Robust Adversarial Imitation Learning via Adaptively-Selected Demonstrations

Yunke Wang[1] , **Chang Xu**[2*] and **Bo Du**[1*]
[1] National Engineering Research Center for Multimedia Software, Institute of Artificial Intelligence, School of Computer Science and Hubei Key Laboratory of Multimedia and Network Communication Engineering, Wuhan University, China
[2] School of Computer Science, Faculty of Engineering, The University of Sydney, Australia
{yunke.wang, dubo}@whu.edu.cn, c.xu@sydney.edu.au

Abstract

The agent in imitation learning (IL) is expected to mimic the behavior of the expert, its performance relies highly on the quality of given expert demonstrations. However, in real-world tasks, the assumption of collected demonstrations are optimal cannot always hold, which would seriously influence the ability of the learned agent. In this paper, we propose a general method within the framework of Generative Adversarial Imitation Learning (GAIL) to address imperfect demonstration issue, in which good demonstrations can be adaptively selected for training while bad demonstrations are abandoned. Specifically, a binary weight is assigned to each expert demonstration to indicate whether to select it for training. The weight is set to be determined by the reward function in GAIL (i.e. higher reward results in higher weight). Different from some existing solutions that require some prior about this weight, we set up the connection between weight and model so that we can jointly optimize GAIL and learn the latent weight. Besides hard binary weighting, we also propose a soft weighting scheme. Experiments on Mujoco demonstrate the great performance of our proposed method over other GAIL-based methods when dealing with imperfect demonstrations.

1 Introduction

Imitation learning (IL) [Hussein *et al.*, 2017] , which aims to let the agent imitate the behavior of the expert, has achieved impressive performance in many applications [Englert *et al.*, 2013; Codevilla *et al.*, 2018]. Compared to the intractable reward engineering [Amodei *et al.*, 2016] in reinforcement learning (RL) [Sutton and Barto, 2018], IL provides a much easier way to directly infer a reward function from expert demonstrations, in such a way that the agent can greatly improve on the demonstrated behavior.

Generative adversarial imitation learning (GAIL) [Ho and Ermon, 2016] is one of the state-of-the-art IL methods. Based on the framework of GAN [Goodfellow *et al.*, 2014], GAIL regards imitation learning as a distribution matching problem between the state-action distribution of expert policy and that of the agent policy. Many variants of GAIL have been proposed in recent years to further improve GAIL's performance from different aspects. Notice that GAIL does not directly recover a reward function since the discriminator will output 0.5 for each state-action pair at the saddle point, AIRL [Fu *et al.*, 2017] separates the reward function from the discriminator, and can learn the reward function during the agent training. VAIL [Peng *et al.*, 2018] uses variational information bottleneck (VIB) to constrain the ability of the discriminator since the discriminator may easily overwhelm the policy in GAIL training, which could result in the uninformative reward. So VAIL could improve upon GAIL by stabilizing the GAIL training. InfoGAIL [Li *et al.*, 2017] follows the idea of InfoGAN [Chen *et al.*, 2016], which aims to learn a multimodal policy with latent variable.

GAIL and its variants have shown great success in benchmark tasks with the underlying assumption that expert demonstrations are optimal. However, optimal demonstrations are often hard to acquire in real-world tasks. For example, since the number of most qualified experts is limited, the demonstrations collected in the real world are more likely to be a combination of demonstrations from both qualified experts and inferior experts. Furthermore, a qualified expert would make mistakes due to the limited energy and presence of distraction, resulting in imperfect demonstrations. As stated above, the imperfect demonstrations issue occurs and it could seriously mislead the training procedure of IL.

Existing solutions proposed to address the imperfect demonstrations issue rely heavily on the prior information or assumptions of demonstrations. In 2IWIL and IC-GAIL [Wu *et al.*, 2019], a fraction of demonstrations labeled with confidence score need to be provided first in training. Preference-based inverse reinforcement learning methods such as T-REX [Brown *et al.*, 2019] need ranked trajectories to learn a relevant reward function and then conduct RL with the new learned reward. Multi-modal imitation learning method InfoGAIL can be used to recover all the demonstrators within demonstrations, however a prior about the number of demonstrators is needed first and we also cannot find the best demonstrator before we evaluate all the modal.

