Machine Learning (2022) 111:977–995
https://doi.org/10.1007/s10994-021-06083-7

Improve generated adversarial imitation learning with reward variance regularization

Yi-Feng Zhang[1] · **Fan-Ming Luo**[1] · **Yang Yu**[1]

Received: 15 May 2021 / Revised: 8 August 2021 / Accepted: 22 September 2021 /
Published online: 3 January 2022
© The Author(s), under exclusive licence to Springer Science+Business Media LLC, part of Springer Nature 2021

Abstract
Imitation learning aims at recovering expert policies from limited demonstration data. Generative Adversarial Imitation Learning (GAIL) employs the generative adversarial learning framework for imitation learning and has shown great potentials. GAIL and its variants, however, are found highly sensitive to hyperparameters and hard to converge well in practice. One key issue is that the supervised learning discriminator has a much faster learning speed than the reinforcement learning generator, making the generator gradient vanishing. Although GAIL is formulated as a zero-sum adversarial game, the ultimate goal of GAIL is to learn the generator, thus the discriminator should play the role more like a teacher rather than a real opponent. Therefore, the learning of the discriminator should consider how the generator could learn. In this paper, we disclose that enhancing the gradient of the generator training is equivalent to increase the variance of the fake reward provided by the discriminator output. We thus propose an improved version of GAIL, GAIL-VR, in which the discriminator also learns to avoid generator gradient vanishing through regularization of the fake rewards variance. Experiments in various tasks, including locomotion tasks and Atari games, indicate that GAIL-VR can improve the training stability and imitation scores.

Keywords Imitation learning · Reinforcement learning · Generative adversarial model · Discriminator reward

Editors: Yu-Feng Li, Mehmet Gönen, Kee-Eung Kim.

✉ Yi-Feng Zhang
zhangyf@lamda.nju.edu.cn

Fan-Ming Luo
luofm@lamda.nju.edu.cn

Yang Yu
yuy@nju.edu.cn

[1] State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing, China

1 Introduction

In recent years, Imitation Learning (IL) has been successfully applied in many domains, including robotics (Wu et al. 2018), natural language processing (Li et al. 2019), as well as autonomous driving (Bhattacharyya et al. 2018). IL tries to use limited expert knowledge (usually expert trajectories) to recover policy or even environments (Shi et al. 2019; Chen et al. 2020) by matching actions of expert data and the learning policies. Inverse Reinforcement Learning (IRL; Russell 1998) is one of the successful approaches for this setting, which tries to learn a cost function from expert demonstrations. Lately, the generative-discriminative framework (Ho and Ermon 2016) has been given research attentions (Song et al. 2018; Chen et al. 2020) in IRL, resulting in performance breakthroughs.

However, such a framework may still fail in many environments, mainly due to a poorly-learned RL generator. We conducted studies on such a phenomenon. We found that the imbalance between generator and discriminator frequently occurs in such a framework, especially in GAIL, and causes severe training problems. One of the key issues is that the supervised learning discriminator learns much faster than the RL generator. The discriminator is designed to tell the difference between expert data and generator data. A fast learner tends to depart such data at the very beginning of training, but at that time the generator is still in initial stages and cannot produce expert-like data. This may make the discriminator easily tell the difference between expert data and generated data, making it provide equally low rewards to the generator without prominent directions. In this case, it is common to finally have a well-trained discriminator but a poor generator that suffers from gradient vanishing. However, in GAIL, our goal is to have a well-trained generator, while the discriminator should act more like a teacher, guiding the generator to the ultimate convergence, by giving generator proper fake rewards that can keep the generator from gradient vanishing.

In this paper, we disclose that enhancing the gradient of generator training can be realized by increasing reward variance of the generator. A step further, we discover that the advantage function, in actor-critic RL algorithms, guarantees the update of critic network, so tuning the advantage function value variance is a more direct way to guide the RL agent. We prove that the variance of advantage function value mainly comes from the following two aspects: (1) The cumulative reward of generated state-action pairs; (2) The value function of states. Thus, the optimization can be staged into two steps: (1) Optimizing the value network for getting the optimized value function parameters. (2) Regularizing the discriminator network for getting the best variance of the advantage function value. Thus we presented our algorithm, GAIL with variance regularization (GAIL-VR), to do the optimization.

Experiments in various tasks are conducted. An average of 30% convergence speedup is recorded, as well as imitation score increment in at least 4 Atari games and 4 MuJoCo/Pybullet locomotion environments. Also, we find that GAIL-VR can help prevent reward drop after extensively long training episodes and perform better with data or parameter constraints, meaning that GAIL-VR is more adaptable in complex environments.

2 Background

2.1 Markov decision process and reinforcement learning

A Markov Decision Process (MDP; Sutton and Barto 1998) is often formalized by a tuple $(\mathcal{S}, \mathcal{A}, P, r, \rho, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $P(s'|s,a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$

the transition distribution, $r(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, $\rho_0(s) : \mathcal{S} \to [0,1]$ the initial state distribution, and $\gamma \in [0,1)$ the discount factor. Policy $\pi(a|s) : \mathcal{S} \times \mathcal{A}$ is the distribution over action conditioned on the current state. The discounted cumulative return is $\hat{R}(s_0,a_0) = \sum_{t=0}^{T} \gamma^t r(s_t,a_t)$, where $a_t \sim \pi(a_t|s_t), s_{t+1} \sim P(s_{t+1}|s_t,a_t)$. $Q^\pi(s,a) = \mathbb{E}[\hat{R}(s_0,a_0)|s_0 = s, a_0 = a]$ is the action value function that measures the expected return at the state-action pair $(s,\ a)$. $V^\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}[Q^\pi(s,a)]$ is the value function. The advantage function $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ denotes the advantage of action a at state s. We mainly focus on the actor-critic algorithms (Schulman et al. 2017, 2015), which consist of two components: actors to execute the policies and critics to score the policies.

2.2 Generative adversarial imitation learning (GAIL)

Imitation learning (IL) recovers the expert policy from the demonstrations and does not need any information of reward function. Current IL methods can be roughly divided into two categories: Behaviour Cloning (BC) and Inverse Reinforcement Learning (IRL) (Ng and Russell 2000; Abbeel and Ng 2004). BC tries to imitate the expert by maximizing the likelihood of the state-action pairs in the demonstrations. However, since the objective of BC does not consider the distribution the policy will generate in the environment, BC often suffers from covariate shift issues (Ross and Bagnell 2010; Ross et al. 2011).

