1}^i, a_{T-1}^i, r_{T-1}^i, s_T^i\}) \sum_{t=0}^{T-1} \gamma^t r(s_t^i, a_t^i)$, where T denotes the total time steps of an episode, and $\{s_0^i, a_0^i, r_0^i, s_1^i, a_1^i, r_1^i, \ldots, s_{T-1}^i, a_{T-1}^i, r_{T-1}^i, s_T^i\}$ represents the ith trajectory.

TABLE II

RESULTS FOR THE MAXIMAL AVERAGE RETURNS AND STANDARD DEVIATIONS DURING THE TRAINING PROCESS. FOR EACH TASK, THE NUMBER AFTER THE AVERAGE RETURN AND STANDARD DEVIATION REPRESENTS THE NUMBER OF RESULTS THAT PERFORM SIGNIFICANTLY WORSE THAN THE CURRENT RESULT ACCORDING TO WELCH'S T-TEST [49] WITH p-VALUE <0.05. BOLDED ARE SIGNIFICANTLY WORSE THAN THE RESULT OF OUR METHOD ACCORDING TO WELCH'S T-TEST WITH p-VALUE < 0.05. NOTE THAT WELCH'S T-TEST IS A COMMONLY USED TEST OF SIGNIFICANCE [50]–[52]

Return	TRPO		Q-Prop		IPG		ACER		Trust-PCL		Off-Policy TRPO (Ours)	
MountainCarContinuous	**93.9 ± 0.2**	(3)	**66.0 ± 42.6**	(2)	75.2 ± 37.6	(2)	**−5.2 ± 0.1**	(0)	**0.0 ± 0.0**	(1)	94.7 ± 0.1	(4)
Pendulum	−196.7 ± 180.4	(0)	−136.8 ± 19.4	(1)	−153.7 ± 28.6	(0)	**−178.6 ± 39.6**	(0)	−132.6 ± 16.5	(2)	−136.3 ± 14.7	(1)
InvertedPendulum	994.6 ± 12.8	(1)	996.4 ± 7.6	(1)	935.9 ± 192.3	(1)	982.9 ± 32.9	(1)	**560.6 ± 340.3**	(0)	1000.0 ± 0.0	(1)
Swimmer	**127.8 ± 10.3**	(3)	**98.7 ± 12.0**	(1)	**94.8 ± 19.7**	(1)	**58.5 ± 2.8**	(0)	191.9 ± 104.4	(4)	195.7 ± 106.3	(4)
Hopper	**2540.5 ± 531.9**	(1)	2941.8 ± 345.3	(1)	**2599.3 ± 326.7**	(1)	**902.8 ± 225.2**	(0)	**2963.0 ± 241.1**	(2)	3308.7 ± 127.0	(4)
HalfCheetah	**2380.4 ± 943.6**	(0)	3802.5 ± 330.4	(2)	4129.0 ± 107.3	(2)	**2416.1 ± 782.2**	(0)	3401.0 ± 1736.5	(0)	4670.3 ± 1547.6	(2)
Walker2d	**1845.5 ± 534.0**	(1)	3001.8 ± 552.6	(2)	3347.8 ± 339.4	(2)	**523.1 ± 286.8**	(0)	**2706.3 ± 974.2**	(2)	3795.8 ± 455.5	(3)
Ant	**2658.5 ± 404.4**	(1)	**3060.6 ± 383.1**	(2)	3502.4 ± 820.2	(1)	**963.8 ± 289.3**	(0)	**2793.6 ± 642.3**	(1)	4390.5 ± 630.9	(4)
Humanoid	**676.7 ± 76.2**	(2)	**3066.9 ± 867.7**	(3)	**2716.9 ± 1212.7**	(3)	**445.3 ± 45.3**	(1)	**113.9 ± 12.4**	(0)	4775.1 ± 307.8	(5)
Total Number		12		15		13		2		12		28

The comparison of the entire learning curves during training is shown in Fig. 3. From Fig. 3, we can observe that the proposed off-policy TRPO achieves the same return with fewer timesteps compared with other methods on most tasks, which is mainly due to that monotonic improvement guarantee stabilizes the policy update to accelerate policy learning of our method. It can also be observed that the proposed off-policy TRPO achieves a higher return than other off-policy trust region methods during the training timesteps on most tasks, which principally benefits from stable policy learning provided by monotonic improvement guarantee. As shown in Fig. 3, the gap between our method and other compared methods on the simpler tasks, i.e., MountainCarContinuous, Pendulum, and InvertedPendulum, is slight due to the simplicity of these tasks; this gap on the more complex tasks, i.e., Swimmer, Hopper, HalfCheetah, Walker2d, Ant, and Humanoid, is noticeable due to that monotonic improvement guarantee stabilizes our method, making it more robust on complex tasks than compared methods.

