Sample Efficient Reinforcement Learning with Double Importance Sampling Weight Clipping

1st Jiale Han
University of Science and Technology of China
Hefei, China
hanjiale@mail.ustc.edu.cn

2nd Mingxiao Feng
University of Science and Technology of China
Hefei, China
fmxustc@mail.ustc.edu.cn

3rd Wengang Zhou
Institute of Artificial Intelligence
Hefei Comprehensive Nation Science Center;
University of Science and Technology of China
Hefei, China
zhwg@ustc.edu.cn

4th Houqiang Li
Institute of Artificial Intelligence
Hefei Comprehensive Nation Science Center;
University of Science and Technology of China
Hefei, China
lihq@ustc.edu.cn

2023 IEEE Conference on Games (CoG) | 979-8-3503-2277-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/CoG57401.2023.10333147

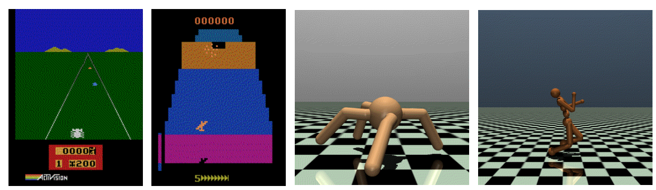
Abstract*—Proximal Policy Optimization (PPO) is a stable on-policy policy gradient (PG) method thanks to its clipped importance sampling (IS) weight objective of policy improvement. However, on-policy PG methods usually suffer from poor sample efficiency. In contrast, off-policy methods have demonstrated better sample efficiency by making more effective use of all collected samples during training. In this work, we aim to develop methods that inherit both the stability of on-policy PG methods and the data efficiency of off-policy methods. To this end, we present GeDISC, an off-policy algorithm that improves sample efficiency by reusing off-policy samples drawn from prior policies. Besides, we propose double IS weight clipping to control the high instability caused by off-policy data. We take the recently proposed generalized clipping mechanism for off-policy data as the first clipping to bound the policy update from the current policy and meanwhile we extend the standard clipping mechanism in PPO as the second clipping to prevent high variance and bias brought by extremely old samples. Extensive experiments on continuous and discrete control tasks show that the proposed new algorithm outperforms PPO and other SOTA PPO-based off-policy algorithms.

***Index Terms*—deep reinforcement learning, off-policy methods, policy gradient, policy optimization**

I. Introduction

In recent years, model-free deep reinforcement learning (RL) has shown remarkable advancements in simulated environments [1]. However, the application of these methods in real-world domains has been hampered by two major obstacles. First, model-free deep RL methods typically exhibit high variance and thus require a substantial amount of data collection, which can be hard and costly in real-world scenarios. Second, high-risk tasks have strong requirements for the stability offered by RL methods. It is quite difficult to satisfy both requirements simultaneously because stability and sample efficiency tend to conflict with each other.

This work was supported by the National Natural Science Foundation of China under Contract 61836011.



(a) Enduro (b) Zaxxon (c) Ant (d) Humanoid

Fig. 1: Snapshots of example environments. (a) and (b) are two video games in Atari [3]. (c) and (d) are two 3D physical simulation tasks in MuJoCo [4].

Model-free RL mainly consists of on- and off-policy methods. Proximal Policy Optimization (PPO) [2] is a popular Monte Carlo on-policy PG method that optimizes a policy improvement lower bound objective with clipped IS weight using samples collected by the current policy. PPO has demonstrated stable and strong performance across various tasks. A major drawback is the inherent high variance that necessitates collecting a large number of on-policy samples to accurately estimate the gradient, which results in sample intensive. In contrast, off-policy TD-style PG methods have better sample efficiency since these methods maintain a replay buffer to store all the collected samples and thus can reuse old samples multiple times to update the current policy. But these methods have to apply extensive hyperparameter tuning to attain stable performance because of convergence and instability issues.

There are heuristic efforts [5]–[8] to integrate data efficiency of off-policy methods into PPO. Dimension-Wise Importance Sampling Weight Clipping (DISC) [7] clips the IS weight of each action dimension and reuses old samples to enhance sample efficiency. Yet, DISC fails to exploit more off-policy data on low action-dimension tasks due to its method of filtering old samples. Additionally, DISC relies on factorized IS weights, making it unsuitable for discrete control tasks. Generalized Proximal Policy Optimization (GePPO) [8] offers theoretical policy improvement guarantees for the off-policy setting but struggles to handle more off-policy data because the high instability caused by much older samples can't be mitigated with its original generalized clipping mechanism. It is challenging to make effective use of the generated off-policy data to improve sample efficiency and meanwhile deliver stable and reliable performance throughout training.

To address this challenge, we introduce GeDISC, a **Ge**neralized Sample Efficient PPO-based algorithm with **D**ouble **I**mportance **S**ampling Weight **C**lipping. GeDISC reuses off-policy samples whose IS weights are close to 1 and filters old samples in a different way from DISC, which allows GeDISC to exploit more off-policy data. Besides, GeDISC applies double IS weight clipping for stability. We take the recently proposed generalized clipping mechanism as the first clipping to bound the policy update from the current policy and meanwhile we extend the standard clipping in PPO as the second clipping to prevent variance and bias brought by those extremely old samples. We demonstrate the strong performance of our algorithm through extensive experiments on both continuous and discrete control tasks (Fig. 1). Our key contributions are summarized as follows:

- We exploit more off-policy data than DISC by filtering old samples via average IS weight and thus enjoy a better exploration (Section III-A).
- We propose a novel double IS weight clipping mechanism that enables GeDISC to effectively exploit the off-policy data and meanwhile overcome the instability caused by old samples (Section III-B).
- We empirically demonstrate that GeDISC strikes a favorable balance between sample efficiency and stability on both MuJoCo and Atari environments (Section IV).

