Learning Policy from Suboptimal Demonstrations under Transition Dynamic Mismatch

Taesu Kim[1] t-s.kim@kaist.ac.kr
Jianhong Wang[2] jianhong.wang16@imperial.ac.uk
Dafni Antotsiou[2] d.antotsiou17@imperial.ac.uk
Tae-Kyun Kim[1,2] tk.kim@imperial.ac.uk
[1]*KAIST*, [2]*Imperial College London*

Abstract

Learning from demonstrations (LfD) is the problem of seeking optimal policies without true reward signals. Practical challenges arise: (1) when the environments of an agent and a demonstrator (especially, transition dynamics functions) are different and (2) when demonstrations have suboptimal performances or are too few. The prior-art, Indirect Imitation Learning (I2L), overcomes different dynamics by matching *state-only* distributions, instead of state-action distributions, however, its performance is limited to that of the demonstrator. On the other hand, the method, Trajectory-ranked Reward Extrapolation (TREX) outperforms the demonstrator by inferring a high-quality reward function from *ranked* demonstrations. The learnt reward model inevitably performs poorly under the dynamic mismatch. In this paper, we propose a novel algorithm that handles both of the challenges. It learns a reward function with ranked demonstrations while considering domain mismatches by I2L algorithm. Additionally, I2L in the proposed method is replaced with Adversarial Inverse Reinforcement Learning (AIRL) for environments with no dynamic mismatch. It takes the benefit of data augmentation effects when demonstrations are few. In the experiments on continuous physical locomotion tasks, the proposed method outperforms I2L and TREX baselines by up to 330%. Our method is shown robust to transition dynamic mismatches between the agent and demonstrator, and achieves good policies from suboptimal demonstrations. Also, the method with AIRL outperforms baselines when no dynamic mismatch.

1. Introduction

Recently, Reinforcement Learning (RL) has demonstrated impressive successes in many domains such as video games (Mnih et al., 2015; Schrittwieser et al., 2020), continuous controls (Lillicrap et al., 2016), and robotics (Levine et al., 2016). However, it is difficult to manually shape a reward function and objective that leads to desired goals. Instead, Learning from Demonstrations (LfD) (Finn et al., 2016; Ho and Ermon, 2016; Fu et al., 2018) seeks an optimal policy using demonstrations without a well-designed reward signal.

In conventional LfD settings, high-quality demonstrations are collected, i.e. a series of states and actions, $\tau \coloneqq \{s_0, a_0, \dots, a_{T-1}, s_T\}$ from the same environment where an agent is trained. Practical challenges arise in the LfD setting: (1) when the environments of an agent and demonstrator (especially, transition dynamic functions) are different and (2) when we have suboptimal or too few demonstrations. In real-world scenarios, the agent usually learns desired behaviors from demonstrations collected from another environment with different physical characteristics. Learning from Observations (LfO) methods reduce requirements for demonstrations. While expert actions strongly supervise the agent, desired actions are not quite useful under the dynamic mismatch since they are varying. Instead, desired states do not change over the dynamic mismatch, the previous methods, e.g. Indirect Imitation Learning (I2L), (Gangwani and Peng, 2020; Liu et al., 2020; Desai et al., 2020; Gangwani et al., 2022) learn from state-only than state-action pairs. Their performances are, however, limited when demonstrations are suboptimal.

Trajectory-ranked Reward Extrapolation (TREX) (Brown et al., 2019) is one of LfO algorithms which achieves beyond-demonstrator performance (Brown et al., 2020; Chen et al., 2020; Jeong et al., 2020; Yuan et al., 2021). Particularly, Brown et al. (2019) addressed the problem via learning a reward function from suboptimal *ranked* demonstrations. The parameterized reward function is first learned with the rank relations, then the agent's policy is optimized to maximize the inferred reward. These methods achieve high-quality policies by outperforming the demonstrator. This preference-based reward function, however, is not updated during the agent's policy optimization process. Under the dynamic mismatches, the model trained with only demonstrations from the demonstrator environment, inevitably performs poorly in the agent environment. Meanwhile, Adversarial Inverse Reinforcement Learning (AIRL) (Fu et al., 2018), is a sample efficient LfD method. It performs state-action distribution matching between the agent and demonstrator via adversarial learning, demonstrating good performances when demonstrations are few.