The comparison of the maximal average returns attained during the training process is summarized in Table II. As shown in Table II, the proposed off-policy TRPO achieves the highest maximal average return during the training process among these compared methods on most tasks, i.e., Swimmer, Hopper, HalfCheetah, Walker2d, Ant, and Humanoid. On these tasks, it is also noticeable that the returns achieved by our method are significantly better than the ones achieved by other most methods. In Table II, the proposed off-policy TRPO achieves higher or comparable returns compared with other methods on MountainCarContinuous, Pendulum, and Inverted-Pendulum. Note that the returns of our method are significantly better than the ones of a part of methods on these three tasks, which is due to the simplicity of these tasks. In Table II, notice that the number after the average return and standard deviation represents the number of results that are significantly worse than the current result, and the largest total number achieved by our method demonstrates our superior performance. These experimental results from Fig. 3 and Table II validate that the proposed off-policy TRPO can make use of samples more

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

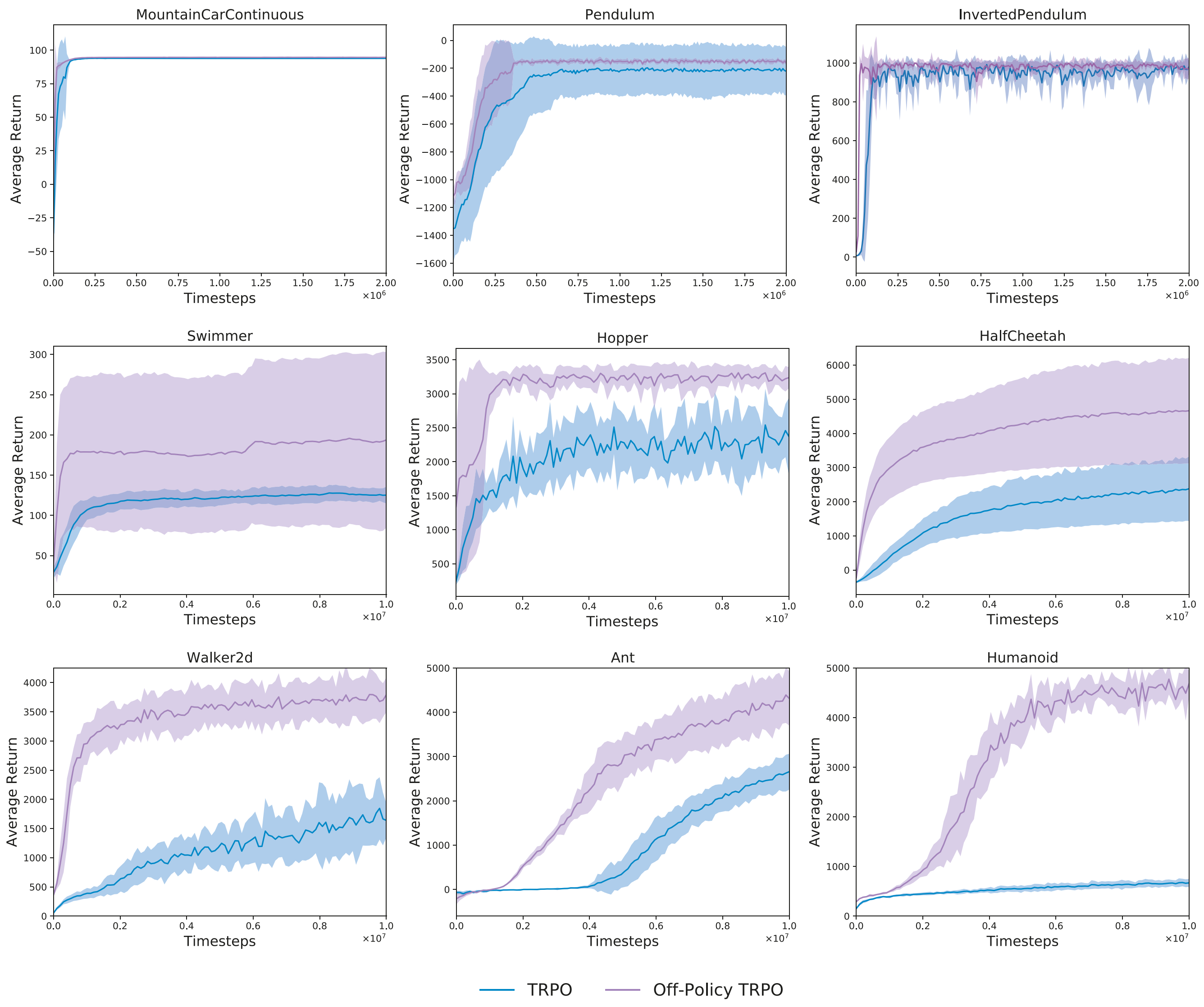


Fig. 5. Results of off-policy TRPO compared with TRPO on nine representative continuous control tasks. The shaded region indicates the standard deviation over ten random seeds. The X-axis represents the timesteps in the environment. The Y-axis represents the average return.

efficiently and achieve better performance in the majority of continuous control tasks compared with other off-policy trust region methods.