In contrast to BC, IRL does not directly optimize the policy according to the expert data. In IRL, a reward function is first recovered. The policy is optimized with the reward function via reinforcement learning methods, instead of supervised learning in BC. Recently, some methods based on adversarial framework have been proposed and outperform conventional IRL methods in terms of data efficiency (Ho and Ermon 2016; Finn et al. 2016b, a; Fu et al. 2017). Among these methods, Generative Adversarial Imitation Learning (GAIL; Ho and Ermon 2016) adopted a discriminator network $D(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ to provide the agent with rewards. The target of the discriminator is distinguishing whether a state-action pair comes from expert demonstration or is generated by the agent. The optimization objective of the discriminator $D(s,a)$ is formalized as a standard cross entropy loss:

$$\max_D \ \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}\left[\log D(s,a)\right] + \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^E}\left[\log(1-D(s,a))\right],$$

where $\mathcal{D}^\pi$ and $\mathcal{D}^E$ denote the state-action pair set generated by agent and expert respectively. Reward of a state-action pair is measured by the similarity of it to the data in the expert data: $r(s,a) = -\log(1-D(s,a))$. To minimize the discrepancy between the distribution generated by the agent and expert, the agent needs to maximize its cumulative reward, which can be reached via RL. Hence, the optimization objective of the agent can be written as:

$$\max_\pi \ \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}\left[-\log(1-D(s,a))\right].$$

3 Related work

The theoretical convergence of GAIL has been proved. Chen et al. (2020) revealed that the convergence of the minimax optimization in GAIL can be proved using stochastic first-order optimization algorithms. However, since there is a considerable gap between the

theoretical convergence and the actual performance of GAIL, there are some modifications focusing on sample efficiency and convergence in various environments. Kostrikov et al. (2019) addressed the reward bias provided by discriminator in GAIL, and pointed out that an unbiased reward can make the algorithm get better performance in environments that have short rounds and need quick actions. However, such improvement does not consider the long-term convergence or stability, and can work in only a few environments. Peng et al. (2019) focused on the information given to generators and discriminators. They borrowed the idea of the information bottleneck (Slonim and Tishby 2000) to encode data into latent space and constrain the information given to generator and discriminator. But this work constrains the problem to transfer learning and does not have a universal application potential on all IRL problems. Baram et al. (2017) tried to address the fragile convergence of GAIL by training policies using the exact gradient of the discriminator, which is like the training fashion of GAN. They introduce a forward model and use Stochastic Value Gradient (Heess et al. 2015) as the base RL algorithm. Besides, Fu et al. (2017) and Geng et al. (2020) tried to recover the ground-truth reward under certain conditions. By recovering such reward function, the transferability of the learned reward can be improved.

To the best of our knowledge, there is no work directly considering the imbalance between the generator and the discriminator and the result of crucial gradient loss, which are key issues our work focuses on.

4 GAIL with variance regularization

In this section, we will focus on GAIL's two main issues when using the generator-discriminator framework and try to depict the difference between GAIL and GAN, which will significantly influence the performance.

4.1 Learning speed imbalance between generator and discriminator

In GAIL, the generator interacts with the discriminator in an adversarial manner. The discriminator tries to give low rewards to the generator by solving a classification problem. Meanwhile, the generator aims at obtaining higher rewards via reinforcement learning, e.g. policy gradient. One significant difference between GAN and GAIL is that, in GAN the generator and the discriminator are solving classification problems by supervised learning, but in GAIL while the discriminator keeps using supervised learning, the generator are trained by RL using rewards. However, the variance of the gradient estimation of reinforcement learning is much higher than supervised learning (Baram et al. 2017), and the partiality or delay of the reward signal as well as the sequential decision behavior of MDP in stochastic environments makes the exploration efficiency of reinforcement learning not satisfactory. As a result, the learning speed of the reinforcement learning will be slower than supervised learning. Thus, the learning speed of the discriminator will be faster than the generator, which is where the learning speed imbalance between the generator and the discriminator comes from.

Such an imbalance will influence the convergence of GAIL. At the beginning of the training, since the discriminator is commonly initialized randomly, it cannot distinguish the expert data and the generated data at all. As the training proceeds, the discriminator will gradually possess the ability of separating the data, and provides the generated data with low rewards and the expert data with high rewards. The generator will update its policy to

obtain higher rewards, and thus be more and more similar to the expert policy. However, with higher learning speed, the discriminator will be able to entirely depart the generated data and the expert data. As a result, the discriminator can figure out every state-action pair generated by the generator and give a low reward to it, rewards the generator gets will be low and have little variation. Figure 1 is a typical illustration of this problem, showing that the reward obtained by the expert data and the reward obtained by the generator are not even in the same order of magnitude. Such low-mean and low-variance rewards will harm the training of the generator. To depict the harm, we take the loss function of TRPO (Schulman et al. 2015) as an example. The loss function of TRPO is $J(\theta) = \mathbb{E}_{s,a\sim\rho_{\text{old}}} \frac{\pi_\theta}{\pi_{\text{old}}} A^{\pi_{\text{old}}}$, where π_{old} denotes the sampling (behavior) policy, ρ_{old} is the distribution generated by π_{old}. The gradient of $J(\theta)$ w.r.t. θ is proportional to $A^{\pi_{\text{old}}}$, i.e., the advantage function. However, as the generator obtains low-mean and low-variance rewards from the discriminator, the return discrepance between different action at a state will be small. Because no matter what action the agent takes, the discriminator is able to classify it to fake data and gives a low reward to the agent. Thus, the advantage function, which measure the advantage of taking an action in a state, will be small. Meanwhile, the policy gradient, $\nabla_\theta J(\theta) = \frac{\nabla_\theta \pi_\theta}{\pi_{\text{old}}} A^{\pi_{\text{old}}}$, which is proportional to the advantage function, will be small. Such a gradient size reduction will slow down the training of the generator, resulting that the discriminator will be stronger than the generator. As the training proceeding, the learning speed imbalance between them will be larger and larger. The learning of the generator will be slower and slower and be stuck eventually.

In Fig. 2, we present two curves concerning the discriminator loss and corresponding policy gradient size during GAIL training. At the beginning of the training, due to the low-variance gradient estimation of supervised learning, the discriminator loss suddenly reduce to a small number. Meanwhile, the gradient size of the generator also reduce fast. Whereas, the generator still possesses certain learning ability, which increases the discriminator loss. However, the gradient size of the generator continuously reduces in the subsequent episodes. At around 280 episodes, the generator fails to compete with the discriminator. Finally, the discriminator loss and the gradient size reduce fast. The learning ability of the generator and the discriminator is completely out of balance. The generator almost stops learning, resulting the training of GAIL fails.

In the following subsection, we will introduce GAIL-VR to alleviate the such an imbalance essentially by increasing the advantage variance.