In this work, we address LfO with suboptimal demonstrations in domains with different transition dynamics. To tackle the dynamics mismatch issue, we build on Indirect Imitation Learning (I2L) (Gangwani and Peng, 2020). To overcome the suboptimal demonstrations, a simple yet effective unified method, preference-based I2L (P-I2L), is proposed, aiming at jointly learning the parameterized reward network and the agent policy. Additionally, we replace I2L with Adversarial Inverse Reinforcement Learning (AIRL) (Fu et al., 2018) when there is no dynamic mismatch. The preference-based AIRL (P-AIRL) achieves higher performances, in settings where the dataset is too small and suboptimal, thanks to the adversarial learning and ranked demonstrations. We test the effectiveness of our method for continuous locomotion tasks under mismatched transition dynamics in MuJoCo (Todorov et al., 2012). The experiments show that our method meaningfully outperforms the demonstrator. When there is no dynamic mismatch, P-AIRL achieves higher performances compared to baseline LfD methods.

2. Preliminaries

The RL environment is modeled as a Markov decision process (MDP) $\mathcal{M}$ consisting of the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma)$. $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^+$ is the transition dynamics, where given an action $a_t \in \mathcal{A}$ the next state is determined by the distribution $s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)$. $r : \mathcal{S} \to \mathbb{R}$ is the reward function, $\gamma \in (0, 1]$ is the discount factor (Puterman, 2014). The policy π provides the probability to take an action, $a \sim \pi(\cdot|s)$. The objective of RL is to maximize the expected discounted return $J(\pi) = \mathbb{E}_{p_\pi(\tau)}[\Sigma_t \gamma^t r(s_t)]$ under the distribution induced by the policy π. Below, we summarise the background components to our proposed solution: AIRL, I2L and TREX.

Adversarial Inverse Reinforcement Learning. Given a set of expert demonstration trajectories $\mathcal{D}^E = \{\tau_1, \tau_2, ...\tau_N\}$, Inverse Reinforcement Learning (IRL) algorithms aim to

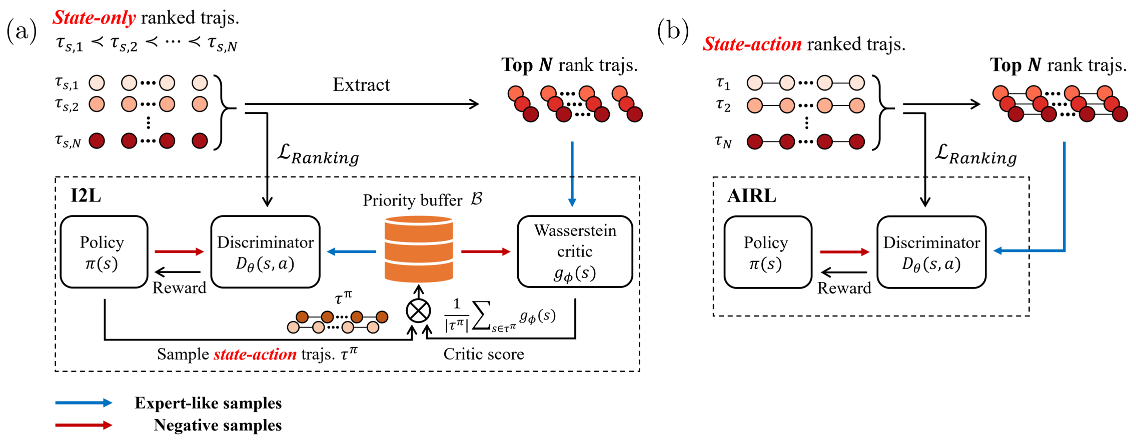
(a)
State-only ranked trajs.
$\tau_{s,1} \prec \tau_{s,2} \prec \cdots \prec \tau_{s,N}$
Extract
Top N rank trajs.
$\mathcal{L}_{Ranking}$
I2L
Priority buffer $\mathcal{B}$
Policy $\pi(s)$
Discriminator $D_\theta(s,a)$
Wasserstein critic $g_\phi(s)$
Reward
τ^π
$\frac{1}{|\tau^\pi|}\sum_{s\in\tau^\pi} g_\phi(s)$
Sample state-action trajs. τ^π
Critic score
Expert-like samples
Negative samples
(b)
State-action ranked trajs.
Top N rank trajs.
$\mathcal{L}_{Ranking}$
AIRL
Policy $\pi(s)$
Discriminator $D_\theta(s,a)$
Reward