D. Effectiveness of the Proposed Off-Policy TRPO

1) Results on Monotonic Improvement Guarantee: We evaluate the monotonic improvement guarantee by validating the policy performance η in the proposed off-policy TRPO can be improved in each step. For a fair comparison, we conduct experiments on the proposed off-policy TRPO and other off-policy trust region methods (Q-Prop, IPG, ACER, and Trust-PCL) to contrast their differences. These off-policy trust region methods that can exploit both on- and off-policy data are chosen for comparison due to that the monotonic improvement guarantee in our method is for both on- and off-policy data. The experiments of this comparison are performed on a representative task, i.e., gridworld [4], which is simple enough to calculate the value of policy performance η, as illustrated in Fig. 4(a). The experimental results are shown in Fig. 4(b). From Fig. 4(b), we observe that the policy performance η value of the proposed off-policy TRPO monotonically increases as the number of updates increases during all these updates. We also note that, during some updates, the policy performance η values of other methods (Q-Prop, IPG, ACER, and Trust-PCL) decrease as the number of updates increases. Results from Fig. 4(b) validate that, among these off-policy trust region methods, only our method can guarantee the monotonic improvement of policy performance η.

2) Results on Using Off-Policy Data: We evaluate the effectiveness of using off-policy data by comparing our method with the TRPO method that exploits on-policy data only. In Fig. 5, the proposed off-policy TRPO achieves the same returns with fewer timesteps than TRPO on these nine tasks, which is because the utilization of off-policy data in our method reduces the need for interaction with the environment.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE III

COMPARISON OF COMPUTATIONAL COST. THE AVERAGE TIMESTEPS PER SECOND [26] AND STANDARD DEVIATIONS DURING TRAINING OVER TEN SEEDED RUNS. NOTE THAT THE RESULTS OF TRPO THAT CAN ONLY USE ON-POLICY DATA ARE INDICATED IN BOLDFACE. AMONG THE METHODS THAT CAN USE BOTH ON- AND OFF-POLICY DATA (Q-PROP, IPG, ACER, TRUST-PCL, AND THE PROPOSED OFF-POLICY TRPO), THE BEST RESULTS ARE INDICATED IN BOLDFACE

Timesteps/Second	Q-Prop	IPG	ACER	Trust-PCL	Off-Policy TRPO	TRPO
MountainCarContinuous	91.9 ± 1.0	91.8 ± 0.6	28.5 ± 1.9	125.3 ± 1.6	$\mathbf{126.2 \pm 3.5}$	$\mathbf{388.7 \pm 9.9}$
Pendulum	94.1 ± 0.5	93.6 ± 0.2	28.6 ± 1.1	113.2 ± 0.8	$\mathbf{125.4 \pm 4.5}$	$\mathbf{362.5 \pm 16.5}$
InvertedPendulum	91.0 ± 0.6	90.8 ± 0.7	29.8 ± 1.4	110.6 ± 2.5	$\mathbf{115.9 \pm 4.1}$	$\mathbf{359.1 \pm 6.5}$
Swimmer	90.2 ± 0.4	89.9 ± 0.5	27.4 ± 0.4	115.1 ± 0.8	$\mathbf{123.5 \pm 3.1}$	$\mathbf{353.7 \pm 12.5}$
Hopper	81.1 ± 0.6	88.2 ± 0.3	24.7 ± 1.0	108.6 ± 1.4	$\mathbf{116.7 \pm 6.4}$	$\mathbf{339.7 \pm 10.0}$
HalfCheetah	80.2 ± 0.3	79.3 ± 0.7	27.1 ± 1.0	111.7 ± 1.4	$\mathbf{125.6 \pm 2.2}$	$\mathbf{349.1 \pm 18.2}$
Walker2d	88.8 ± 0.9	88.1 ± 0.6	27.0 ± 0.9	112.5 ± 0.9	$\mathbf{118.5 \pm 2.7}$	$\mathbf{334.4 \pm 20.4}$
Ant	73.7 ± 0.5	70.8 ± 0.7	28.4 ± 3.4	93.8 ± 0.6	$\mathbf{116.5 \pm 4.7}$	$\mathbf{294.8 \pm 9.6}$
Humanoid	45.0 ± 0.9	62.1 ± 0.9	25.8 ± 1.0	89.8 ± 4.6	$\mathbf{107.4 \pm 3.8}$	$\mathbf{285.0 \pm 8.0}$

As shown in Fig. 5, our off-policy TRPO obtains higher returns than TRPO on these tasks as a consequence of off-policy training in our method. In Table II, it is noticeable that the maximal average returns of our method are higher than those of TRPO on these tasks. These results in Fig. 5 and Table II are especially significant on the more complex tasks, i.e., Swimmer, Hopper, HalfCheetah, Walker2d, Ant, and Humanoid, which is because the off-policy training in our method is stable enough to help to solve these complex tasks. The results from Fig. 5 and Table II validate that our off-policy TRPO can outperform TRPO in terms of sample efficiency and maximal average returns by making use of off-policy data.