II. Related Work

A. On- and Off-policy PG

TRPO [9] achieves monotonic policy improvement based on Kullback-Leibler (KL) constraint. PPO [2] removes the KL constraint and instead clips the IS weight to prevent excessive policy changes. Popular off-policy PG methods such as DDPG [10], TD3 [11], and SAC [12] store all collected samples in a replay buffer and reuse these samples to update the current policy with TD learning [13]. Some efforts [14]–[16] have been made to combine on-policy learning with off-policy data to improve sample efficiency. ACER [15] and P3O [16] both apply a KL constraint to enhance stability and truncate large IS weights to mitigate high variance.

B. PPO with off-policy data

PPO. Consider the current policy π_k and the future policy π_θ, PPO [2] clips the IS weight $\rho = \frac{\pi_\theta(a|s)}{\pi_k(a|s)}$ as

$$\text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_k(a|s)}, 1-\epsilon, 1+\epsilon\right), \tag{1}$$

where $\text{clip}(x, l, h)$ means $\min(\max(x, l), h)$. The clipping mechanism reduces the possibility of the IS weight outside of the clipping interval $[1-\epsilon, 1+\epsilon]$. At each policy update, PPO optimizes the clipped surrogate objective as

$$L^{\text{PPO}}(\theta) = \mathop{\mathbb{E}}_{(s,a)\sim\pi_k}\Bigg[\min\Bigg(\frac{\pi_\theta(a|s)}{\pi_k(a|s)}A^{\pi_k}(s,a), \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_k(a|s)}, 1-\epsilon, 1+\epsilon\right)A^{\pi_k}(s,a)\Bigg)\Bigg]. \tag{2}$$

In practice, PPO collects an N-step trajectory following the current policy π_k, then uses GAE [17] to estimate the advantage $A^{\pi_k}(s,a)$. The objective (2) enables PPO to use stochastic gradient ascent for multiple epochs of minibatch update. Note that the IS weight $\frac{\pi_\theta(a|s)}{\pi_k(a|s)} = 1$ before each policy update.

PPO is on-policy and has to suffer from high variance. Accurately estimating the objective (2) necessitates a substantial number of on-policy samples. It's crucial to effectively utilize off-policy data to improve sample efficiency.

DISC. For continuous control tasks, PG methods [2], [9], [12] typically sample action from independent Gaussian distribution at each dimension. So the policy can be factorized into the action dimensions as $\pi_\theta(a_t|s_t) = \prod_{d=0}^{D-1}\pi_{\theta,d}(a_{t,d}|s_t)$, where $a_{t,d}$ is the action of d-th dimension, $\pi_{\theta,d}$ is the policy of d-th dimension, and D is the total action dimension. The IS weight ρ_t can be also factorized as $\rho_t = \prod_{d=0}^{D-1}\rho_{t,d}$, where $\rho_{t,d}$ is the IS weight of d-th dimension. DISC [7] clips $\rho_{t,d}$ of each dimension as $\text{clip}(\rho_{t,d}, 1-\epsilon, 1+\epsilon)$ to alleviate gradient vanishing. To enhance sample efficiency, DISC reuses old trajectories satisfying $\frac{1}{ND}\sum_{t=0}^{N-1}\sum_{d=0}^{D-1}\rho'_{t,d} < 1+\epsilon_b$, where N is the trajectory length, $\rho'_{t,d} := |\rho_{t,d} - 1| + 1$, and ϵ_b is a threshold parameter. $\epsilon_b = 0.1$ is the default setting.

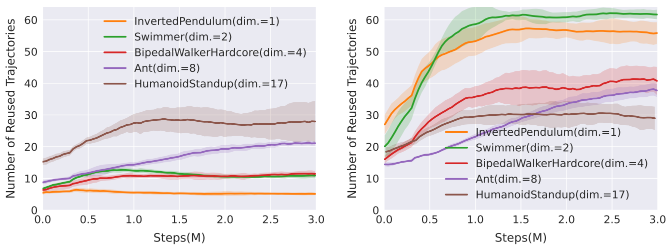
However, DISC fails to reuse more off-policy data on low action-dimensional tasks due to its method of filtering old samples. Additionally, DISC can't be applied for discrete control tasks, because the IS weight can't be factorized on these tasks.

GePPO. GePPO [8] develops a generalized clipping mechanism for off-policy data based on its generalized policy improvement lower bound, which makes it practical to exploit both on- and off-policy data in a principled way. Consider the last M policies π_{k-i}, $i = 0, 1, \ldots, M-1$, where π_k represents the current policy, the generalized clipping mechanism can be written as

$$\text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{k-i}(a|s)}, \frac{\pi_k(a|s)}{\pi_{k-i}(a|s)} - \epsilon, \frac{\pi_k(a|s)}{\pi_{k-i}(a|s)} + \epsilon\right). \tag{3}$$

Compared to the standard clipping mechanism (1) in PPO, GePPO clips the IS weight around the center of $\frac{\pi_k(a|s)}{\pi_{k-i}(a|s)}$ instead of 1. When $M = 1$, i.e. all samples are generated by the current policy π_k, (3) reduces to (1).