Figure 1: (a) P-I2L has four components: the Wasserstein critic, priority buffer, discriminator, and policy. The discriminator including the reward model is learnt by the: (1) adversarial loss to distinguish high score samples in the buffer from samples from the policy and (2) supervised loss with ranked demonstrations. (b) P-AIRL plugges the ranking loss into the adversarial inverse reinforcement learning framework. Note top ranked trajectories are obtained from state-action pairs here, while they are from state-only data in I2L.

learn reward functions that explain task intentions. Adversarial Imitation Learing (Ho and Ermon, 2016; Fu et al., 2018) tries to find the reward function by generative adversarial learning. Similar to GANs (Goodfellow et al., 2014), the generator in the form of policy π and the discriminator $D : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ are trained to optimize the min-max objective $\mathbb{E}_{(s,a)\sim\mathcal{D}^E}[\log D_\theta(s,a)] + \mathbb{E}_{(s,a)\sim\pi}[\log(1 - D_\theta(s,a))]$. To infer the reward, Adversarial Inverse Reinforcement Learning (AIRL) (Fu et al., 2018) constructs the discriminator D_θ by $D_\theta(s,a) = \exp(R_\theta(s)) / (\exp(R_\theta(s)) + \pi(a|s))$. The reward function R_θ is learnt, s.t. the discriminator distinguishes expert trajectories from generator policy rollouts, and the policy π maximizes the pseudo reward obtained from R_θ.

Indirect Imitation Learning (I2L) proposed by Gangwani and Peng performs AIRL with *state-only* expert demonstrations where demonstrator actions are missing. See Figure 1a. Since AIRL requires the demonstrator actions to compute the objective scores, I2L splits the process into 2 sub-processes and introduces an intermediate demonstrator (i.e. the priority buffer) that is represented by high-quality samples generated by the policy π. In the first part, the agent policy is optimized to mimic the intermediate demonstrator, by matching *state-action* distributions via the discriminator. In the second part, the intermediate demonstrator is updated with *state-only* expert demonstrations via the critic. This way, the learnt policy is resistant to domain mismatches and yet accurate. Note ideal states do not change under different dynamic environments.

Trajectory-ranked Reward Extrapolation (TREX) (Brown et al., 2019) performs IRL using ranking information, extrapolating beyond suboptimal demonstrations. Given the ranked demonstrations, TREX first trains the reward function R_θ to approximate the true reward function. With the learnt reward function R_θ, the policy π is optimized via RL and the high-quality reward model enables the agent to perform better than the demonstrator. However, in two cases: first when the imitator is learned in a different environment than the demonstrator, and second when the number of ranked demonstrations is small, the reward model inevitably performs poorly degrading performance.

3. Our Approach

We propose a unified framework that learns a parameterized reward function and optimizes a high-quality policy network in an end-to-end manner, using suboptimal ranked demonstrations. The pipeline of our method is shown in Figure 1a. Our method is motivated by that TREX, which enables the agent to perform better than given demonstrations, might overfit to an environment where the demonstrations are collected. Under the transition dynamics mismatch between the agent and demonstrator, the reward model inevitably performs poorly. On the other hand, I2L and AIRL, which train the reward model (i.e. discriminator) with demonstrations collected in the agent and demonstrator environments, do not provide a powerful reward signal that enables the agent to generate a better behavior than the demonstrator. In the following Sections, we introduce Preference-based Indirect Imitation Learning (P-I2L) that learns from demonstrations when the imitator and demonstrator MDP have different transition dynamics, then Preference based Adversarial Inverse Reinforcement Learning (P-AIRL).