3) Results on Computational Efficiency: We evaluate the effectiveness of computational efficiency by comparing our method with other trust region methods (TRPO, Q-Prop, IPG, ACER, and Trust-PCL) in terms of wall-clock time [26]. The specific comparison of computational cost is summarized in Table III, which shows the average timesteps per second [26] during training. The results in Table III are obtained with the same setup as previous experiments. As shown in Table III, the proposed off-policy TRPO achieves the maximal timesteps per second among the off-policy trust region methods, which can exploit both on- and off-policy data, i.e., Q-Prop, IPG, ACER, Trust-PCL, and off-policy TRPO. From Table III, we also note that our method achieves smaller timesteps per second than TRPO due to that our method needs extra computational time to train off-policy data. Nevertheless, compared with TRPO that can only utilize on-policy data, the utilization of off-policy data in our method can help reduce on-policy interaction with the environment. The results shown in Table III demonstrate that the proposed off-policy TRPO can achieve better computational efficiency than the other off-policy trust region methods (Q-Prop, IPG, ACER, and Trust-PCL).

VI. CONCLUSION

In this article, we develop a new surrogate objective function that can leverage both on- and off-policy data. We prove that the maximization of the proposed function can guarantee the monotonic improvement of policy performance. Based on such a theoretical guarantee, we develop a practical method (off-policy TRPO) that iteratively solves the optimization problem of the proposed function. The proposed off-policy TRPO can make use of both on- and off-policy data while guaranteeing monotonic improvement for policy performance, which provides a different insight for trust region methods using off-policy data. Experimental results demonstrate that our method improves the sample efficiency and average return over the state-of-the-art trust region methods in the majority of continuous control tasks. Moreover, experimental results validate the effectiveness of the proposed off-policy TRPO in terms of monotonic improvement guarantee, using off-policy data, and computational efficiency.

Even though the computational complexity analysis in terms of wall-clock time for these trust region methods in Section V-D is commonly used [26], [53], [54], it could be challenging to theoretically analyze their computational complexity in terms of convergence. Note that it could be interesting to apply our method to other fields of deep reinforcement learning, e.g., model-based reinforcement learning and multi-agent reinforcement learning.

APPENDIX A
PROOF OF POLICY IMPROVEMENT BOUND IN THEOREM 1

The proof of Theorem 1 in this article uses the techniques from the proof of Theorem 1 in [25]. Different from the proof in [25], we derive the lower bound of policy performance $\eta(\tilde{\pi})$ using our performance approximation $L_{\pi,\mu}(\tilde{\pi})$ in (10). In this appendix, we first introduce these techniques from [25]. We then show the difference between $\eta(\tilde{\pi})$ and $L_{\pi,\mu}(\tilde{\pi})$ in each timestep. Finally, we derive the lower bound of $\eta(\tilde{\pi})$ in Theorem 1.

We start by briefly introducing a lemma from [25], which shows that the policy performance difference $\eta(\tilde{\pi}) - \eta(\pi)$ can be decomposed as a sum of per-timestep advantage.

Lemma 1: Given two policies π and $\tilde{\pi}$

$$\eta(\tilde{\pi}) = \eta(\pi) + \mathbb{E}_{\tau\sim\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right]. \tag{30}$$

This expectation is taken over trajectories $\tau := (s_0, a_0, s_1, a_1, \ldots)$, and the notation $\mathbb{E}_{\tau\sim\tilde{\pi}}[\cdots]$ indicates that actions are sampled from $\tilde{\pi}$ to generate τ. The proof of Lemma 1 can be found in [25].

As defined in [25], $\bar{A}(s)$ represents the expected advantage of $\tilde{\pi}$ over π at state s

$$\bar{A}(s) = \mathbb{E}_{a\sim\tilde{\pi}(\cdot|s)}[A_\pi(s, a)]. \tag{31}$$

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Based on the definition of $\bar{A}(s)$, $\eta(\tilde{\pi})$ in Lemma 1 can be represented as

$$\eta(\tilde{\pi}) = \eta(\pi) + \mathbb{E}_{\tau\sim\tilde{\pi}}\left[\sum_{t=0}^{\infty}\gamma^t \bar{A}(s_t)\right]. \tag{32}$$

$L_{\pi,\mu}(\tilde{\pi})$ in Equation (10) can be written as

$$L_{\pi,\mu}(\tilde{\pi}) = \eta(\pi) + \mathbb{E}_{\tau\sim\mu}\left[\sum_{t=0}^{\infty}\gamma^t \bar{A}(s_t)\right]. \tag{33}$$

Note that the difference between $\eta(\tilde{\pi})$ and $L_{\pi,\mu}(\tilde{\pi})$ arises from each timestep. In order to bound the difference in each timestep, we subsequently introduce a measure of how much two policies agree and its corresponding lemma from [25].