GePPO performs well when samples are generated from the last four prior policies, but struggles to cope with numerous off-policy data from older policies because the generalized clipping mechanism can't alleviate the huge instability caused by those off-policy samples.



(a) DISC (b) GeDISC

Fig. 2: The number of reused trajectories during training, where dim. is the action dimension. Comparing (a) with (b), it can be seen that GeDISC exploits more trajectories than DISC on most tasks.

III. Algorithm

A. Reusing Off-policy Samples

We maintain a replay buffer to store M prior trajectories $\{B_{k-i} \mid i = 0, \ldots, M-1\}$, where B_{k-i} is generated by the prior policy π_{k-i} and π_k is the current policy. If $M = 1$, the algorithm is on-policy otherwise it is off-policy. If the IS weights of the old samples deviate too much from 1, this suggests that these samples would bring huge bias and variance. Thus, we would not like to reuse all the trajectories in the replay buffer. Instead, we only consider these trajectories whose average IS weights are close to 1. Here, we follow [5], use $\rho_t' := |\rho_t - 1| + 1$ where ρ_t is the IS weight, to measure how much the IS weight deviates from 1. GeDISC filters trajectories following

$$\frac{1}{N}\sum_{t=0}^{N-1} \rho_t' < 1 + \epsilon_b, \tag{4}$$

where N is the length of the trajectory and ϵ_b is a threshold parameter. Here $\epsilon_b = 0.45$ is different from that of DISC [7] as described in Section II-B because we concern the average ρ_t' instead of $\rho_{t,d}'$ introduced in DISC.

Fig. 2 shows the number of reused old trajectories of DISC and GeDISC on some MuJoCo [4] tasks. Both DISC and GeDISC work well on the high action-dimensional task namely HumanoidStandup while GeDISC can exploit more old samples on all lower action-dimensional tasks. More off-policy samples for experience replay provide more policy gradients for policy update and yield better exploration, which is part of the reason for the strong performance of GeDISC on these tasks. On the other hand, more old samples also indicate a more complex and biased problem. It is challenging to make effective use of off-policy data while overcoming the instability brought by them. GeDISC addresses this challenge in Section III-B.

B. Double IS Weight Clipping

In order to exploit off-policy data effectively, we first consider the recently proposed generalized clipping mechanism (3). $\frac{\pi_k(a|s)}{\pi_{k-i}(a|s)}$, the center of the clipping range, typically deviates from 1 because prior policies can be different from the current policy. The IS weight $\frac{\pi_\theta(a|s)}{\pi_{k-i}(a|s)}$ begins from the center, then the generalized clipping bounds the changed IS weight around the center to ensure that π_θ does not deviate too much from the current policy π_k. However, if the center is far from 1 (as shown in Fig. 3a), extremely large or small IS weights bring high variance, which leads to instability as shown in Fig. 3b.

To address the high variance, we clip again with a wider clipping range after the first generalized clipping to bound the IS weight into a maximum tolerable interval. Hence, our double IS weight clipping can be written as

$$\mathrm{dclip}\left(\frac{\pi_\theta(a|s)}{\pi_{k-i}(a|s)}\right) = \mathrm{clip}\Bigg(\mathrm{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{k-i}(a|s)},\right. \\ \left.\frac{\pi_k(a|s)}{\pi_{k-i}(a|s)} - \epsilon_1, \frac{\pi_k(a|s)}{\pi_{k-i}(a|s)} + \epsilon_1\right), 1-\epsilon_2, 1+\epsilon_2\Bigg), \tag{5}$$

where $\mathrm{dclip}(\cdot)$ is the double clipping function with two factors ϵ_1 and ϵ_2. The inner (first) clipping bounds the policy update from the current policy no matter how much π_{k-i} deviates from π_k. The outer (second) clipping directly ignores those samples whose IS weights are far from 1 and prevents the IS weight from being extremely large or small, which safeguards against high variance and more bias. Intuitively, ϵ_2 should be larger than ϵ_1. As shown in Fig. 3b, the second clipping does work.

To control the variation of IS weight at each gradient step, we follow DISC [7] to use an explicit penalty on the IS weight: $J_{IS} = \mathbb{E}_t\left[\frac{1}{2}(\log(\rho_t))^2\right]$. J_{IS} helps for stability at the cost of extra slight bias. Thus, our objective for GeDISC is given by

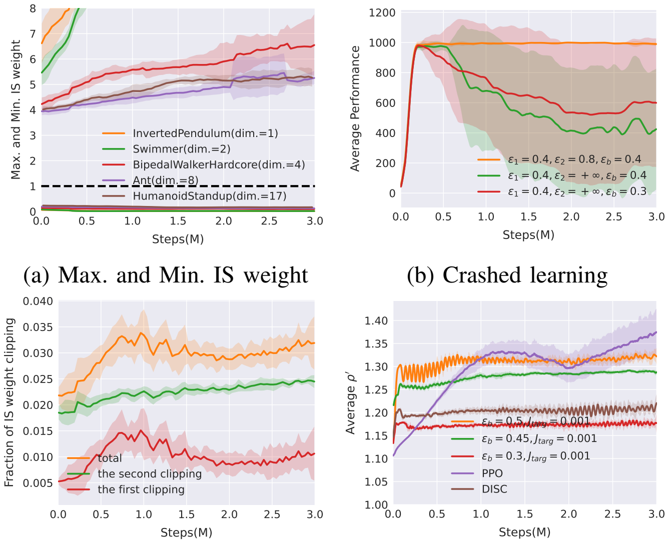
$$L(\theta) = \mathop{\mathbb{E}}_{i\sim v}\Bigg[\mathop{\mathbb{E}}_{(s,a)\sim\pi_{k-i}}\Bigg[\min\left(\frac{\pi_\theta(a|s)}{\pi_{k-i}(a|s)}A^{\pi_k}(s,a),\right. \\ \left.\mathrm{dclip}\left(\frac{\pi_\theta(a|s)}{\pi_{k-i}(a|s)}\right)A^{\pi_k}(s,a)\right)\Bigg]\Bigg] - \alpha_{IS}J_{IS}, \tag{6}$$