Fig. 1 A typical average given reward (Left) and reward standard error (Right) for each transition, compared between expert data and generated data

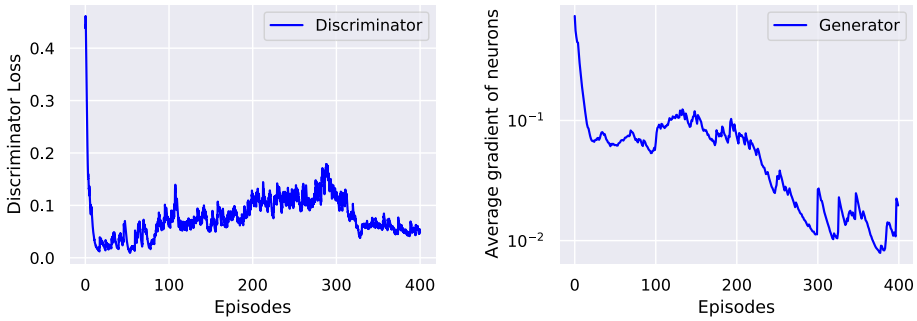


Fig. 2 Left: Discriminator loss during GAIL training. Right: Average gradient of RL generator neurons

4.2 Solution: discriminator reward variance loss

RL algorithms rely on the difference between environment-offered rewards and the current value function, e.g., the advantage function in actor-critic algorithms. In most of the actor-critic algorithms, the empirical advantage function can be computed as:

$$A(s,a) = r(s,a) + \gamma V_\phi(s') - V_\phi(s), \tag{1}$$

where s' is the next state, V_ϕ is a neural network parameterized by ϕ. The target of the V_ϕ is to approximate the value function of π. The optimization objective of V_ϕ is:

$$\min_\phi \ [\hat{R}(s,a) - V_\phi(s)]^2, \tag{2}$$

The value of Eq. (2) is related to the value network of the generator and the cumulative discounted rewards $\hat{R}(s,a)$, which is provided by the discriminator in GAIL. We try to eliminate the gradient vanishing phenomenon by optimizing Eq. (2) using both the generator and discriminator:

$$\max_w \min_\phi \ \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi} [\hat{R}_w(s,a) - V_\phi(s)]^2, \tag{3}$$

where $\hat{R}_w(s,a) = -\sum_{t=l}^{T} \gamma^{t-l}(\log(1 - D_w(s_t,a_t)))$ denotes the cumulative discounted reward given by the discriminator parameterized by w. Compared with Eq. (2), a maximization operator over the parameter of discriminator is added. Such an operator will keep the variance of the advantage function from reaching a small value, and thus keep a part of advantage from falling to low value.

To show the connection between the maximization operator and the maximization of the advantage variance, we decompose the loss of the value function to the summation of the variance of the advantage function and estimation bias of V_ϕ:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi} \left[(\hat{R}(s,a) - V_\phi(s))^2\right] \\
&= \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi} (V^\pi(s) - V_\phi(s))^2 + \sum_{t=0}^{\infty} \gamma^{2t} \frac{|\mathcal{D}_t^\pi|}{|\mathcal{D}^\pi|} \mathop{\mathrm{Var}}_{s,a\in\mathcal{D}_t^\pi} \left[A^\pi(s,a)\right] \\
&\geq \sum_{t=0}^{\infty} \gamma^{2t} \frac{|\mathcal{D}_t^\pi|}{|\mathcal{D}^\pi|} \mathop{\mathrm{Var}}_{s,a\in\mathcal{D}_t^\pi} \left[A^\pi(s,a)\right],
\end{aligned} \tag{4}$$

where Var represents the variance, $\mathcal{D}^\pi$ denotes the total data collected by π, $\mathcal{D}_t^\pi$ denotes the data collected starting at time step t (e.g., $|\mathcal{D}_0^\pi|$ denotes the number of data collected from the initial states). The equality in Eq. (4) holds iff $\forall s \in \mathcal{D}^\pi, V^\pi(s) = V_\phi(s)$. From Eq. (4), we can conclude that the loss of the value function consists of two parts: (i) the difference between the value function of current policy and the value network; (ii) the variance of the advantage function. Part (ii) is not related to the value network. Thus, the minimization operator w.r.t. the parameter of the value network does not influence the advantage variance at all. As a result, we can get the advantage variance in a such way: find a $\phi^\star$ to minimize the value loss and regard the minimum value loss as the advantage variance. Ideally, $\phi^\star$ can guarantee the discrepancy between V^π and V_ϕ to a small level. That is:

$$\begin{aligned}
\min_\phi \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V_\phi(s))^2\right] &= \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V_{\phi^\star}(s))^2\right] \\
&= \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}(V^\pi(s)-V_{\phi^\star}(s))^2 + \sum_{t=0}^{\infty}\gamma^{2t}\frac{|\mathcal{D}_t^\pi|}{|\mathcal{D}^\pi|}\mathop{\mathrm{Var}}_{s,a\in\mathcal{D}_t^\pi}\left[A^\pi(s,a)\right] \\
&\approx \sum_{t=0}^{\infty}\gamma^{2t}\frac{|\mathcal{D}_t^\pi|}{|\mathcal{D}^\pi|}\mathop{\mathrm{Var}}_{s,a\in\mathcal{D}_t^\pi}\left[A^\pi(s,a)\right].
\end{aligned} \tag{5}$$

Thus, we (approximately) obtain the advantage variance by seeking an (sub-)optimal value network. In order to keep the discriminator from providing rewards that leads to policy gradient vanishing, we optimize the discriminator to maximize the advantage variance:

$$\begin{aligned}
&\max_w \sum_{t=0}^{\infty}\gamma^{2t}\frac{|\mathcal{D}_t^\pi|}{|\mathcal{D}^\pi|}\mathop{\mathrm{Var}}_{s,a\in\mathcal{D}_t^\pi}\left[A^\pi(s,a)\right] \\
&\approx \max_w \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V_{\phi^\star}(s))^2\right] \\
&= \max_w \min_\phi \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}\left[\hat{R}_w(s,a)-V_\phi(s)\right]^2,
\end{aligned} \tag{6}$$

which is exactly Eq. (4).

In the original GAIL optimization objective of discriminator, there are no constraint about the generator's virtual rewards. Such an optimization goal only focuses on telling the difference between agents and experts, with no attention to the inner reward distribution. We will try to solve this issue through the relationship between discriminator loss, reward, and RL advantage functions, using the results in Sect. 4.2.