Definition 1: (p, q) is an α-coupled policy pair if it defines a joint distribution $(a, \tilde{a})|s$, such that $P(a \neq \tilde{a}) \leq \alpha$ for all s. p and q will denote the marginal distributions of a and $\tilde{a}$, respectively.

That is, α-coupling means that, if we randomly choose a seed for our random number generator and then we sample from each of p and q after setting that seed, the results will agree for at least fraction $1-\alpha$ of seeds [25]. The corresponding lemma based on this definition is given as follows:

Lemma 2: Given that π and $\tilde{\pi}$ are α-coupled policies and their α value is α_π for all s

$$|\bar{A}(s)| \leq 2\alpha_\pi \max_{s,a}|A_\pi(s,a)|. \tag{34}$$

The proof of Lemma 2 can be found in [25].

Based on these above techniques from [25], we next introduce a lemma that bounds the difference between $\eta(\tilde{\pi})$ and $L_{\pi,\mu}(\tilde{\pi})$ in each timestep and give its proof in detail.

Lemma 3: Let $(\pi, \tilde{\pi})$ and $(\mu, \tilde{\pi})$ be α-coupled policy pairs, and their α values separately are α_π and α_μ. Then

$$\begin{aligned}|\mathbb{E}_{s_t\sim\tilde{\pi}}[\bar{A}(s_t)] - \mathbb{E}_{s_t\sim\mu}[\bar{A}(s_t)]| \\ \leq 4\alpha_\pi(1-(1-\alpha_\mu)^t)\max_{s,a}|A_\pi(s,a)|.\end{aligned} \tag{35}$$

Proof: A coupling over the trajectory distributions can be produced by coupled policy pair $(\mu, \tilde{\pi})$. That is, after setting the same random seed, the trajectories τ and $\tilde{\tau}$ are separately obtained according to policies μ and $\tilde{\pi}$. We consider the advantage of $\tilde{\pi}$ over π at timestep t ($\bar{A}(s_t)$) and decompose its corresponding expectation based on whether μ agrees with $\tilde{\pi}$ at all timesteps $i < t$.

Specifically, the decomposition of the expectation depends on the number of times that μ and $\tilde{\pi}$ disagree before timestep t. Here, we use n_t to denote this number, i.e., the number of times that $a_i \neq \tilde{a}_i$ for $i < t$. The specific expectation decomposition for actions sampled using $\tilde{\pi}$ is shown as follows:

$$\begin{aligned}\mathbb{E}_{s_t\sim\tilde{\pi}}[\bar{A}(s_t)] = P(n_t = 0)\mathbb{E}_{s_t\sim\tilde{\pi}|n_t=0}[\bar{A}(s_t)] \\ + P(n_t > 0)\mathbb{E}_{s_t\sim\tilde{\pi}|n_t>0}[\bar{A}(s_t)].\end{aligned} \tag{36}$$

The expectation decomposition for actions sampled using μ can similarly be represented as

$$\begin{aligned}\mathbb{E}_{s_t\sim\mu}[\bar{A}(s_t)] = P(n_t = 0)\mathbb{E}_{s_t\sim\mu|n_t=0}[\bar{A}(s_t)] \\ + P(n_t > 0)\mathbb{E}_{s_t\sim\mu|n_t>0}[\bar{A}(s_t)].\end{aligned} \tag{37}$$

It is noticed that $n_t = 0$ terms are equal [25]

$$\mathbb{E}_{s_t\sim\tilde{\pi}|n_t=0}[\bar{A}(s_t)] = \mathbb{E}_{s_t\sim\mu|n_t=0}[\bar{A}(s_t)] \tag{38}$$

since $n_t = 0$ indicates that μ and $\tilde{\pi}$ agreed on all timesteps less than t. Subtracting (36) and (37), we obtain

$$\begin{aligned}\mathbb{E}_{s_t\sim\tilde{\pi}}[\bar{A}(s_t)] - \mathbb{E}_{s_t\sim\mu}[\bar{A}(s_t)] = P(n_t > 0)(\mathbb{E}_{s_t\sim\tilde{\pi}|n_t>0}[\bar{A}(s_t)] \\ -\mathbb{E}_{s_t\sim\mu|n_t>0}[\bar{A}(s_t)]).\end{aligned} \tag{39}$$

Since $(\mu, \tilde{\pi})$ is the α-coupled policy pair and its α value is α_μ, $P(\mu, \tilde{\pi}$ agree at timestep $i) \geq 1-\alpha_\mu$, so $P(n_t = 0) \geq (1-\alpha_\mu)^t$, and

$$P(n_t > 0) \leq 1-(1-\alpha_\mu)^t. \tag{40}$$

Based on the above observation, the following inequality can be derived:

$$\begin{aligned}&|\mathbb{E}_{s_t\sim\tilde{\pi}|n_t>0}[\bar{A}(s_t)] - \mathbb{E}_{s_t\sim\mu|n_t>0}[\bar{A}(s_t)]| \\ &\quad\leq |\mathbb{E}_{s_t\sim\tilde{\pi}|n_t>0}[\bar{A}(s_t)]| + |\mathbb{E}_{s_t\sim\mu|n_t>0}[\bar{A}(s_t)]| \\ &\quad\leq \mathbb{E}_{s_t\sim\tilde{\pi}|n_t>0}[|\bar{A}(s_t)|] + \mathbb{E}_{s_t\sim\mu|n_t>0}[|\bar{A}(s_t)|] \\ &\quad\leq 4\alpha_\pi \max_{s,a}|A_\pi(s,a)|\end{aligned} \tag{41}$$

where the last inequality is derived according to Lemma 2.