3.1 Preference-based Indirect Imitation Learning (P-I2L)

Let's consider the loss function of TREX. Given the *state-only* ranked demonstrations $\mathcal{D}_\mathrm{s} = \{\tau_{\mathrm{s},1}, \tau_{\mathrm{s},2}, ..., \tau_{\mathrm{s},N}\}$, where $\tau_{\mathrm{s},i} \prec \tau_{\mathrm{s},j}$ if $i < j$, the reward function R_θ is trained via supervised learning. When $\tau_{\mathrm{s},i} \prec \tau_{\mathrm{s},j}$, $\Sigma_{s\in\tau_{\mathrm{s},i}} R_\theta(s) < \Sigma_{s\in\tau_{\mathrm{s},j}} R_\theta(s)$, with the following according to the Luce-Shepard rule (Luce, 2012):

$$\mathcal{L}_{\text{ranking}}(\theta) = -\sum_{\tau_{\mathrm{s},i}\prec\tau_{\mathrm{s},j}} \log \frac{\exp\big(\sum_{s\in\tau_{\mathrm{s},j}} R_\theta(s)\big)}{\exp\big(\sum_{s\in\tau_{\mathrm{s},i}} R_\theta(s)\big) + \exp\big(\sum_{s\in\tau_{\mathrm{s},j}} R_\theta(s)\big)}. \tag{1}$$

The I2L architecture is shown in Figure 1a. Let $\mathcal{D}_\mathrm{s}^*$ be the top N rank demonstrations that are extracted from the suboptimal *state-only* demonstrations $\mathcal{D}_\mathrm{s}$. I2L divides the process into 2 sub-processes. Let $\mathcal{B}$ be a priority buffer that consists of the trajectories generated by the policy in the imitator environment during the training. Let $\tilde{\rho}(s)$ and $\tilde{\rho}(s,a)$ be the state and state-action distribution of the trajectories in $\mathcal{B}$. I2L measures Wasserstein-1 distance between $\tilde{\rho}(s)$ and $\rho^*(s)$, where $\rho^*(s)$ is the state distribution of expert demonstrations, and trains the critic network g_ϕ with Lipschitz continuity constraint s.t.

$$W_1(\tilde{\rho}(s), \rho^*(s)) = \sup_{||g_\phi||_L\le 1} \mathbb{E}_{s\sim\rho^*}[g_\phi(s)] - \mathbb{E}_{s\sim\tilde{\rho}}[g_\phi(s)]. \tag{2}$$

The critic network measures the score reflecting the similarity of the state distributions between the best trajectories and samples generated by the policy network.

The priority buffer selects trajectories from the policy with scores provided by the critic network. Since the actions are available in $\mathcal{B}$, the state-action distribution is used to train the policy in AIRL manner. Thus, the discriminator D_θ trains the reward model R_θ to distinguish high quality samples in $\mathcal{B}$ and samples from the policy.

$$\mathcal{L}_{\text{adv}}(\theta) = \mathbb{E}_{(s,a)\sim\mathcal{B}}[\log(1 - D_\theta(s,a))] + \mathbb{E}_{(s,a)\sim\pi}[\log D_\theta(s,a)]. \tag{3}$$

The agent policy π is trained with R_θ. The reward function R_θ is learnt with the total loss

$$\mathcal{L}_{\text{total}}(\theta) = \mathcal{L}_{\text{adv}}(\theta; (s,a)\sim\mathcal{B}, (s,a)\sim\pi) + \mathcal{L}_{\text{ranking}}(\theta; (s)\sim\mathcal{D}_\mathrm{s}). \tag{4}$$

The reward R_θ in P-I2L is trained with ranked demonstrations as in TREX, while learnt to distinguish sampled demonstrations by the policy τ^π and $\mathcal{B}$. Therefore, it is robust under transition dynamics mismatch. See Algorithm 1 in Appendix A.