where $v \leq M$ is the number of trajectories satisfying (4), $(s,a) \sim \pi_{k-i}$ represents that the sample is generated by π_{k-i}, $\mathrm{dclip}(\cdot)$ denotes the double IS weight clipping as (5) and α_{IS} is the adaptive penalty coefficient according to

$$\begin{cases} \text{If} \quad J_{IS} < J_{targ}\ /\ 2, & \alpha_{IS} \leftarrow \alpha_{IS}\ /\ 1.5 \\ \text{If} \quad J_{IS} > J_{targ} \times 2, & \alpha_{IS} \leftarrow \alpha_{IS} \times 1.5 \end{cases}. \tag{7}$$

So that we can achieve a small target value of J_{targ} at each policy update. Similar to DISC [7], we compute the penalty J_{IS} only using the on-policy samples because J_{IS} with respect to all past policies would severely limit the gradient step.

Fig. 3c shows the clipping fraction of double IS weight clipping on Ant task. We can see that each clipping works respectively. Fig. 3d shows the average ρ' of GeDISC for $\epsilon_b = 0.3, 0.45, 0.5$ on Ant task, where ρ' is defined in Section III-A. Combining Fig. 2 and Fig. 3d, it can be seen that GeDISC could stably control the IS weight though



(a) Max. and Min. IS weight (b) Crashed learning

(c) Clipping fraction (d) Average ρ'_t

Fig. 3: (a) The average maximum and minimum IS weight during training on serval tasks without the second clipping. The threshold $\epsilon_b = 0.4$. It can be seen that maximum IS weights are far above 1.0 while minimum IS weights are far below 1.0. (b) The training curve on InvertedPendulum task. $+\infty$ means no second clipping, which leads to crashed learning. (c) The clipping fraction during training on Ant task. (d) Comparison of the average ρ'_t between GeDISC, DISC, and PPO on Ant task.

GeDISC reuses more old samples. Hence, we can say that GeDISC effectively makes use of more off-policy samples and meanwhile overcomes the instability caused by off-policy data.

C. Advantage Estimation

We have to estimate the advantage $A^{\pi_k}(s, a)$ using old samples collected from prior policies, which is the main source of bias that should be concerned about in the GeDISC gradient. Both DISC [7] and GePPO [8] combine GAE [17] and V-trace [18], a multi-step estimates correction with truncated importance sampling, to compute the advantage for low variance at the cost of some bias. Specifically,

$$\hat{A}^{\pi_k}(s_t, a_t) = \delta_t^V + \sum_{j=1}^{N-1} (\gamma\lambda)^j \left(\prod_{i=1}^{j} c_{t+i} \right) \delta_{t+j}^V, \tag{8}$$

where N is the trajectory length, $c_t = \min\left(1, \frac{\pi_k(a_t|s_t)}{\pi_{k-i}(a_t|s_t)}\right)$ is the truncated IS weight, $\delta_t^V = r(s_t, a_t) + \gamma V^{\pi_k}(s_{t+1}) - V^{\pi_k}(s_t)$ is the TD error [13], and λ is GAE hyperparameter. For those samples whose IS weights are far from 1, which induce high variance and very biased advantage estimates, double IS weight clipping would ignore their gradients and thus safeguard against both variance and bias as described in Section III-B.

The final algorithm is summarized in Algorithm 1.

Algorithm 1: GeDISC

Initialize parameters $\alpha_{IS} \leftarrow 1$;
for $k = 0, 1, 2, \ldots$ **do**
  Collect an on-policy trajectory B_k following π_k;
  Store B_k in replay buffer $\boldsymbol{R}$;
  Filter trajectories satisfying (4) from $\boldsymbol{R}$ as $\bar{\boldsymbol{R}}$;
  for *each epoch* **do**
    for *each gradient step* **do**
      Sample minibatch from $\bar{\boldsymbol{R}}$;
      maximize the empirical objective (6);
    end
  end
  Update α_{IS} as (7);
end

IV. Experiments

In this section, we seek to answer the following questions:

- Can GeDISC improve the sample efficiency of PPO [2] on both continuous and discrete control tasks?
- Does GeDISC work better than other existing sample efficient algorithms, such as ACER [15], DISC [7], and GePPO [8]?
- How important are reusing old samples and double IS weight clipping to GeDISC?
- How to tune those critical parameters properly?

We compare GeDISC against competitive baselines on the MuJoCo [4] environments and Arcade Learning Environment [3] (Atari) benchmarks, both interfaced through OpenAI Gym [19]. For the plots, The solid lines indicate the mean across different random seeds and the shaded region represents a standard deviation. Curves are smoothed uniformly for visual clarity. Hyperparameter settings are detailed in Appendix A.