To address the weakness of existing solutions, we propose a new approach based on GAIL called Selective Adversarial

[*]Corresponding author

Imitation Learning (SAIL) to solve imperfect demonstrations issue without auxiliary information in this paper. In SAIL, we first present a weighted adversarial imitation learning framework based on the Wasserstein variant of GAIL and a simple binary weight is assigned to each training demonstration to indicate whether to choose it into the training set. The weight itself can be also interpreted as the quality of the demonstration from another perspective. Instead of asking for auxiliary information about the weight such as 2IWIL, SAIL considers weight can be exactly determined by the feedback from GAIL model. Motivated by that the good demonstration is always related to a higher reward while the bad demonstration is always related to a lower reward, we ask the reward function in Wasserstein GAIL to predict weight for each demonstration. After we obtain this relation between weight prediction and model training, both of them can be jointly optimized as a whole. Besides hard binary weighting, we also propose a soft weighting scheme for SAIL. We provide a theoretical analysis to show that SAIL can converge to the saddle point. The experiment result shows the advantage of SAIL compared to baseline methods when dealing with imperfect demonstrations and the idea of SAIL can be easily extended to other GAIL-based methods such as InfoGAIL.

2 Related Work

Several relevant researches to address imperfect demonstration issue are given below, however most of methods require prior information or assumption about demonstrations, which constrains them to be applied to a more universal setting.

Preference learning [Christiano *et al.*, 2017] based IRL [Ziebart *et al.*, 2008] methods have shown to be effective when handling imperfect demonstrations, especially in Atari games. With a set of ranked trajectories, T-REX [Brown *et al.*, 2019] aims to learn a reward function that can well fit the rankings. However, these ranked trajectories may not be available in the standard IL setting. Beyond T-REX, D-REX [Brown *et al.*, 2020] is proposed to acquire ranked trajectories automatically. Specifically, D-REX firstly learns a base policy by behavioral cloning (BC) [Bain and Sammut, 1995] over imperfect demonstrations and then derives noisy policies to generate ranked trajectories. Multi-modal imitation learning methods can be used to handle imperfect demonstrations if imperfect demonstrations are from multiple demonstrators with diverse quality. InfoGAIL [Li *et al.*, 2017] learns a multi-modal policy where each mode of the policy represents a single demonstrator and good policies can be then obtained by selecting good modes of the multi-modal policy. VILD [Tangkaratt *et al.*, 2019] models the distribution of multi-modal demonstrations with the assumption that each demonstrator is formulated by adding different ratios of Gaussian noise to the optimal demonstrator. Thus it may suffer a performance degradation under more universal settings. From the perspective of weighting samples, imperfect demonstrations issue can be solved by assigning proper weight for different demonstrations. With a fraction of labeled demonstrations, SSIRL [Valko *et al.*, 2013] uses semi-supervised support vector machines [Bennett and Demiriz, 1999] to separates good or bad demonstrations when learning a policy, thus improving the performance of the agent. [Wu *et al.*, 2019] introduces weight into GAIL framework by considering a fraction of demonstrations labeled with confidence are available for training. [Wang *et al.*, 2021] also conducts a weighted GAIL framework and learns to weight imperfect demonstrations during training without exposing too much prior.

3 Preliminaries

The framework of reinforcement learning is generally based on the Markov Decision Process (MDP) [Puterman, 1994]. An MDP consists of five elements $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma \rangle$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the transition probability distribution, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1]$ is the discounting factor for future rewards. The return in the MDP is calculated as the discounted sum of rewards obtained by the agent over all episodes, the goal of RL is thus to learn a policy $\pi : \mathcal{S} \to \mathcal{A}$ that can maximize the expected return over all episodes. For any policy π, there is a one-to-one correspondence between the policy and its occupancy measure $\rho_\pi : S \times A \to R$.

Generative Adversarial Imitation Learning. The general framework of Generative Adversarial Networks (GANs) had been applied in the imitation learning problem, which results in GAIL algorithm. In GAIL, the classical imitation learning problem is treated as an occupancy measure matching between the expert policy and the agent policy via Jensen-Shannon Divergence. A discriminator D_ψ is introduced to distinguish expert transitions from agent transitions, while the agent is to "fool" the discriminator into taking agent transitions as those expert transitions. Formally, the objective function of GAIL is written as

$$\min_\theta \max_\psi \; \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_\psi(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_E}}[\log(1 - D_\psi(s,a))], \tag{1}$$

where ρ_{π_θ} and ρ_{π_E} denote the occupancy measures of agent policies π_θ and π_E, respectively.