3.2 Preference-based Adversarial Inverse Reinforcement Learning (P-AIRL)

Our method can also be adapted to the settings where suboptimal demonstrations are collected under the same dynamics environments. Let $\mathcal{D}$ be *state-action* ranked demonstrations. Under a small number of $\mathcal{D}$, TREX fails to recover the true reward signal. Instead, AIRL trains the reward network in adversarial manner, using expert demonstrations $\mathcal{D}^*$ and policy-generated data. The proposed method P-AIRL learns R_θ with the total loss

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{adv}}(\theta, (s, a) \sim \mathcal{D}^*, (s, a) \sim \pi) + \mathcal{L}_{\text{ranking}}(\theta, (s) \sim \mathcal{D}). \tag{5}$$

It exploits not only a small number of $\mathcal{D}$ but also policy-generated data. The P-AIRL is implemented by replacing I2L with AIRL in P-I2L. The discriminator D_θ tries to maximize the chance to distinguish top N demonstrations $\mathcal{D}^*$ as expert and policy-generated data as negative samples. The detailed algorithm is provided in Algorithm 2 in Appendix A.

4. Experiments

We evaluated the proposed methods for three robotic control tasks in MuJoCo (Todorov et al., 2012) within OpenAI Gym (Brockman et al., 2016): HalfCheetah, Hopper, Walker2D.

P-I2L with Transition Dynamics Mismatch. We created GravityDouble /Heavy/ HighFrictionBody environments by modifying properties such as gravity, density, frictional coefficients of the three original demonstration environments. In order to evaluate under transition dynamic mismatches between the demonstrator and imitator, we collected ranked demonstrations from default environments, and evaluated the methods in the modified environments. We compared our P-I2L algorithm with the following baselines, GAIL, GAIL-s(GAIL with the state-dependent discriminator), I2L, and TREX. See Appendix B.1 for the experiment setup in details.

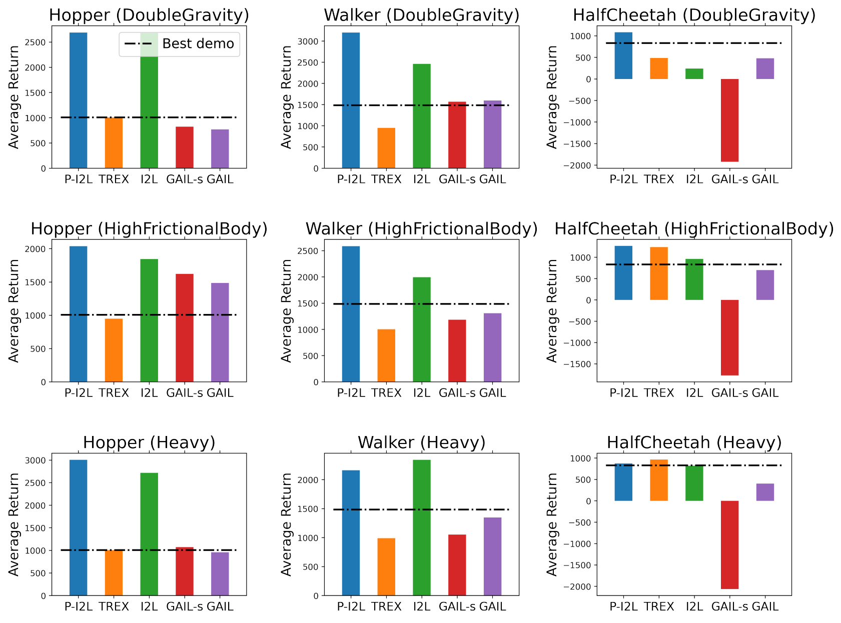

Figure 2: Comparison of P-I2L, TREX, I2L, GAIL-s, GAIL, and the demonstrator's best performance, under dynamic mismatches. The imitator and demonstrator MDPs are different.

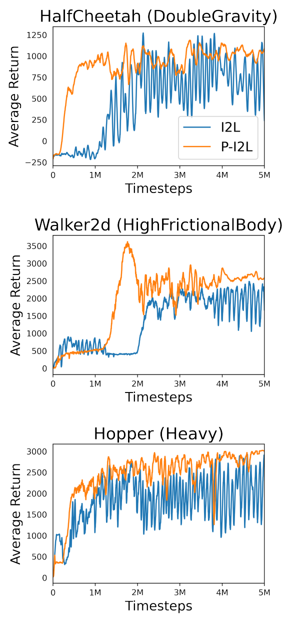

Figure 3: Training progress for P-I2L and I2L.