4.3 Discriminator reward variance regularization algorithm

We define the difference between the real reward and the value function estimation as:

$$\begin{aligned}
f(w,\phi) &= \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}[\hat{R}_w(s,a)-V_\phi(s)]^2, \\
g(w) &= \arg\min_\phi f(w,\phi) = \phi^\star,
\end{aligned} \tag{7}$$

where f is the value loss, $g(w)$ is the optimum parameter of the value function given a discriminator D_w. Based on Eq. (7), we can get the partial derivative:

$$f_\phi(w, g(w)) = \left[\frac{\partial f(w,\phi)}{\partial \phi}\right]_{\phi=g(w)} = 0. \tag{8}$$

By examing the gradients of $f(w, g(w))$, we can find that,

$$\nabla_w f(w, g(w)) = f_\phi(w, g(w))\nabla_w g(w) + f_w(w, g(w)) = f_w(w, g(w)), \tag{9}$$

where ∇_w denotes the total derivative, f_w is the partial derivative of $f(w, \phi)$ w.r.t. w: $f_w(w, g(w)) = \left[\frac{\partial f(w,\phi)}{\partial w}\right]_{\phi=g(w)}$. Equation (9) tells us that, although $\phi^\star$ and w are highly related, we can ignore the gradient of $\phi^\star$ w.r.t. w when calculating $\nabla_w f(w, g(w))$. This characteristic makes this regularization practical, because $\nabla_w \phi^\star$ is hard to be computed. We now have the way to optimize the generator and discriminator iteratively:

First, we train the RL agent to get the value network converged, thus getting $\phi^\star = g(w)$;

Then we can get the gradients of $f(w, g(w))$ with respect to w: $f_w(w, g(w))$;

By combining the Discriminator Reward Variance Loss with the GAIL loss, we can obtain our final regularized optimization objective:

$$\begin{aligned}\max_w \; & \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}[\lambda(\hat{R}_w(s,a) - V_{\phi^*}(s))^2 + \log(D_w(s,a))] \\ & + \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^E}[\log(1 - D_w(s,a))],\end{aligned} \tag{10}$$

where λ is a regularization factor.

The training procedure of GAIL-VR is summarized in the Algorithm 1. Note that such an algorithm is widely applicable in almost all actor-critic RL algorithms with value functions, including A3C (Mnih et al. 2016), TRPO, as well as PPO (Schulman et al. 2017).

Algorithm 1 GAIL-VR

Input: Expert trajectory $\mathcal{D}^E$, policy π_θ, value function V_ϕ, discriminator D_w parameterized by θ_0, ϕ_0, w_0, regularization coefficient λ

for $i = 0, 1, 2, \ldots$ **do**

Sample $\mathcal{D}^\pi$ using policy π_{θ_i}

Compute the cumulative discounted reward $\hat{R}_{w_i}(s,a)$, $\forall s, a \in \mathcal{D}^\pi$

Use ϕ_i as initial value, compute ϕ^*:

$$\phi^* = \arg\min_\phi \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}[\hat{R}_{w_i}(s,a) - V_\phi(s)]^2$$

Update w_i by optimizing:

$$\begin{aligned}& \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^\pi}[\lambda(\hat{R}_w(s,a) - V_{\phi^*}(s))^2 + \log(D_w(s,a))] \\ & + \mathop{\mathbb{E}}_{s,a\in\mathcal{D}^E}[\log(1 - D_w(s,a))]\end{aligned}$$

Update the policy with discriminator reward $\log(1 - D_{w_{i+1}}(s,a))$

end for

Output: Generator parameter θ, discriminator parameter w

5 Empirical study

5.1 Experiment settings

We choose three kinds of mainstream Reinforcement Learning test environments: MuJoCo, Atari, and PyBullet.

We use four contenders in all the experiments. The first is GAIL, without any modifications or tricks. Then we implement Wasserstein Adversarial Imitation Learning (W-GAIL), GAIL with gradient penalty (GAIL-GP; Gulrajani et al. 2017), and Model-based GAIL (MGAIL; Baram et al. 2017) as contenders. Note that we do not take the results of the original GAIL paper, since the paper is testing GAIL on the MuJoCo-v0 environments, but these environments are outdated, and recent research mostly uses MuJoCo-v2 as test environments. However, the original paper code no longer supports v2 environments since it's based on Python2. As a result, we instead collect our own PPO expert trajectories on v2 environments and implemented the GAIL on our own. We have tested our implementation of GAIL on v0 environments and got similar results to the original paper, which verifies our implementation. However, due to differences in expert trajectory data, environmental characteristics, convergence failure in some random seeds and other factors, the result on MuJoCo-v2 which is shown in the next section may be a bit different, and we promise that extensive hyper-parameter searches has been conducted and this is the best result within our reach.

The computing infrastructure and parameter settings are presented in Appendix B. The common hyper-parameters (e.g., network architecture, learning rate, seeds per environment, etc.) of each method are the same and without extra fine-tuning. However, for original GAIL, the best hyper-parameter exceeds the common search range of hyper-parameters for HalfCheetah-v2 and Walker2d-v2. In line with the principle of optimizing the control group's results as much as possible, extra fin-tuning of GAIL results on these two environments are conducted. We thus provide the hyper-parameter search range and the final hyper-parameters in the appendix.

5.2 Results on algorithm performance

Figure 3 demonstrates the test results on MuJoCo environments. GAIL-VR obtains both better final performance (rated as an average reward) and training speed than counterparts in most of the environments. We listed four typical drawings with average reward in initial training processes (usually around 250–500 training steps in MuJoCo environments) and variance.

We have also tested rewards obtained in various games, compared with GAIL and W-GAIL. The comparison result is listed in Table 1.

The results show that GAIL-VR has a universally positive effect on most of the typical RL environments. We can see that it works both in low or high dimensional state/action spaces and compatible with either state data input or raw game image input. Such a performance improvement implies the excellent application potential of GAIL-VR. Besides, GAIL-VR only depends on an actor-critic RL algorithm and a generator-discriminator framework and does not require a specific environment. The problem GAIL-VR hopes to tackle is also universal in imitation learning with this framework. What's more, the hyper-parameters used in the experiments are set to the parameter we commonly used, instead of fine-tuning for each environment. This also indicates that GAIL-VR possesses high stability w.r.t. the hyper-parameter, making GAIL-VR a parameter-tuning-free method.

Table 1 Rewards obtained in each game using GAIL-VR and GAIL without modifications. The first three rows show the state and action space and the number of expert trajectories used; the bottom five rows show the rewards obtained by GAIL-VR and its counterparts.

	Pong	MsPacman	SpaceInvaders	Seaquest	Reacher	HalfCheetah	Hopper	Walker2d	Humanoid-BulletEnv
State Space	100,800	100,800	100,800	100,800	111	17	11	17	6
Action Space	6	9	6	18	8	6	3	6	6
Expert Trajectories	20	20	20	20	4	4	4	4	10
GAIL	−9.2	1109	327	970	−27.4	4012	2589	4818	−173.4
GAIL-GP	−17.65	1210	356	1361	−26.61	4323	3627	5968	−252.1
W-GAIL	−18.3	1208	**401**	551	−30.65	4654	3613	4805	−206.0
MGAIL	−7.0	1007	257	1279	−45.2	3598	**3635**	5951	−242.7
GAIL-VR	**−6.69**	**1331**	399	**1560**	**−21.83**	**4781**	3619	**6096**	**−117.5**

Bolded value indicates highest return among the algorithms in each environment.