The plain GAN is known to easily suffer unstable training issues (i.e. difficult to balance the performance of generator and discriminator), and GAIL may inevitably inherit these properties. WGAN [Arjovsky *et al.*, 2017], which minimizes the Wasserstein distance instead of Jensen-Shannon divergence between distributions, is a well-known method to solve unstable training problem in GAN. While only few modifications need to be made from original GAN to WGAN, we can easily extend this idea into GAIL. The objective function of Wasserstein GAIL (WGAIL) [Xiao *et al.*, 2019] can be defined as

$$\min_\theta \max_\varphi \; \mathbb{E}_{(s,a)\sim\rho_{\pi_E}}[r_\varphi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[r_\varphi(s,a)] + \lambda\Psi(r_\varphi), \tag{2}$$

where the critic r_φ is the reward function, $\Psi(r_\varphi) = -\mathbb{E}_{(s,a)\sim\rho_{\hat{\pi}}}(||\nabla r_\varphi(\hat{s},\hat{a})||_2 - 1)^2$ is a regularizer term to satisfy the Lipschitz constraint on reward r_φ and λ is the hyperparameter. When training to convergence, Wasserstein GAIL may learn the expert policy as well as recover the corresponding reward function.

4 Methodology

Despite GAIL has achieved great success in benchmark tasks, its performance relies highly on the quality of expert demonstrations. However in some real-world imitation learning tasks, the number of most qualified experts is often limited, which means more demonstrations might come from the inferior experts. The imperfect demonstration issue therefore occurs in imitation learning, which seriously influences the performance of the learned agent.

A natural idea to exploit these imperfect demonstrations in imitation learning is to weight their influence in the objective function. In particular, we introduce a weight $w(s, a)$ for each expert state-action pair, and then obtain the following weighted Wasserstein GAIL

$$\min_{\theta} \max_{\varphi} \mathbb{E}_{(s,a)\sim\rho_{\pi_E}}[w(s,a)r_\varphi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[r_\varphi(s,a)] + \lambda\Psi(r_\varphi). \tag{3}$$

However Eq. (3) could be intractable for the following reasons. In classical imitation learning, state and action pairs are the only information we collect from experts, while the ground-truth weights of the demonstrations are unknown. We therefore expect an appropriate estimation of the weights, so that they can well indicate the "goodness" of expert demonstrations.

For simplicity, we begin with the analysis on the binary weight, i.e. $w(s, a) = \{0, 1\}$. In particular, we choose a demonstration for training by setting $w(s, a) = 1$; on the other hand, $w(s, a) = 0$ implies that the corresponding demonstration will not be chosen as it does not participate in the calculation in Eq. (3). This extreme case thus provides an intuitive explanation on the use of the weight, i.e., setting the weights of good demonstration to 1 while removing those bad demonstrations via zeroing their weights. By doing this, the negative influence of the imperfect demonstrations could be largely alleviated and the performance of the agent learning can be preserved. However, the remaining question is a criterion to determine whether the weight of a given state-action pair is 1 or 0.

Recall that the aim of reinforcement learning (including imitation learning) is to maximize the reward of the agent. If a state-action pair produces a larger reward, we are then confident in taking it as a good demonstration. Given this fact, we tend to estimate the "goodness" of the expert demonstration by investigating its reward. According to Eq. (3), the reward of the state action pair (s, a) can be calculated with the help of the critic r_φ, i.e. $r_\varphi(s, a)$. Given a threshold K, the demonstration with reward larger than K can be regarded as good demonstrations with weight equal to 1, otherwise the weight can be set to 0, i.e.

$$w^* = \begin{cases} 1, & if \quad r_\varphi(s,a) \geq K, \\ 0, & if \quad r_\varphi(s,a) < K. \end{cases} \tag{4}$$

We find that the optimal weight $w^*(s, a)$ could be dynamically determined by the critic function r_φ in Wasserstein GAIL. Along with the optimization of r_φ, we proceed to rewrite Eq. (4) to the following maximization problem,

$$\max_{w} wr_\varphi(s,a) - Kw, \quad s.t. \quad w_i \in \{0,1\}. \tag{5}$$

Hence, by integrating the objective function of w into the weighted Wasserstein GAIL, we achieve

$$\min_{\theta} \max_{w,\varphi} \mathbb{E}_{(s,a)\sim\rho_{\pi_E}}\left[w(s,a)r_\varphi(s,a) - Kw(s,a)\right] - \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[r_\varphi(s,a)] + \lambda\Psi(r_\varphi), \quad s.t. \quad w_i \in \{0,1\}. \tag{6}$$

The weight learning and Wasserstein GAIL training can thus be optimized alternately. For weight learning, the reward function r_φ in Wasserstein GAIL can act as a critic to weight each state-action pair. The state-action pair of a larger estimated reward will get a larger weight (e.g. $w = 1$) otherwise it will get a smaller weight (e.g. $w = 0$) in iterations. For Wasserstein GAIL training, agent policy π_θ can be optimized with the solved weight variable $w(s, a)$ from the last iteration. K is the threshold to distinguish expert demonstrations. By gradually decreasing K during the optimization, we would lower the standard to accept more expert demonstrations to be good and allow their participation in the optimization of agent policy.