A. Results on Continuous Control Tasks

We compare GeDISC against PPO [2], DISC [7], and GePPO [8] on six challenging continuous control tasks (Fig. 4). Most are MuJoCo [4] environments, except for BipedalWalkerHardcore which is powered by Box2d [20]. For all four algorithms, we use the same policy and value network as used in PPO, i.e. MLP with two hidden layers $(64, 64)$ and tanh activations. We set $\epsilon_1 = 0.4$, $\epsilon_2 = 0.8$, and $\epsilon_b = 0.45$. Each trial meets one evaluation every 4096 timesteps, where each evaluation reports the average reward over five episodes with no exploration. All algorithms on each environment are run for five random seeded trials.

Fig. 4 shows that GeDISC outperforms the baseline algorithms on all the continuous control tasks. Besides, we provide additional baseline results compared with other SOTA model-free RL algorithms in Appendix B.

B. Results on Discrete Control Tasks

We compare GeDISC against PPO [2] and ACER [15] on all 49 Atari games with raw pixels. We omit DISC [7] and GePPO [8] because DISC can't be applied for discrete control

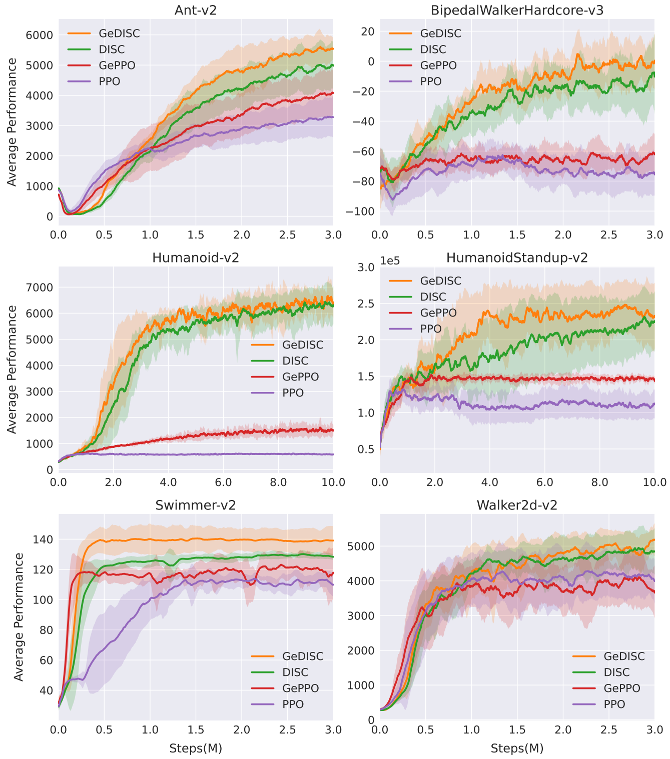


Fig. 4: Learning curves on the continuous control tasks.

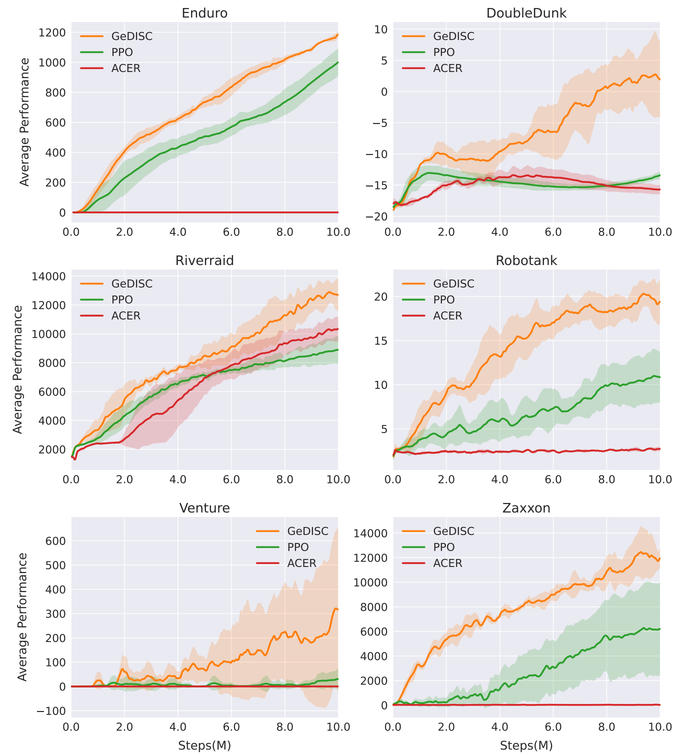


Fig. 5: Learning curves on the Atari games.

TABLE I: Number of games "won" by each algorithm.

Metric	ACER	PPO	GeDISC
(1) avg. episode reward over the entire training	17	5	**27**
(2) avg. episode reward over the last 100 episodes	12	12	**25**

tasks and the hyperparameter setting of GePPO for Atari is not given. For all three algorithms, we use the same policy network as that of Mnih et al. [21]. Atari environments are more sensitive to IS weight, so we set $\epsilon_1 = 0.1$, $\epsilon_2 = 0.4$, and $\epsilon_b = 0.1$. We follow PPO [2] to measure performance in two metrics: (1) average reward per episode over the entire training period, and (2) average reward per episode over the last 100 episodes of training. The former focuses on sample efficiency while the latter prefers the final performance. All algorithms on each environment are run for three random seeded trials.