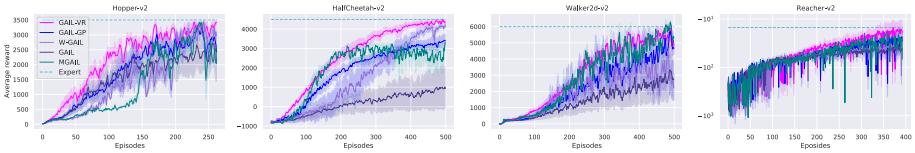


Fig. 3 Reward curves of GAIL-VR and contender methods in MuJoCo environments. Shaded region indicates the standard deviation

5.3 Results on stability

Stability on less data GAIL-VR is found to perform well with little expert data. From the experiment result of ours and Ho and Ermon (2016), GAIL and its variants perform well with abundant expert data. However, when the data is not sufficient, GAIL and its variants may not perform well. Figure 4 shows the test results in insufficient data. Our method outperforms most of the GAIL variants significantly.

The performance improvement of GAIL-VR may result from improved data efficiency by highlighting the importance of critical expert action data. For each expert trajectory, there will always be some state-action pairs that are critical to learning the whole policy. However, in previous works using the generator-discriminator framework, even if agents were able to take actions that are very similar to expert actions, some minor flaws may cause the discriminator to appear shortcut during training and recognize the action features of the agent, resulting in the inability to give a higher reward. But in our algorithm, we strengthen the identification of the key data regions (where the advantage function variance is larger, indicating that there is a more critical policy improvement) so that the

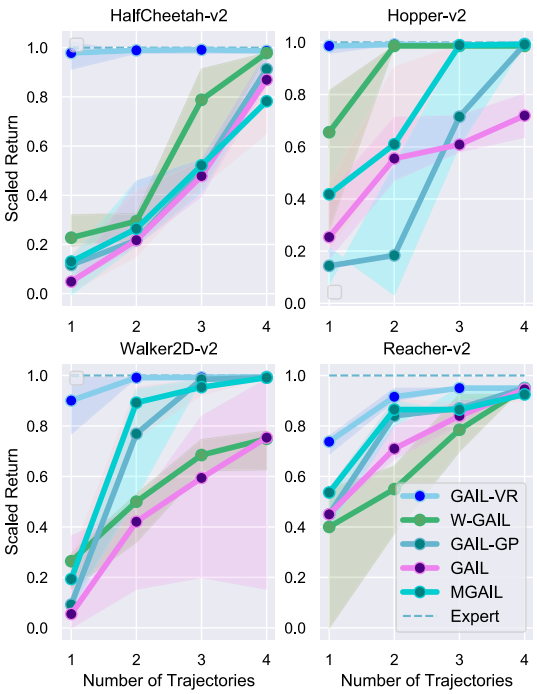


Fig. 4 Performances of GAIL-VR in 4 MuJoCo environments with comparison to other methods. Performance is evaluated by average return and scaled

discriminator can enhance the reward variance of these regions to avoid premature occurrence of the discriminator failure that gives all the results of low reward.

Stability of long training periods It is quite common that, after training for a long time, the achieved reward decreases from the highest performances. However, in long-training scenarios, our method outperforms GAIL in stopping such a decrease. Table 2 shows that GAIL-VR benefits from preventing reward drop after long training processes, and even recorded cases where reward is still rising during long-term training, such as Reacher-v2. Note that we compress the counterpart results into one column because there is no significant statistical difference between different algorithms in performance decline happen time and decline range. We believe that such a performance boost can be explained by the prevention of "low reward trap", mentioned in Sect. 4. By keeping the mean value and maximizing the variance, the discriminator is forced to give higher rewards to state-action pairs closer to experts, keeping the gradient alive in the complete training process. In addition, the lowered reward will be more focused on the bad transitions, making the distribution of value function more ideal.

5.4 Results on our motivations

Our algorithm shifts the distribution of rewards to a better one. Figure 5 shows the given rewards distribution between GAIL and GAIL-VR. Our algorithm makes the median reward much higher and strengthens the variance, proving that GAIL-VR really works under our assumptions.

Also, GAIL-VR successfully reversed the trend that Discriminator Loss (D-loss) declined and then stopped rising. Figure 6 (Left) shows a typical set of D-loss training data. Except for using different algorithms, the other parameters of the two groups of training are kept the same. From the figure, our algorithm successfully makes the D-loss rise, which shows that we have succeeded in preventing the disappearance of the gradient.

In addition, we have done test on the gradient of generator. Figure 6 (Right) shows a typical training on HalfCheetah environment under the same parameter setting. GAIL-VR can prevent quick gradient vanishing of the generator. Note that the figure is drawn in logarithmic coordinates, so the gradient enhancement is significant.

Table 2 Rewards ± standard deviation obtained through long training periods.

	GAIL and variants	GAIL-VR
Pong	-19.0 ± 3.6	$\mathbf{-9.2 \pm 1.4}$
MsPacman	888 ± 57	$\mathbf{997 \pm 39}$
SpaceInvaders	352 ± 10	$\mathbf{399 \pm 14}$
Seaquest	1288 ± 32	$\mathbf{1466 \pm 39}$
Reacher	-27.72 ± 5.71	$\mathbf{-10.01 \pm 3.05}$
HalfCheetah	3703 ± 576	$\mathbf{4471 \pm 268}$
Hopper	3086 ± 527	$\mathbf{3609 \pm 355}$
Walker2d	3944 ± 992	$\mathbf{4592 \pm 625}$
Humanoid-Bullet	-252.1 ± 25.7	$\mathbf{-127.1 \pm 17.6}$

Bolded value indicates higher return in each environment.