Remark. The algorithm stemming from Eq. (6) is related to self-paced learning, since it will dynamically determine the pace to include "good" demonstrations into the optimization of the agent policy. However compared with the classical self-paced learning [Kumar *et al.*, 2010], we highlight the following novelties of the proposed algorithm.

- Self-paced learning has been widely studied in various problems, including event detection [Jiang *et al.*, 2015], object tracking [Supancic and Ramanan, 2013] and multi-view clustering [Xu *et al.*, 2015]. But to the best of our knowledge, this is the first time to investigate the capability of self-paced learning in dealing with imperfect demonstrations in imitation learning.
- The weight w contributes to setting up a *three-player adversarial game* in Eq. (6), where w and the critic r_φ are optimized to maximize the Wasserstein distance of occupancy measure between agent policy π_θ and expert policy π_E, while the agent policy stood by θ is optimized to minimized the Wasserstein distance. In contrast, model parameters and the weight in classical self-paced learning often lead to a simple minimization problem (e.g. minimizing classification loss), and there is no complex adversarial process.
- At last, the motivations of the weighting scheme are different. Classical self-paced learning tends to progressively include examples whose loss is *smaller* than a threshold, while we expect to use the demonstrations whose reward is *larger* than a threshold.

Soft Weighting. In Eq. (4), the binary weight $w = \{0, 1\}$ simply classifies the demonstrations into two categories, however, for some certain demonstrations it is hard to tell that one is absolute "good" or "bad" demonstration. So the soft weighting scheme can be conducted and weight w can be regarded as the probability of the demonstration to be "good" demonstration. Beyond Eq. (4), we suggest the optimal soft weight as

$$w^* = \frac{1}{1 + e^{K - r_\varphi(s,a)}}. \tag{7}$$

Notice that Eq. (7) can be exactly adapted to the sigmoid function. The soft weight in Eq. (7) is a smooth function of hard weight in Eq. (4). Instead of directly separating the demonstrations into "good" and "bad" by the hard weight, the soft weighting scheme provides a more feasible interpretation to address the probability or likelihood of a demonstration to be "good". So we may have a more robust measurement about the quality of demonstrations, especially for those demonstrations whose reward is around the threshold K.

To further naturally embed the optimization of w into Wasserstein GAIL, we can develop the following objective function as a reformulation of Eq. (7),

$$\max_{w} \quad wr(s,a)+w\log(w^{-1}-1) - \log(1-w) - Kw, \tag{8}$$

whose solution is exactly w^* in Eq. (7). By defining

$$f(w) = w\log(w^{-1}-1) - \log(1-w) - Kw, \tag{9}$$

we can then easily extend Eq. (6) to a soft weight version,

$$\min_{\theta}\max_{w,\varphi} \mathbb{E}_{(s,a)\sim\rho_{\pi_E}}\left[w(s,a)r_\varphi(s,a) + f(w(s,a))\right] - \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}\left[r_\varphi(s,a)\right] + \lambda\Psi(r_\varphi). \tag{10}$$

Optimization. Eq. (6) and (10) can be solved by ACS (Alternative Convex Search) [Bazaraa *et al.*, 2013], which suggests that weight w and GAIL can be alternatively updated.

The threshold K should be declined to include more samples during the training process. Different from [Kumar *et al.*, 2010], which provides a fixed way to update K, we consider using a new dynamic way to update the threshold K to ensure that more demonstrations could be included in during training. Suppose K_j is the threshold in the j_{th} iteration, and we can calculate it as

$$K_j = r_{mid} - \left(\frac{j - N_{pre}}{N}\right)^m (r_{mid} - r_{min}), \tag{11}$$

where r_{mid} and r_{min} are the median and min value within batched demonstration reward estimated by r_φ, m is the hyper-parameter, N_{pre} and N denote the number of pre-training iteration and total iteration. In Eq. (11), the threshold K is always constrained to be in the interval $[r_{min}, r_{mid}]$ and gradually decreases. We can also easily observe that $K_N > r_{min}$ is satisfied. So the idea underlying Eq. (11) is to ensure more demonstrations could be gradually included during training while a few outliers will not be considered eventually. The hyper-parameter m is used to control the speed to include good demonstrations into training. For example, K_i would decrease slowly in the initial stage when m is large, which means it may slowly include more samples during the early training.

To demonstrate the advantage of our proposed dynamic decrease way to update threshold K, we compare it with two naive fixed update way in self-paced learning, i.e., *exponential decrease* $K = K/\mu$, *linear decrease* $K = K - \mu$, the learning curve of three decrease ways is provided in Figure 1.