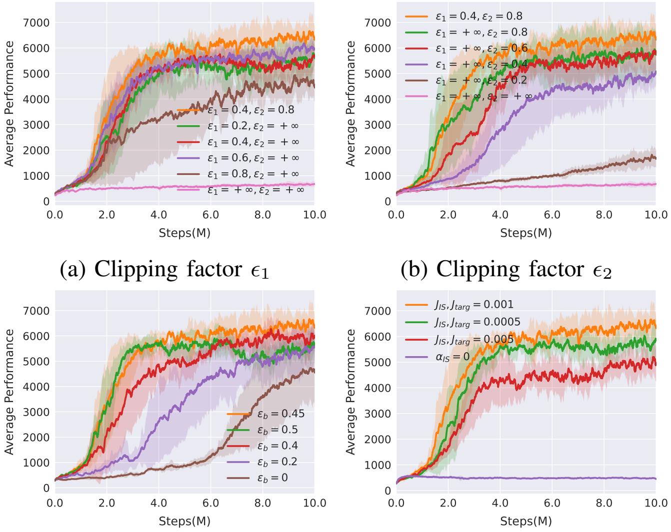
Table I shows that GeDISC won most games under both metrics, where "won" means achieving the highest performance by averaging the metric across three trials. Fig. 5 shows that GeDISC outperforms PPO and ACER with clear margin. Results for all 49 Atari games are shown in Appendix C.

C. Ablation Study

In this subsection, we further investigate which components of GeDISC are important and introduce how we tune some critical hyperparameters intuitively. Fig. 6 shows the results of the ablation study on Humanoid task.

Double Clipping Factor ϵ_1 and ϵ_2. As described in Section III-B, double IS weight clipping works for stability: the first clipping deals with off-policy data to bound the policy update from the current policy and the second clipping clips those samples whose IS weights are far from 1. We now observe the effect of each clipping separately. Fig. 6a shows the performance of GeDISC only with the first clipping: $\epsilon_1 = 0.2, 0.4, 0.8, +\infty$, where $+\infty$ means no clipping. We can see that, without the second clipping, high variance and bias do harm the performance. ϵ_1 should be reasonably small because large policy update causes instability. Fig. 6b shows the performance of GeDISC only with the second clipping: $\epsilon_2 = 0.2, 0.4, 0.8, +\infty$. Without the first clipping, excessively large policy update is detrimental to stability. If ϵ_2 is small, most of the samples are clipped resulting in poor performance. So ϵ_2 should be reasonably large to only prevent extreme IS weights, but should not be larger than 1 because the clipping interval $[1-\epsilon_2, 1+\epsilon_2]$ won't work for those samples whose IS weights are far below 1.

Threshold ϵ_b. As described in Section III-A, we reuse old trajectories that satisfy (4). Threshold ϵ_b roughly controls the variance and bias brought by the old samples. Fig. 6d shows the performance of GeDISC with several values of threshold: $\epsilon_b = 0, 0.2, 0.4, 0.45, 0.5$, where $\epsilon_b = 0$ indicates that no old samples are reused. If ϵ_b is too small, GeDISC can not exploit enough old samples. If ϵ_b is too large, GeDISC has to suffer from huge bias caused by extremely old samples. We observe that ϵ_b around 0.45 can well balance the sample efficiency and



(a) Clipping factor ϵ_1 (b) Clipping factor ϵ_2

(c) Threshold ϵ_b (d) IS weight penalty J_{IS}

Fig. 6: Abaltion study results on Humanoid-v2

bias.

IS Weight Penalty J_{IS}**.** The IS weight penalty J_{IS} is proposed by DISC [7]. In Section III-B, we apply J_{IS} in GeDISC to control the variation of IS weight at each gradient step. In Fig. 6c, $\alpha_{IS} = 0$ indicates no J_{IS}. We can see that $J_{IS} = 0.001$ works well.

D. Parameter tuning

In Section IV-C, we have mentioned how to tune some critical parameters intuitively. In short, ϵ_1 has a similar role with PPO's clipping factor and should be reasonably small. ϵ_2 should be in $(\epsilon_1, 1.0]$. Threshold ϵ_b roughly controls the average IS weights and thus is related to ϵ_1 and ϵ_2. We gradually find that GeDISC works well when ϵ_1 and ϵ_b are similar. Besides, some important metrics also help for parameter tuning, such as the amount of reused trajectories and the average IS weight ρ'. Finally, we keep $\epsilon_1 = 0.4$, $\epsilon_2 = 0.8$, $\epsilon_b = 0.45$ for all MuJoCo tasks. Due to Atari environments are more sensitive to large IS weights, we keep $\epsilon_1 = 0.1$, $\epsilon_2 = 0.4$, $\epsilon_b = 0.1$ for all Atari games.

V. Conclusion

In this paper, we introduce GeDISC, a generalized sample efficient PPO-based algorithm that reuses the old samples whose average IS weights do not deviate too much from 1 and applies double IS weight clipping for stability. The first clipping bounds the policy update from the current policy while the second clipping prevents high variance and bias. Extensive results show that GeDISC can significantly improve the sample efficiency of PPO and deliver stable and better performance than other SOTA PPO-based algorithms on both continuous and discrete control tasks. Future works may focus on the threshold parameter ϵ_b about adjusting its value along the training process.

Appendix A
Implementation details

For OpenAI GYM continuous control tasks, the hyperparameters of all algorithms are detailed in Table II. To prevent convergence to local optimum, the learning rate anneals from 0.0003 to 0.0001 and then remains constant. GePPO [8] uses adaptive learning rate as described in its original paper.