The training length is at least two times the length needed to achieve the best reward. GAIL and its variances behaves the same in this experiment

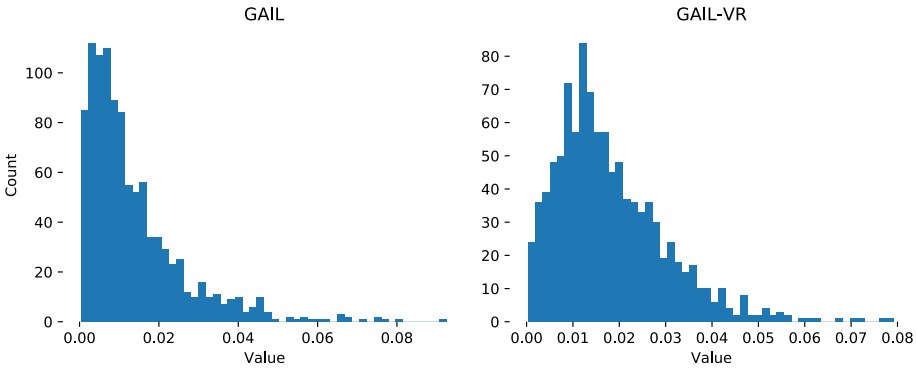


Fig. 5 Statistical frequency histogram of rewards given by the generator. The data is collected in Mujoco Hopper environment at timestep 1700. More data is available in the supporting materials

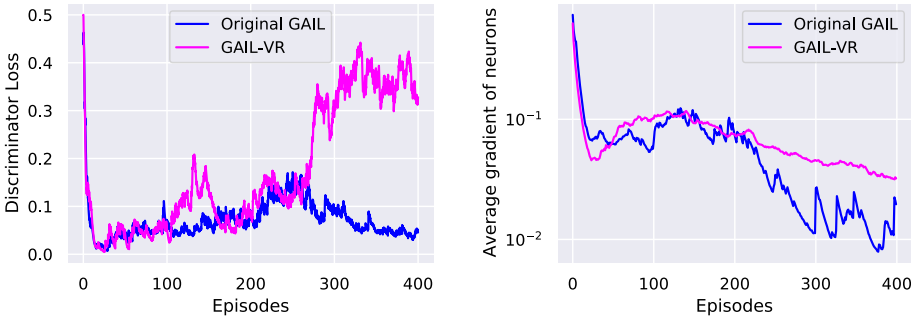


Fig. 6 Left: A typical discriminator loss trend comparison during training. Right: Average gradient comparison during training. The datasets and parameters are kept the same

6 Conclusion

In this paper, we introduced GAIL-VR, a simple method to improve the convergence ability of GAIL, especially in complex environments. The principle behind GAIL-VR is to alter the role of discriminator from competitor to teacher by optimizing the discriminator to maximize the variance of the advantage functions. GAIL-VR can be applied to any actor-critic algorithm under the generator-discriminator framework. Empirical studies show that GAIL-VR can improve the convergence, boost performance as well as help stabilize the training. We also provide a new sight to the research of the generator-discriminator frameworks. Our study outlines that the balance between generator and discriminator can be easily broken, especially in complex environments, and provided an efficient way to tackle this issue.

Appendix A: Proof of Equations 4 and 8

A.1 Proof of Equation 4

The value loss:

$$
\begin{aligned}
&\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V(s))^2\right]\\
&\quad=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V(s)-V^\pi(s)+V^\pi(s))^2\right]\\
&\quad=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V^\pi(s)+V^\pi(s)-V(s))^2\right]\\
&\quad=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V^\pi(s))^2+(V^\pi(s)-V(s))^2\right.\\
&\qquad\left.+2(\hat{R}(s,a)-V^\pi(s))(V^\pi(s)-V(s))\right].
\end{aligned}
\tag{11}
$$

Here $\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V^\pi(s))^2\right]$ can be written as:

$$
\begin{aligned}
\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V^\pi(s))^2\right]
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V^\pi(s)-Q^\pi(s,a)+Q^\pi(s,a))^2\right]\\
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a)+Q^\pi(s,a)-V^\pi(s))^2\right]\\
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a))^2+(Q^\pi(s,a)-V^\pi(s))^2\right.\\
&\quad\left.+2(\hat{R}(s,a)-Q^\pi(s,a))(Q^\pi(s,a)-V^\pi(s))\right]\\
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a))^2+(A^\pi(s,a))^2\right.\\
&\quad\left.+2(\hat{R}(s,a)-Q^\pi(s,a))(Q^\pi(s,a)-V^\pi(s))\right]\\
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a))^2\right]\\
&\quad+\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(A^\pi(s,a))^2\right]\\
&\quad+\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[2(\hat{R}(s,a)-Q^\pi(s,a))(Q^\pi(s,a)-V^\pi(s))\right]\\
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a))^2\right]\\
&\quad+\mathrm{Var}_{s,a\in\mathcal{D}^\pi_t}A^\pi(s,a)\\
&\quad+\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[2(\hat{R}(s,a)-Q^\pi(s,a))(Q^\pi(s,a)-V^\pi(s))\right].
\end{aligned}
\tag{12}
$$

Here,

$$
\begin{aligned}
\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a))^2\right]
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left\{(r(s,a)+\gamma\hat{R}(s',a')-r(s,a)-\gamma\mathbb{E}_{S'}\left[V^\pi(S')\right])^2\right\}\\
&=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left\{(\gamma\hat{R}(s',a')-\gamma\mathbb{E}_{S'}\left[V^\pi(S')\right])^2\right\}\\
&=\gamma^2\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left\{(\hat{R}(s',a')-\mathbb{E}_{S'}\left[V^\pi(S')\right])^2\right\}
\end{aligned}
\tag{13}
$$

Suppose the environment is deterministic, $S'=s'$:

$$
\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a))^2\right]=\gamma^2\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s',a')-V^\pi(s'))^2\right].
\tag{14}
$$

Let $\mathcal{D}^\pi_n$ denotes the data collected by π that removes the first n steps. For example, $\mathcal{D}^\pi_0=\mathcal{D}^\pi$; $\mathcal{D}^\pi_1$ does not contain any initial state.

$$
\begin{aligned}
\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-Q^\pi(s,a))^2\right]
&=\gamma^2\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s',a')-V^\pi(s'))^2\right]\\
&=\gamma^2\frac{|\mathcal{D}^\pi_1|}{|\mathcal{D}^\pi|}\mathbb{E}_{s,a\in\mathcal{D}^\pi_1}\left[(\hat{R}(s,a)-V^\pi(s))^2\right]
\end{aligned}
\tag{15}
$$

Thus,

$$\begin{aligned}\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V^\pi(s))^2\right] &= \gamma^2\frac{|\mathcal{D}_1^\pi|}{|\mathcal{D}^\pi|}\mathbb{E}_{s,a\in\mathcal{D}_1^\pi}\left[(\hat{R}(s,a)-V^\pi(s))^2\right]\\ &\quad+\mathrm{Var}_{s,a\in\mathcal{D}^\pi}A^\pi(s,a)\\ &\quad+\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[2(\hat{R}(s,a)-Q^\pi(s,a))(Q^\pi(s,a)-V^\pi(s))\right]\\ &=\sum_{t=0}\gamma^{2t}\left\{\frac{|\mathcal{D}_t^\pi|}{|D^\pi|}\mathrm{Var}_{s,a\in\mathcal{D}_t^\pi}A^\pi(s,a)\right.\\ &\quad\left.+\mathbb{E}_{s,a\in\mathcal{D}_t^\pi}\left[2(\hat{R}(s,a)-Q^\pi(s,a))(Q^\pi(s,a)-V^\pi(s))\right]\right\},\end{aligned} \tag{16}$$

where the second equality is a recursive expansion of the RHS of the first equality.