Convergence Analysis. We also provide theoretical analysis on the convergence of SAIL. We first show that the objective function of SAIL can converge with respect to θ by Theorem

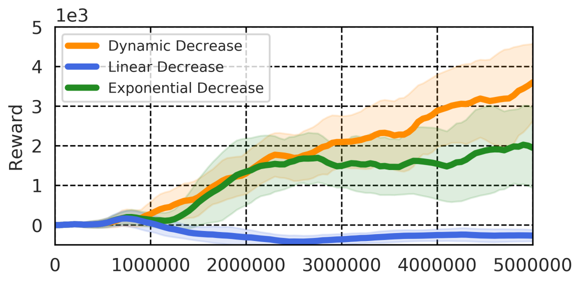


Figure 1: The learning curve on HalfCheetah-v2 with different ways to update threshold k. The x-axis denotes the number of interactions.

1. Then we discuss the convergence of φ and w respectively since they are independent of θ. After showing all the variables can converge to a unique saddle point, we can therefore alternatively update them in each iteration.

Theorem 1. *Rewrite the objective function of SAIL as* $\min_\theta \max_{w,\varphi} \mathcal{J}_{w,\varphi}(\theta)$*, where*

$$\mathcal{J}_{w,\varphi}(\theta) = \mathbb{E}_{(s,a)\sim\rho_{\pi_E}}\left[w(s,a)r_\varphi(s,a) + f(w(s,a))\right] - \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}\left[r_\varphi(s,a)\right] + \lambda\Psi(r_\varphi). \tag{12}$$

Assume that δ_k is the upper bound of the KL-constraint step in TRPO in k_{th} iteration, i.e $D_{KL}(\pi_\theta^k, \pi_\theta^{k+1}) \le \delta_k$ and r_φ is bounded by M, then $\mathcal{J}_{w,\varphi}(\theta)$ can converge with repect to θ.

The proof of Theorem 1 is provided in supplementary material. As stated in Theorem 1, $\mathcal{J}_{w,\varphi}(\theta)$ can converge with respect to θ, which means for any fixed w and φ we have $\mathcal{J}_{w,\varphi}(\theta_k)$ converges uniformly to $\mathcal{J}_{w,\varphi}(\theta^*)$. On the other hand, $\mathcal{J}_{w,\varphi}(\theta)$ is a continuous and concave function of φ, so we have $\varphi_k^* = \arg\sup_\varphi \mathcal{J}_{w,\varphi}(\theta_k)$ also converges to $\varphi^* = \arg\sup_\varphi \mathcal{J}_{w,\varphi}(\theta^*)$. This result implies that there is a saddle point (θ^*, φ^*) in the objective function of SAIL and both θ and r_φ can converge to their optimal solutions alternately by gradient descent. As for weight w, since it is largely determined by the r_φ dynamically, it could also converge as r_φ reaches convergence. Another intuitive explanation on convergence of weight can be stated from self-paced learning. The weight could reach convergence since all the weight is set to 1 at the end of the training.

5 Experiment

In this section, we are going to figure out two questions with the experiment. (1) Does SAIL outperform other comparison methods when dealing with imperfect demonstrations? (2) Can the idea of SAIL be extended to other GAIL-based methods?

5.1 Experiment Setting

We conduct experiments to evaluate our proposed method on four Mujoco [Todorov *et al.*, 2012] continuous control tasks with two kinds of imperfect demonstrations, i.e. suboptimal demonstrations (stage 1) and near-optimal demonstrations (stage 2). Six checkpoints (π_1, π_2, π_3, π_4, π_5, π_6) with increasing quality are used to form stage 1 and 2 demonstrations. Stage 1 demonstrations are formed by trajectories

Method	Stage 1				Stage 2			
	Ant	HalfCheetah	Walker2d	Swimmer	Ant	HalfCheetah	Walker2d	Swimmer
GAIL	36±113	326±172	1623±98	52±3	2610±36	3747±164	2907±91	**54±6**
WGAIL	1210±38	462±71	2170±47	52±6	2638±72	3437±246	2193±198	44±3
SAIL(Hard)	1655±110	3924±415	2244±42	**58±3**	3244±94	4299±89	2717±224	48±3
SAIL(Soft)	**1729±49**	**4035±393**	**2278±99**	54±4	**3289±93**	**4415±146**	**3105±98**	51±3
InfoGAIL	510±82	289±176	1565±123	45±2	2700±69	**4448±78**	2900±108	**48±2**
InfoWGAIL	1142±57	402±69	2032±123	39±4	2771±89	3523±486	2404±137	47±1
InfoSAIL(Hard)	1596±147	3809±399	**2426±176**	**47±6**	3159±80	4220±90	2818±137	45±2
InfoSAIL(Soft)	**1690±83**	**4030±162**	2346±120	37±2	**3190±78**	4277±33	**3012±38**	26±13
AIRL	1815±191	320±118	1928±208	38±7	3009±165	1220±280	2531±116	42±4
VAIL	218±97	438±148	1087±151	47±3	2672± 73	3477±148	2343±271	58±5
D-REX	-366±84	232±84	921±70	24±3	-27±20	2588±75	1433±104	39±12
2IWIL	1387±206	870±86	2374±165	61±7	2904±116	3965±119	3046±86	50±8