For Atari environments, the hyperparameters of all algorithms are detailed in Table III. We use the implementation of OpenAI baselines [22] for the ACER [15] baseline on Atari.

Appendix B
Additional baseline results on continuous control tasks

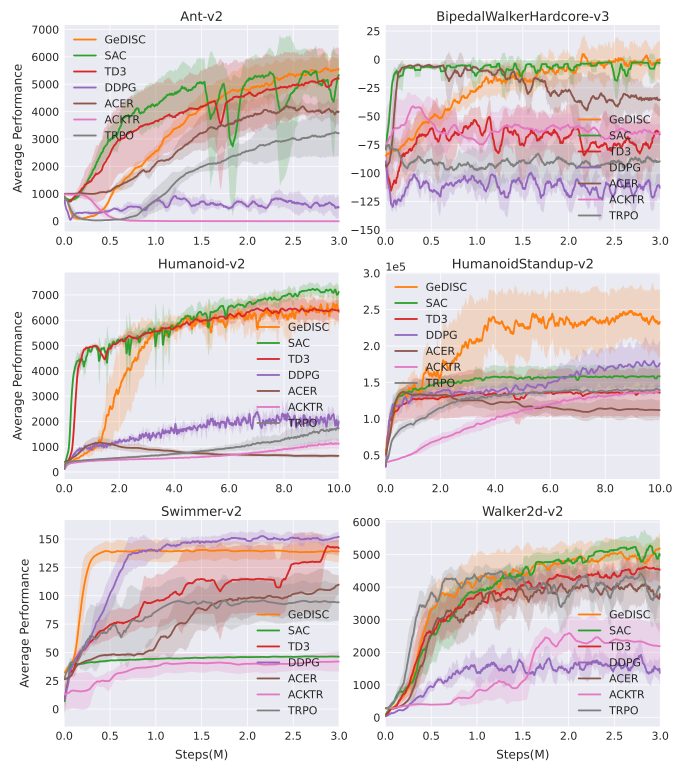


Fig. 7: Learning Curves of GeDISC and Other SOTA model-free RL Algorithms on continuous control tasks.

In order to demonstrate the strong performance of our algorithm, we compare GeDISC with several other SOTA model-free RL algorithms: DDPG [10], TRPO [9], ACER [15], ACKTR [23], TD3 [11], and SAC [12]. As shown in Fig. 7, GeDISC delivers the highest performance on HumanoidStandup, which other SOTA RL algorithms could not catch up with. Besides, GeDISC also shows comparable competitive performance on other tasks.

Appendix C
Experimental results on all 49 Atari games

Learning curves of all 49 Atari games are shown in Fig. 8.

TABLE II: Hyperparameter setting of PPO, GePPO, DISC, and GeDISC for continuous control tasks.

Hyperparameter	GePPO	PPO	DISC	GeDISC
IS weight Clipping factor	0.1	0.2	0.4	$\epsilon_1 = 0.4, \epsilon_2 = 0.8$
Trajectory length (N)	1024	2048	2048	2048
Discount factor (γ)	0.99	0.99	0.99	0.99
GAE (λ)	0.95	0.95	0.95	0.95
Epochs per update	10	10	10	10
Minibatches per epoch	32	32	32	32
Optimizer	Adam	Adam	Adam	Adam
Learning rate	Adaptive	max(0.0001, Anneal(0.0003, 0))		
Policy distribution	Gaussian distribution			
Policy and value network	FC(64)-FC(64) with tanh activations			
Threshold (ϵ_b)	-	-	0.1	0.45
Replay length (M)	-	-	64	64
IS weight penalty J_{targ}	-	-	0.001	0.001
Initial α_{IS}	-	-	1	1

TABLE III: Hyperparameter setting of ACER, PPO, and GeDISC for Atari environments.

Hyperparameter	ACER	PPO	GeDISC
IS weight Clipping factor	10	0.1	$\epsilon_1 = 0.1, \epsilon_2 = 0.4$
Trajectory length (N)	20	128	128
Discount factor (γ)	0.99	0.99	0.99
GAE (λ)	-	0.95	0.95
Entropy regularization	0.01	0.01	0.01
Number of environments	8	8	8
Epochs per update	Possion(4)	4	4
Minibatches per epoch	-	4	4
Optimizer	RMSProp	Adam	Adam
Learning rate	0.0007	Anneal(0.00025, 0)	Anneal(0.00025, 0)
Policy distribution	Categorical distribution		
Policy network	Conv$(32, 8 \times 8, 4)$-Conv$(64, 4 \times 4, 2)$-Conv$(64, 3 \times 1, 1)$-FC(512) with relu activations		
	Use Trust region: True	-	Threshold (ϵ_b): 0.1
	Replay buffer size: 5×10^4	-	Replay length (M): 16
	Momentum factor: 0.99	-	IS weight penalty J_{targg}: 0.001
	Maximum KL: 1	-	Initial α_{IS}: 1

REFERENCES

[1] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *ICML*, 2016.

[2] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[3] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, jun 2013.

[4] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[5] S. Han and Y. Sung, "Amber: Adaptive multi-batch experience replay for continuous action control," *arXiv:1710.04423*, 2017.

[6] F. Sovrano, "Combining experience replay with exploration by random network distillation," in *CoG*, 2019.