By plugging (40) and (41) into (39), we finish the proof for Lemma 3

$$\begin{aligned}|\mathbb{E}_{s_t\sim\tilde{\pi}}[\bar{A}(s_t)] - \mathbb{E}_{s_t\sim\mu}[\bar{A}(s_t)]| \\ \leq 4\alpha_\pi(1-(1-\alpha_\mu)^t)\max_{s,a}|A_\pi(s,a)|.\end{aligned} \tag{42}$$

Finally, we obtain the difference bound between $\eta(\tilde{\pi})$ and $L_{\pi,\mu}(\tilde{\pi})$ and derive the lower bound of $\eta(\tilde{\pi})$ in Theorem 1. Specifically, this difference bound is obtained by summing the difference bound in Lemma 3 [25]. Subtracting (32) and (33) and defining $\epsilon = \max_{s,a}|A_\pi(s,a)|$

$$\begin{aligned}|\eta(\tilde{\pi}) - L_{\pi,\mu}(\tilde{\pi})| &= \sum_{t=0}^{\infty}\gamma^t|\mathbb{E}_{s_t\sim\tilde{\pi}}[\bar{A}(s_t)] - \mathbb{E}_{s_t\sim\mu}[\bar{A}(s_t)]| \\ &\leq \sum_{t=0}^{\infty}\gamma^t\cdot 4\epsilon\alpha_\pi(1-(1-\alpha_\mu)^t) \\ &= 4\epsilon\alpha_\pi\left(\frac{1}{1-\gamma} - \frac{1}{1-\gamma(1-\alpha_\mu)}\right) \\ &= \frac{4\epsilon\gamma\alpha_\pi\alpha_\mu}{(1-\gamma)(1-\gamma(1-\alpha_\mu))} \\ &\leq \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha_\pi\alpha_\mu.\end{aligned} \tag{43}$$

Based on the difference bound in (43), the lower bound of $\eta(\tilde{\pi})$ can be derived as

$$\eta(\tilde{\pi}) \geq L_{\pi,\mu}(\tilde{\pi}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha_\pi\alpha_\mu. \tag{44}$$

In order to replace α_π and α_μ by the TV divergence, we need to use [55, Proposition 4.7], which describes the correspondence between TV divergence and coupled random variables. Such correspondence is described as follows.

Suppose that p_X and p_Y are distributions with $D_{\mathrm{TV}}(p_X||p_Y) = \alpha$. Then, there exists a joint distribution

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

(X, Y) whose marginals are p_X and p_Y, for which $X = Y$ with probability $1 - \alpha$.

Such correspondence means that, if we have two policies p and q, we can define an α-coupled policy pair (p, q) by setting the $\alpha = \max_s D_{\text{TV}}(p(\cdot|s) \parallel q(\cdot|s))$ [25]. Thus, taking $\alpha_\pi = \max_s D_{\text{TV}}(\pi(\cdot|s) \parallel \tilde{\pi}(\cdot|s))$ and $\alpha_\mu = \max_s D_{\text{TV}}(\mu(\cdot|s) \parallel \tilde{\pi}(\cdot|s))$ in (44), Theorem 1 follows.

Appendix B
Upper Bound of α_μ

Note that TV divergence $D_{\text{TV}}(\mu(\cdot|s) \parallel \pi_{\text{new}}(\cdot|s))$ used in Theorem 1 satisfies

$$D_{\text{TV}}(\mu(\cdot|s) \parallel \pi_{\text{new}}(\cdot|s)) \leq D_{\text{TV}}(\mu(\cdot|s) \parallel \pi_{\text{old}}(\cdot|s)) + D_{\text{TV}}(\pi_{\text{old}}(\cdot|s) \parallel \pi_{\text{new}}(\cdot|s)). \quad (45)$$

Thus, α_μ can be represented as

$$\begin{aligned} \alpha_\mu &= \max_s D_{\text{TV}}(\mu(\cdot|s) \parallel \pi_{\text{new}}(\cdot|s)) \\ &\leq \max_s [D_{\text{TV}}(\mu(\cdot|s) \parallel \pi_{\text{old}}(\cdot|s)) \\ &\quad + D_{\text{TV}}(\pi_{\text{old}}(\cdot|s) \parallel \pi_{\text{new}}(\cdot|s))] \\ &\leq \max_s D_{\text{TV}}(\mu(\cdot|s) \parallel \pi_{\text{old}}(\cdot|s)) \\ &\quad + \max_s D_{\text{TV}}(\pi_{\text{old}}(\cdot|s) \parallel \pi_{\text{new}}(\cdot|s)) \\ &= D_{\text{TV}}^{\max}(\mu, \pi_{\text{old}}) + D_{\text{TV}}^{\max}(\pi_{\text{old}}, \pi_{\text{new}}). \end{aligned} \quad (46)$$