Table 1: Performance of proposed methods and compared methods in Mujoco tasks with both stage 1 and stage 2 demonstrations, which is measured by the average and standard variance of ground-truth cumulative reward along 10 trajectories (i.e. higher average value is better).

from π_1, π_2, π_3 and π_4 while stage 2 demonstrations are formed by trajectories from π_1, π_2, π_5 and π_6, so the quality of stage 2 demonstrations is obviously better.

The reward function r_φ in WGAIL and SAIL is constrained into $[0, 5]$ by a sigmoid function. As [Kumar *et al.*, 2010] suggested, we conduct pre-training on WGAIL with about 10% of total interactions before the weight learning step in SAIL. The threshold K is initialized such that half of the demonstrations can be included. We evaluate the agent every 5,000 transitions in training and the reported result in Table 1 is the average of the last 100 evaluations. Also, we conduct our experiment with five random seeds. More details on data quality and the hyper-parameters used to reproduce the results are available in the supplementary material.

5.2 Performance

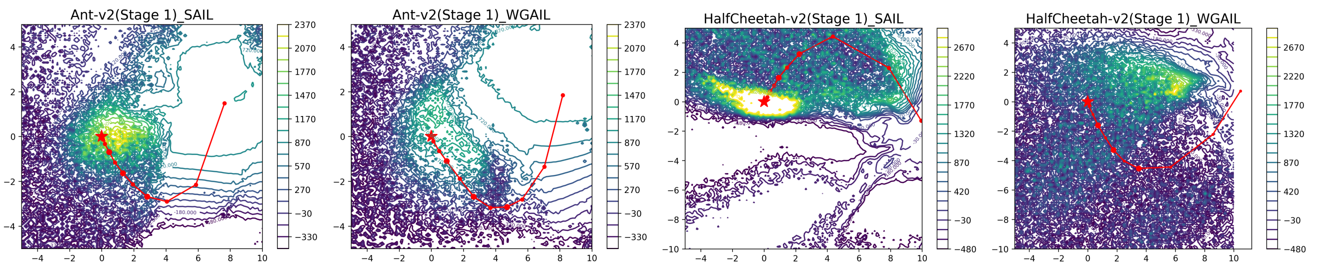
The result provided in Table 1 shows that the SAIL performs generally better than WGAIL and GAIL, which demonstrates the effect of adding weight for demonstrations. Recall that the Wasserstein variant can improve the performance over GAIL by improving the stability of GAN training empirically. We can observe that WGAIL could outperform GAIL when dealing with highly sub-optimal demonstrations (Stage 1), but it does not show quite an advantage when dealing with near-optimal demonstrations (Stage 2). We think the reason is that plain GAIL can generally perform well given near-optimal demonstrations (Stage 2), however it may suffer serious unstable training issues with highly sub-optimal demonstrations (Stage 1), which makes it hard to perform well as the results suggested. By contrast, WGAIL can well solve the unstable training problem so it could perform better than GAIL under highly imperfect demonstrations setting, but it is not likely to substantially outperform GAIL much. As for SAIL, credit to the use of weight, it can have a significant improvement over WGAIL with both stage 1 and stage 2 demonstrations. The soft label performs better than the hard label at most time and further improves the performance of SAIL.

We also compare SAIL to other GAIL-based methods such as InfoGAIL, AIRL and VAIL. Since the performance of InfoGAIL is measured by the average of all recovered modals, it may not outperform GAIL. AIRL and VAIL are proposed to address different issues about GAIL, which may help them achieve better results. As the results suggested, AIRL and VAIL only have minor improvements compared to plain GAIL and both of them do not perform well when dealing with highly imperfect demonstrations. D-REX is an IRL-based method proposed to handle imperfect demonstrations without auxiliary information, which properly suits our setting. However, since its performance may highly rely on the initial policy trained by BC, imperfect demonstrations may therefore have a negative influence on the final result of D-REX. 2IWIL requires auxiliary confidence scores as a prior for training and we can observe that SAIL reaches comparable performance with 2IWIL.