Consequently, in a deterministic environment, we can factorize the value loss:

$$\begin{aligned}&\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V(s))^2\right]\\ &\quad=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(V^\pi(s)-V(s))^2\right]\\ &\qquad+2\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V^\pi(s))(V^\pi(s)-V(s))\right]\\ &\qquad+\sum_{t=0}\gamma^{2t}\left\{\frac{|\mathcal{D}_t^\pi|}{|D^\pi|}\mathrm{Var}_{s,a\in\mathcal{D}_t^\pi}A^\pi(s,a)\right.\\ &\qquad\left.+\mathbb{E}_{s,a\in\mathcal{D}_t^\pi}\left[2(\hat{R}(s,a)-Q^\pi(s,a))(Q^\pi(s,a)-V^\pi(s))\right]\right\}.\end{aligned} \tag{17}$$

Because

$$\begin{aligned}Q^\pi(s,a)&=\mathbb{E}[\hat{R}(S,A)|S=s,A=a],\\ V^\pi(s)&=\mathbb{E}[\hat{R}(S,A)|S=s]=\mathbb{E}[Q(S,A)|S=s].\end{aligned}$$

We can omit the cross term in Eq. 17:

$$\begin{aligned}&\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(\hat{R}(s,a)-V(s))^2\right]\\ &\quad=\mathbb{E}_{s,a\in\mathcal{D}^\pi}\left[(V^\pi(s)-V(s))^2\right]+\sum_{t=0}\gamma^{2t}\frac{|\mathcal{D}_t^\pi|}{|D^\pi|}\mathrm{Var}_{s,a\in\mathcal{D}_t^\pi}\left[A^\pi(s,a)\right].\end{aligned} \tag{18}$$

A.2 Proof of Equation 8

The proof of Eq. 8 in original paper is as follows:

$$\frac{df(w,g(w))}{dw}=\left[\frac{dw}{dw}\frac{\partial f(w,\phi)}{\partial w}+\frac{dg(w)}{dw}\frac{\partial f(w,\phi)}{\partial\phi}\right] \tag{19}$$

$$=\left[\frac{\partial f(w,\phi)}{\partial w}\right]_{\phi=g(w)}+\frac{dg(w)}{dw}\left[\frac{\partial f(w,\phi)}{\partial w}\right] \tag{20}$$

$$= [\frac{\partial f(w,\phi)}{\partial w}]_{\phi=g(w)} + \frac{dg(w)}{dw} \cdot 0 \tag{21}$$

$$= [\frac{\partial f(w,\phi)}{\partial w}] \tag{22}$$

$$= f_w(w, g(w)) \tag{23}$$

Appendix B: Hyperparameter settings and computing infrastructure

Table 3 shows the paramters shared between the generator and discriminator for GAIL/GAIL-VR in the experiment section. The additional parameters for the PPO generator is:

PPO clip parameter: 0.2
Reward discount: 0.99; GAE lambda parameter: 0.95
RMSprop optimizer epsilon: 1e−5; RMSprop optimizer alpha: 0.99
Entropy coefficient: 0.5; Value loss coefficient: 0.5

However, the best results of GAIL on HalfCheetah-v2 and Walker2d-v2 exceeds the learning rate search range ([3e−6,5e−4]) and the reward discount search range ([0.9,0.99]). So for GAIL on HalfCheetah-v2, the learning rate is 1e−3 and the reward discount is 0.9, and for GAIL on Walker2d-v2 the learning rate is 1e−6 and the reward discount is 0.995. The λ in our added loss is 0.1, we have tested various settings and found that λ from 0.01 to 0.43 can benefit the original GAIL algorithm.

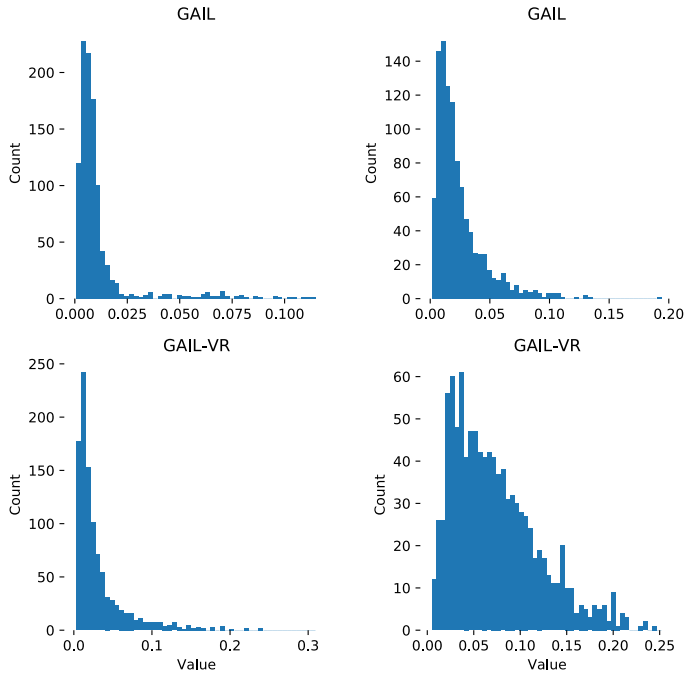
All of the experiments are conducted on a machine with 2 Intel Xeon 4110 CPU (16 cores, 32 threads in total), 128GB memory and 2 NVIDIA RTX2080Ti GPUs. The operation system is Ubuntu 16.04 with Python version of 3.7, TensorFlow version of 1.14 and PyTorch version of 1.0.

Appendix C: Additional comparison between GAIL and GAIL-VR concerning the reward distribution

As shown in Fig. 3 in our paper, GAIL-VR can give a better reward distribution. Here we provide more experiments in this perspective (Fig. 7).