[7] S. Han and Y. Sung, "Dimension-wise importance sampling weight clipping for sample-efficient reinforcement learning," in *ICML*, 2019.

[8] J. Queeney, Y. Paschalidis, and C. G. Cassandras, "Generalized proximal policy optimization with sample reuse," in *NeurIPS*, 2021.

[9] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *ICML*, 2015.

[10] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *ICLR*, 2016.

[11] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *ICML*, 2018.

[12] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *ICML*, 2018.

[13] R. S. Sutton, A. G. Barto *et al.*, "Introduction to reinforcement learning," 1998.

[14] S. Gu, T. Lillicrap, Z. Ghahramani, R. E. Turner, and S. Levine, "Q-prop: Sample-efficient policy gradient with an off-policy critic," in *ICLR*, 2017.

[15] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas, "Sample efficient actor-critic with experience replay," in *ICLR*, 2017.

[16] R. Fakoor, P. Chaudhari, and A. J. Smola, "P3o: Policy-on policy-off policy optimization," in *UAI*, 2020.

[17] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *ICLR*, 2016.

[18] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning *et al.*, "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *ICML*, 2018.

[19] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv:1606.01540*, 2016.

[20] E. Catto, "Box2d: A 2d physics engine for games," 2011. [Online]. Available: https://box2d.org/

[21] 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 *ICML*, 2016.

[22] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, and P. Zhokhov, "Openai baselines," 2017. [Online]. Available: https://github.com/openai/baselines

[23] 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 *NeurIPS*, 2017.

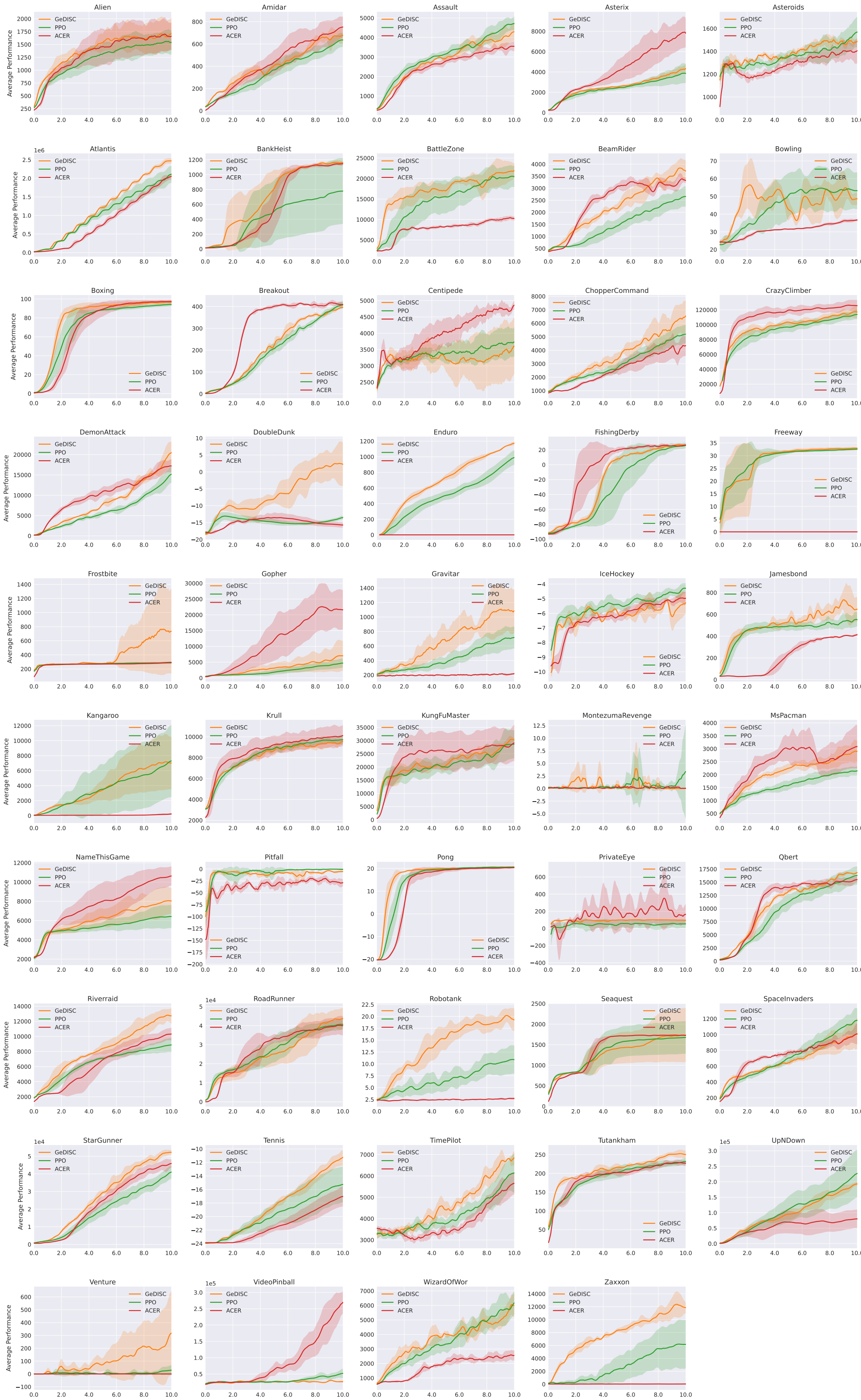


Fig. 8: Comparison of GeDISC, PPO, and ACER on all 49 Atari games.