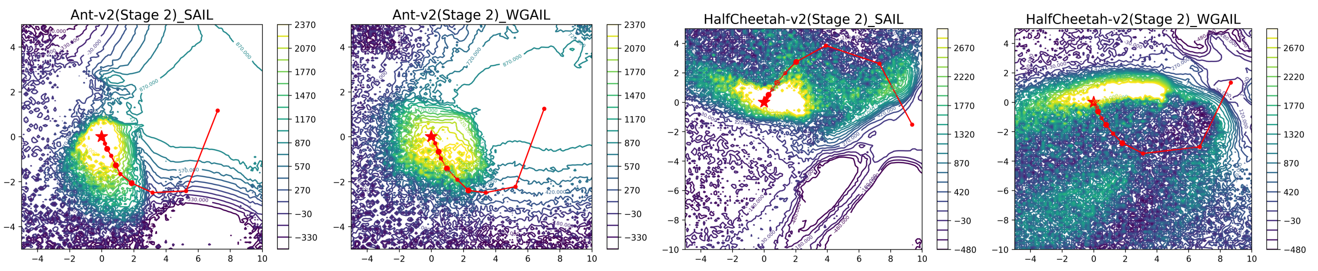
Extend to InfoGAIL. We also show that the idea of SAIL can be easily adapted to other GAIL-based methods such as InfoGAIL, as shown in Table 1. Follow the idea of SAIL, we first rewrite InfoGAIL into its Wasserstein variant (InfoWGAIL), then the weight and a regularizer on weight can be added into InfoWGAIL to get InfoSAIL. InfoGAIL performs well in near-optimal demonstrations (Stage 2), especially in HalfCheetah-v2 it nearly reaches the performance of the optimal policy. However, when given highly sub-optimal demonstrations, InfoGAIL may also inevitably suffer unstable training issues and does not perform quite well like plain GAIL. Wasserstein variant might help for improving the performance, but there is an upper bound for further improving its performance. Due to the use of weight, InfoSAIL can perform generally better than InfoWGAIL while the soft label is slightly better than the hard label generally, which is consistent with GAIL. As the result suggested, we believe the idea of SAIL can be easily adapted to other GAIL-based methods to help deal with imperfect demonstrations.

5.3 Sample Efficiency

We provide an intuitive model optimization trajectories map via a 2D weight-space slice in Figure 2, from which we can conclude that SAIL is sample efficiency. The optimiza-



(a) Sub-optimal demonstrations (Stage 1)



(b) Near-optimal demonstrations (Stage 2)

Figure 2: The optimization trajectories of policy models and reward surface in SAIL and WGAIL. The stars denote the location of the final policy model. The large points denote the location of the policy model of 1.5×10^6, 2.5×10^6 and 3.5×10^6 interactions respectively.

tion trajectories are plotted along the contours of the average reward surface for this slice. To plot this figure, we follow the instruction of [Li *et al.*, 2018]. Let θ_i denotes the parameter of policy model π_{θ_i} in i_{th} iteration and θ_n is the final policy model parameter π_{θ_n}. Given the matrix $M = [\theta_0 - \theta_n; \cdots; \theta_{n-1} - \theta_n]$, we can apply PCA to M and then select two most explanatory directions. The parameters of the policy model in different checkpoints are then projected onto the plane defined by these two directions for plotting. The model on each point is evaluated by the average cumulative reward. We save 10 checkpoints of the policy model in every 5×10^5 interactions during training, then plot the optimizer trajectories (Red dots) and average reward surface along with PCA directions.

We can observe the advantages of SAIL in Figure 2 from two aspects. First, SAIL can finally achieve higher performance than WGAIL when given sub-optimal demonstrations (Stage 1). Second, both SAIL and WGAIL achieve similar performance when given near-optimal demonstrations (Stage 2), but SAIL can converge fast.

In sub-optimal demonstrations (Stage 1), SAIL's final model (see the star symbol) is always located in the higher reward zone of parameter space while the model of WGAIL is not, as shown in Figure 2(a). Therefore, we can conclude that SAIL can achieve higher performance than WGAIL when given sub-optimal demonstrations. When given near-optimal demonstrations (Stage 2), both WGAIL and SAIL can perform well. The advantage of SAIL under this condition is that it can converge fast, which means it is sample efficiency. As shown in Figure 2(b), we can observe that in Ant-v2, both SAIL and WGAIL start heading to the optimal solution directly at around 1.5×10^6 interactions (see the first large point). However, SAIL approaches the final high reward zone at around 2.5×10^6 interactions while GAIL approaches the high reward zone at around 3.5×10^6 interactions (see the second and third large points). The same thing happens in HalfCheetah-v2. SAIL approaches the high reward zone at around 2.5×10^6 interactions. However in WGAIL, only the final policy model reaches the high reward zone. As stated above, we can conclude that SAIL is sample efficiency.