Appendix C
Definitions of the Average KL Divergence

The formulations of average KL divergence are defined as

$$\overline{D}_{\text{KL}}^{\rho_\mu}(\theta_{\text{old}}, \theta) := \mathbb{E}_{s\sim\rho_\mu}[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot|s) \parallel \pi_\theta(\cdot|s))] \quad (47)$$

$$\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\mu, \theta_{\text{old}}) := \mathbb{E}_{s\sim\rho_\mu}[\sqrt{D_{\text{KL}}(\mu(\cdot|s) \parallel \pi_{\theta_{\text{old}}}(\cdot|s))}] \quad (48)$$

$$\overline{D}_{\text{KL}}^{\rho_\mu,\text{sqrt}}(\theta_{\text{old}}, \theta) := \mathbb{E}_{s\sim\rho_\mu}[\sqrt{D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot|s) \parallel \pi_\theta(\cdot|s))}]. \quad (49)$$

References

[1] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[2] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, p. 484, 2016.

[3] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[5] O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans, "Trust-PCL: An off-policy trust region method for continuous control," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–14.

[6] A. Gruslys, W. Dabney, M. G. Azar, B. Piot, M. Bellemare, and R. Munos, "The reactor: A fast and sample-efficient actor-critic agent for reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–18.

[7] O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans, "Bridging the gap between value and policy based reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2775–2785.

[8] B. O'Donoghue, R. Munos, K. Kavukcuoglu, and V. Mnih, "Combining policy gradient and Q-learning," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–15.

[9] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–21.

[10] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 2094–2100.

[11] Z. Wang *et al.*, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.

[12] M. Hessel *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3215–3222.

[13] W. Meng, Q. Zheng, L. Yang, P. Li, and G. Pan, "Qualitative measurements of policy discrepancy for return-based deep Q-network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4374–4380, Oct. 2020.

[14] L. Yang, M. Shi, Q. Zheng, W. Meng, and G. Pan, "A unified approach for multi-step temporal-difference learning with eligibility traces in reinforcement learning," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 2984–2990.

[15] L. Shi, S. Li, L. Cao, L. Yang, and G. Pan, "TBQ(σ): Improving efficiency of trace utilization for off-policy reinforcement learning," in *Proc. 18th Int. Conf. Auto. Agents MultiAgent Syst.*, 2019, pp. 1025–1032.

[16] Z. Wang *et al.*, "Sample efficient actor-critic with experience replay," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–20.

[17] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–14.

[18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*. [Online]. Available: http://arxiv.org/abs/1707.06347

[19] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1856–1865.

[20] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[21] L. Shi, S. Li, Q. Zheng, M. Yao, and G. Pan, "Efficient novelty search through deep reinforcement learning," *IEEE Access*, vol. 8, pp. 128809–128818, 2020.

[22] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–14.

[23] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[24] E. Imani, E. Graves, and M. White, "An off-policy policy gradient theorem using emphatic weightings," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 96–106.

[25] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.

[26] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba, "Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5279–5288.

[27] R. Akrour, A. Abdolmaleki, H. Abdulsamad, J. Peters, and G. Neumann, "Model-free trajectory-based policy optimization with monotonic improvement," *J. Mach. Learn. Res.*, vol. 19, no. 1, pp. 565–589, 2018.

[28] G. Liu *et al.*, "Trust region evolution strategies," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 4352–4359.

[29] H. Liu, Y. Wu, and F. Sun, "Extreme trust region policy optimization for active object recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2253–2258, Jun. 2018.

[30] Y. Yu, S.-Y. Chen, Q. Da, and Z.-H. Zhou, "Reusable reinforcement learning via shallow trails," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2204–2215, Jun. 2018.

[31] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2042–2062, Jun. 2018.

[32] Y. Yang, Z. Guo, H. Xiong, D.-W. Ding, Y. Yin, and D. C. Wunsch, "Data-driven robust control of discrete-time uncertain linear systems via off-policy reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3735–3747, Dec. 2019.

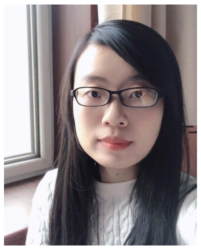
[33] S. Gu, T. Lillicrap, Z. Ghahramani, R. E. Turner, and S. Levine, "Q-prop: Sample-efficient policy gradient with an off-policy critic," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–13.

[34] S. S. Gu, T. Lillicrap, R. E. Turner, Z. Ghahramani, B. Schölkopf, and S. Levine, "Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3846–3855.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.