Figure 2 shows the performance comparison of P-I2L and baselines. The policies learned by P-I2L successfully outperformed the demonstrator in all cases. I2L also outperforms the demonstrator in some cases, however, as shown in Figure 3, the performance of I2L highly fluctuates over training iterations. Since GAIL, GAIL-s and I2L imitates the demonstrations by performing state-action/state distribution matching, the agent policy fails to achieve beyond-demonstrator performance. TREX outperforms the demonstrations in 2 out of 9 environments. It is designed to infer the reward from states, being less sensitive to dynamic differences. However, TREX trains the reward function R_θ only using the ranked demonstrations collected from the demonstrator's environment. In our experimental settings, the demonstrations are not enough to generalize R_θ and R_θ causes the poor performance in the agent's environment. Similar to I2L, P-I2L trains R_θ using demonstrations in $\mathcal{B}$, which are sampled from the agent environment, as well as ranked demonstrations from the demonstration environment. This way, R_θ is more robust under different domains and better approximates the true environmental reward, which enables the agent to achieve better-than-demonstrator performance. P-I2L outperforms other baselines in 7 out of 9 environments.

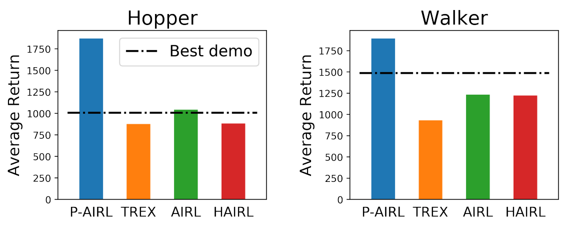

Figure 4: Comparison of P-AIRL, TREX, AIRL, HAIRL, and the demonstrator's performance.

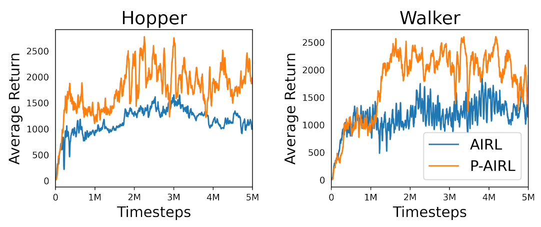

Figure 5: Training progress for P-AIRL and AIRL in Hopper and Walker environments.

P-AIRL without Transition Dynamics Mismatch. We compared our method, P-AIRL with the following baselines, AIRL (P-AIRL without $\mathcal{L}_{\text{ranking}}$, TREX, and HAIRL, proposed by Yuan et al. (2021), which achieves better-than-demonstrator performance, based on curiosity-driven reward inference.

In TREX, the trained reward network is not well generalized due to the small number of ranked demonstrations; thus, it fails to outperform the demonstrator. In AIRL, the agent successfully imitated the demonstrations for Hopper, but the agent performance for Walker is about 40% lower than the demonstrations. As shown in Figure 5, the agent performance fluctuates over time, since the reward network is trained to maximize the reward for a given demonstration. On the other hand, our method relatively less fluctuates and is shown outperforming the baseline methods.

5. Conclusion

In this paper, we propose P-I2L, an end-to-end LfO method that learns from suboptimal demonstration, while considering transition dynamics mismatch between the agent and the demonstrator. We next propose P-AIRL, an LfD algorithm that enables to learn the reward when only a small number of demonstrations is available. We empirically evaluated P-I2L and P-AIRL on several continuous physical simulations and showed that our algorithms successfully outperform the best demonstration and even state-of-the-art methods in 9 out of 11 experiments.

Acknowledgments

We would like to acknowledge support for this project from KAIA grant (22CTAP-C163793-02) funded by the Korea government(MOLIT), NST grant (CRC 21011) funded by the Korea government(MSIT), KOCCA grant (R2022020028) funded by the Korea government(MCST), and a grant from the Samsung Display corporation.