6 Conclusion

In this paper, we propose a new method called SAIL to improve the performance of GAIL when dealing with imperfect demonstrations. We use Wasserstein GAIL as a baseline and a binary weight variable can be assigned to each expert demonstration to address their influence on the objective function. The weight can be also interpreted with whether to choose the demonstration into training or not. Different from some existing solutions which require prior information about weight, we can automatically acquire this weight. Besides hard binary weighting, we also propose a soft weighting scheme for SAIL. Experiment results on Mujoco show that the advantage of SAIL over compared methods and our idea can be also easily adapted to other GAIL-based methods.

Acknowledgements

This work was supported in part by the National Key R & D Program of China under Grant 2018AAA0101100, National Natural Science Foundation of China under Grants 61822113, 41871243, the Science and Technology Major Project of Hubei Province (Next-Generation AI Technologies) under Grant 2019AEA170, the Australian Research Council under Projects DE180101438 and DP210101859.

References

[Amodei *et al.*, 2016] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[Arjovsky *et al.*, 2017] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[Bain and Sammut, 1995] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129, 1995.

[Bazaraa *et al.*, 2013] Mokhtar S Bazaraa, Hanif D Sherali, and Chitharanjan M Shetty. *Nonlinear programming: theory and algorithms*. John Wiley & Sons, 2013.

[Bennett and Demiriz, 1999] Kristin P Bennett and Ayhan Demiriz. Semi-supervised support vector machines. In *Advances in Neural Information processing systems*, pages 368–374, 1999.

[Brown *et al.*, 2019] Daniel S Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. *arXiv preprint arXiv:1904.06387*, 2019.

[Brown *et al.*, 2020] Daniel S Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on Robot Learning*, pages 330–359, 2020.

[Chen *et al.*, 2016] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016.

[Christiano *et al.*, 2017] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, pages 4299–4307, 2017.

[Codevilla *et al.*, 2018] Felipe Codevilla, Matthias Miiller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018.

[Englert *et al.*, 2013] Peter Englert, Alexandros Paraschos, Jan Peters, and Marc Peter Deisenroth. Model-based imitation learning by probabilistic trajectory matching. In *2013 IEEE International Conference on Robotics and Automation*, pages 1922–1927. IEEE, 2013.

[Fu *et al.*, 2017] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[Ho and Ermon, 2016] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in neural information processing systems*, pages 4565–4573, 2016.

[Hussein *et al.*, 2017] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

[Jiang *et al.*, 2015] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *AAAI*, volume 2, page 6, 2015.

[Kumar *et al.*, 2010] M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *Advances in neural information processing systems*, pages 1189–1197, 2010.

[Li *et al.*, 2017] Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. In *Advances in Neural Information Processing Systems*, pages 3812–3822, 2017.

[Li *et al.*, 2018] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6389–6399, 2018.

[Peng *et al.*, 2018] Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, and Sergey Levine. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. *arXiv preprint arXiv:1810.00821*, 2018.

[Puterman, 1994] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994.

[Supancic and Ramanan, 2013] James S Supancic and Deva Ramanan. Self-paced learning for long-term tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2379–2386, 2013.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Tangkaratt *et al.*, 2019] Voot Tangkaratt, Bo Han, Mohammad Emtiyaz Khan, and Masashi Sugiyama. Vild: Variational imitation learning with diverse-quality demonstrations. *arXiv preprint arXiv:1909.06769*, 2019.

[Todorov *et al.*, 2012] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[Valko *et al.*, 2013] Michal Valko, Mohammad Ghavamzadeh, and Alessandro Lazaric. Semi-supervised apprenticeship learning. In *European Workshop on Reinforcement Learning*, pages 131–142, 2013.

[Wang *et al.*, 2021] Yunke Wang, Chang Xu, Bo Du, and Honglak Lee. Learning to weight imperfect demonstrations. In *Proceedings of the 38th annual international conference on machine learning*, 2021.

[Wu *et al.*, 2019] Yueh-Hua Wu, Nontawat Charoenphakdee, Han Bao, Voot Tangkaratt, and Masashi Sugiyama. Imitation learning from imperfect demonstration. *arXiv preprint arXiv:1901.09387*, 2019.

[Xiao *et al.*, 2019] Huang Xiao, Michael Herman, Joerg Wagner, Sebastian Ziesche, Jalal Etesami, and Thai Hong Linh. Wasserstein adversarial imitation learning. *arXiv preprint arXiv:1906.08113*, 2019.

[Xu *et al.*, 2015] Chang Xu, Dacheng Tao, and Chao Xu. Multi-view self-paced learning for clustering. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[Ziebart *et al.*, 2008] Brian D Ziebart, Andrew Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. 2008.