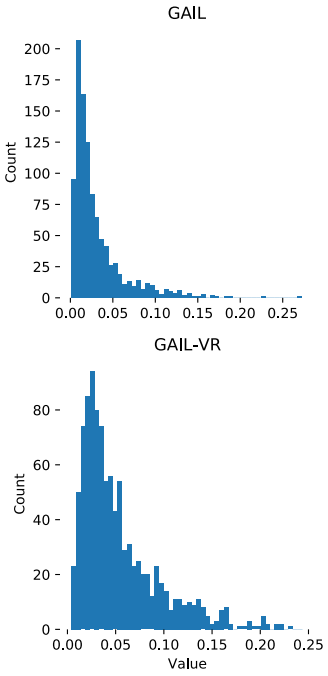
Table 3 Shared hyperparameter settings for GAIL/GAIL-VR results in the experiment section

	Generator	Discriminator
Learning rate	3e−4	
Epoch per iteration	4	5
Batch size	32	128
Network layers	3	
Layer type	Fully connected	
Seeds per env.	10	

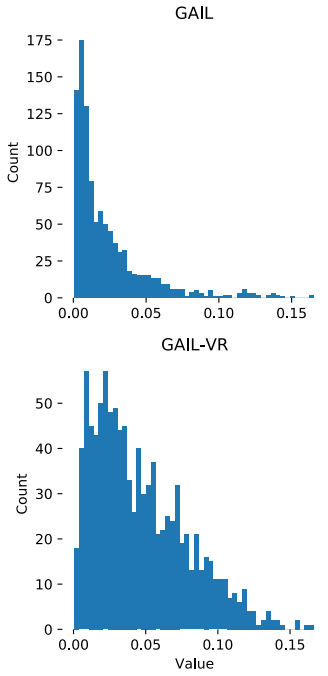


(a) Hopper:short training

(b) Halfcheetah



(c) Humanoid



(d) Walker2D

Fig. 7 Test results concerning reward distribution

Acknowledgements This work is supported by national key research and development program of China (2020aa0107200), and NSFC (61876077). We thank Xin-Qiang Cai, Yao-Xiang Ding and Jing-cheng Pang for providing very useful information about imitation learning in Atari environments. Xiong-hui Chen and Yu-ren Liu gave a lot of advice for the writing of papers for journals, we thank them for their help. We also thank Tian Xu and Zi-niu Li for their kind advice with regard to imitation learning theory.

Authors' contributions The first author completed the overall design of the algorithm, carried out experiments, and wrote the paper. The second author provided support in theoretical derivation and played an important part in writing the paper. The third author provided propositional guidance, and provided assistance throughout the process.

Funding This work is supported by National Key Research and Development Program of China (2020AAA0107200), and NSFC (61876077).

Availability of data and material The datasets used in experiments are all free-of-use. We provided the data sources in the references of the paper.

Code availability As this code has been used by the partners of the lab, we are currently communicating with them, and we will upload the code upon agreement.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable. All the experiments in this paper are computer simulations of games and do not involve experiments on animals, plants, or human entities.

Consent to participate Not applicable. All the experiments in this paper are computer simulations of games and do not involve experiments on animals, plants, or human entities.

Consent for publication Not applicable. The paper does not include data or images that require permissions to be published.

References

Abbeel, P., & Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *ICML 2004*, Alberta, Canada.

Baram, N., Anschel, O., Caspi, I., & Mannor, S. (2017). End-to-end differentiable adversarial imitation learning. In *ICML 2017*, Sydney, Australia (pp. 390–399).

Bhattacharyya, R. P., Phillips, D. J., Wulfe, B., Morton, J., Kuefler, A., & Kochenderfer, M. J. (2018). Multi-agent imitation learning for driving simulation. In *IROS 2018*, Madrid, Spain (pp. 1534–1539).

Chen, M., Wang, Y., Liu, T., Yang, Z., Li, X., Wang, Z., & Zhao, T. (2020). On computation and generalization of generative adversarial imitation learning. CoRR arXiv:2001.02792.

Finn, C., Christiano, P. F., Abbeel, P., & Levine, S. (2016a). A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. CoRR arXiv:1611.03852.

Finn, C., Levine, S., & Abbeel, P. (2016b). Guided cost learning: Deep inverse optimal control via policy optimization. In *ICML 2016*, New York City, NY (pp. 49–58).

Fu, J., Luo, K., & Levine, S. (2017). Learning robust rewards with adversarial inverse reinforcement learning. CoRR arXiv:1710.11248.

Geng, S., Nassif, H., Manzanares, C. A., Reppen, A. M., & Sircar, R. (2020). Identifying reward functions using anchor actions. CoRR arXiv:2007.07443.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of Wasserstein Gans. In *Advances in neural information processing systems*, Long Beach, CA (Vol. 30, pp. 5767–5777).

Heess, N., Wayne, G., Silver, D., Lillicrap, T. P., Erez, T., & Tassa, Y. (2015). Learning continuous control policies by stochastic value gradients. In *Advances in neural information processing systems*, Quebec, Canada (Vol. 28, pp. 2944–2952).

Ho, J., & Ermon, S. (2016). Generative adversarial imitation learning. In *Advances in neural information processing systems*, Barcelona, Spain (Vol. 29, pp. 4565–4573).

Kostrikov, I., Agrawal, K. K., Dwibedi, D., Levine, S., & Tompson, J. (2019). Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *ICLR 2019*, New Orleans, LA.

Li, Z., Kiseleva, J., & Rijke, Md. (2019). Dialogue generation: From imitation learning to inverse reinforcement learning. *AAAI 2019*, Honolulu, HI (pp. 6722–6729).

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *ICML 2016*, New York City, NY (pp. 1928–1937).

Ng, A. Y., & Russell, S. J. (2000). Algorithms for inverse reinforcement learning. In *ICML 2000*, Stanford, CA (pp. 663–670).

Peng, X. B., Kanazawa, A., Toyer, S., Abbeel, P., & Levine, S. (2019). Variational discriminator bottleneck: Improving imitation learning, inverse RL, and GANs by constraining information flow. In *ICLR 2019*, New Orleans, LA.

Ross, S., & Bagnell, D. (2010). Efficient reductions for imitation learning. In *AISTATS 2010*, Sardinia, Italy (pp. 661–668).

Ross, S., Gordon, G. J., & Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS 2011*, Fort Lauderdale, FL (pp. 627–635).

Russell, S. J. (1998). Learning agents for uncertain environments (extended abstract). In *COLT 1998*, Madison, WI (pp. 101–103).

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., & Moritz, P. (2015). Trust region policy optimization. In *ICML 2015*, Lille, France (pp. 1889–1897).

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. CoRR arXiv:1707.06347.

Shi, J. C., Yu, Y., Da, Q., Chen, S. Y., & Zeng, A. (2019). Virtual-taobao: virtualizing real-world online retail environment for reinforcement learning. In *AAAI 2019*, Honolulu, HI, January 27–February 1, 2019 (pp. 4902–4909). AAAI Press.

Slonim, N., & Tishby, N. (2000). Document clustering using word clusters via the information bottleneck method. In *ACM SIGIR 2000*, Greece, Athens (pp. 208–215).

Song, J., Ren, H., Sadigh, D., & Ermon, S. (2018). Multi-agent generative adversarial imitation learning. In *Advances in neural information processing systems*, Montréal, Canada (Vol. 31, pp. 7461–7472).

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT Press.

Wu, A., Piergiovanni, A. J., & Ryoo, M. S. (2018). Action-conditioned convolutional future regression models for robot imitation learning. In *CVPR 2018 workshops*, Salt Lake City, UT (pp. 2035–2037).

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.