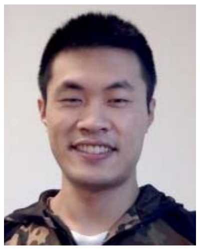
This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[35] X. Xu, Z. Huang, L. Zuo, and H. He, "Manifold-based reinforcement learning via locally linear reconstruction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 4, pp. 934–947, Apr. 2017.
[36] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.
[37] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1057–1063.
[38] T. Degris, M. White, and R. S. Sutton, "Off-policy actor-critic," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 179–186.
[39] S. Kakade and J. Langford, "Approximately optimal approximate reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2002, pp. 267–274.
[40] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.
[41] D. Pollard. (2000). *Asymptopia: An Exposition of Statistical Asymptotic Theory*. [Online]. Available: http://www.stat.yale.edu/pollard/Books/Asymptopia
[42] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *J. Res. Nat. Bur. Standards*, vol. 49, no. 6, pp. 409–436, 1952.
[43] R. Munos, T. Stepleton, A. Harutyunyan, and M. Bellemare, "Safe and efficient off-policy reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1054–1062.
[44] F. Pardo, A. Tavakoli, V. Levdik, and P. Kormushev, "Time limits in reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4045–4054.
[45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–15.
[46] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3207–3214.
[47] G. Brockman *et al.*, "OpenAI gym," 2016, *arXiv:1606.01540*. [Online]. Available: http://arxiv.org/abs/1606.01540
[48] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 5026–5033.
[49] B. L. Welch, "The generalization of student's' problem when several different population variances are involved," *Biometrika*, vol. 34, nos. 1–2, pp. 28–35, 1947.
[50] R. Akrour, J. Pajarinen, J. Peters, and G. Neumann, "Projections for approximate policy iteration algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 181–190.
[51] J. Pajarinen, H. L. Thai, R. Akrour, J. Peters, and G. Neumann, "Compatible natural gradient policy search," *Mach. Learn.*, vol. 108, nos. 8–9, pp. 1443–1466, Sep. 2019.
[52] H. W. Park, I. Grover, S. Spaulding, L. Gomez, and C. Breazeal, "A model-free affective reinforcement learning approach to personalization of an autonomous social robot companion for early literacy education," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 687–694.
[53] Y. Sun, X. Yuan, W. Liu, and C. Sun, "Model-based reinforcement learning via proximal policy optimization," in *Proc. Chin. Autom. Congr. (CAC)*, Nov. 2019, pp. 617–629.
[54] J. Nam, Y.-B. Kim, E. L. Mencia, S. Park, R. Sarikaya, and J. Fürnkranz, "Learning context-dependent label permutations for multi-label classification," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4733–4742.
[55] D. A. Levin and Y. Peres, *Markov Chains Mixing Times*. Providence, RL, USA: American Mathematical Society, 2017.



Wenjia Meng received the B.E. degree in software engineering from Shandong University, Jinan, China, in 2014. She is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China.

Her research interests include reinforcement learning and artificial intelligence.



Qian Zheng received the B.E. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 2011 and 2017, respectively.

He is currently a Research Fellow with the ROSE Lab, Nanyang Technological University, Singapore. He has published several articles in international journals and conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI), IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY (TIFS), IEEE TRANSACTIONS ON IMAGE PROCESSING (TIP), IEEE Conference on Computer Vision and Pattern Recognition (CVPR), and IEEE International Conference on Computer Vision (ICCV).

Dr. Zheng has served on the Executive Area Chairs Committee (EACC) of Vision And Learning SEminar (VALSE). He is also a reviewer of the IEEE TIP, CVPR, Asian Conference on Computer Vision (ACCV), and British Machine Vision Conference (BMVC).

Yue Shi received the B.E. degree in computer science from Zhejiang University, Hangzhou, China, in 2018, where he is currently pursuing the master's degree with the College of Computer Science and Technology.

His research interests include machine learning and reinforcement learning.

Gang Pan (Member, IEEE) received the B.Eng. and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1998 and 2004, respectively.

He is currently a Professor with the College of Computer Science and Technology and the Vice-Director of the State Key Lab of CAD&CG, Zhejiang University. He has coauthored more than 100 refereed articles and has 39 patents granted. His interests include artificial intelligence, brain-inspired computing, brain–machine interfaces, pervasive computing, and computer vision.

Dr. Pan was a recipient of NSF for Distinguished Young Scholars in 2019, the IEEE TCSC Award for Excellence (Middle Career Researcher) in 2018, and the CCF-IEEE CS Young Computer Scientist Award in 2016. He also received many technical awards, including the TOP-10 Achievements in Science and Technology in Chinese Universities in 2016, the National Science and Technology Progress Award in 2015, the Best Paper Award of ACM UbiComp'16, and the IEEE UIC Test-of-Time Paper Award in 2019. He also serves as an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON CYBERNETICS, and *Pervasive and Mobile Computing*.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 29,2021 at 03:10:02 UTC from IEEE Xplore. Restrictions apply.