References

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International Conference on Machine Learning*, 2019.

Daniel S Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on Robot Learning*, 2020.

Letian Chen, Rohan Paleja, and Matthew Gombolay. Learning from suboptimal demonstration via self-supervised reward regression. *arXiv preprint arXiv:2010.11723*, 2020.

Siddharth Desai, Ishan Durugkar, Haresh Karnan, Garrett Warnell, Josiah Hanna, Peter Stone, and AI Sony. An imitation from observation approach to transfer learning with dynamics mismatch. *Advances in Neural Information Processing Systems*, 2020.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. Openai baselines. `https://github.com/openai/baselines`, 2017.

Chelsea Finn, Paul F. Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *CoRR*, 2016. URL `http://arxiv.org/abs/1611.03852`.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learning. In *International Conference on Learning Representations*, 2018.

Tanmay Gangwani and Jian Peng. State-only imitation with transition dynamics mismatch. In *International Conference on Learning Representations*, 2020.

Tanmay Gangwani, Yuan Zhou, and Jian Peng. Imitation learning from observations under transition model disparity. *arXiv preprint arXiv:2204.11446*, 2022. URL `https://arxiv.org/abs/2204.11446`.

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*, 2014.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, 2016.

Rae Jeong, Jost Tobias Springenberg, Jackie Kay, Daniel Zheng, Yuxiang Zhou, Alexandre Galashov, Nicolas Heess, and Francesco Nori. Learning dexterous manipulation from suboptimal experts. *arXiv preprint arXiv:2010.08587*, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 2016.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Fangchen Liu, Zhan Ling, Tongzhou Mu, and Hao Su. State alignment-based imitation learning. In *International Conference on Learning Representations*, 2020.

R Duncan Luce. *Individual choice behavior: A theoretical analysis.* Courier Corporation, 2012.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 2015.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming.* John Wiley & Sons, 2014.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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*, 2012.

Mingqi Yuan, Man-On Pun, Yi Chen, and Qi Cao. Hybrid adversarial inverse reinforcement learning. *arXiv preprint arXiv:2102.02454*, 2021.

Appendix A. Algorithms

Algorithm 1 Preference-based Indirect Imitation Learning

```
1: Input: Suboptimal state-only demonstrations $\mathcal{D}_\mathrm{s} = \{\tau_{\mathrm{s},i}\}_{i=1}^N$
2: Extract top $N$ rank demonstrations $\mathcal{D}_\mathrm{s}^*$ from $\mathcal{D}_\mathrm{s}$
3: Initialize the policy network $\pi$, discriminator $D_\theta$ and Wasserstein critic $g_\phi$
4: Empty the priority buffer $\mathcal{B}$
5: for step t do
6:     Generate trajectories $\tau_i^\pi = (s_0^i, a_0^i, ..., s_{T_i}^i, a_{T_i}^i)$ by the policy $\pi$
7:     Train $g_\phi$ using $\mathcal{B}$ and $\mathcal{D}_\mathrm{s}^*$ from Eqn 2
8:     Update the score for each $\tau^\pi$ using $g_\phi$
9:     Update $\mathcal{B}$ with $\tau^\pi$ to store highest score trajectories
10:    Update $R_\theta$ from Eqn 4 using $\tau^\pi$, $\mathcal{B}$ and $\mathcal{D}_\mathrm{s}$
11:    Update $\pi$ with PPO using the reward function $R_\theta$
12: end for
13: Output: Policy $\pi$
```

Algorithm 2 Preference-based AIRL

```
1: Input: Suboptimal state-action demonstrations $\mathcal{D} = \{\tau_i\}_{i=1}^N$
2: Extract top $N$ rank demonstrations $\mathcal{D}^*$ from $\mathcal{D}$
3: Initialize poicy $\pi$ and Discriminator $D_\theta$
4: for step t do
5:     Generate trajectories $\tau_i^\pi = (s_0^i, a_0^i, ..., s_{T_i}^i, a_{T_i}^i)$ by the policy $\pi$
6:     Update $R_\theta$ to minimize $\mathcal{L}_\mathrm{total}$ using $\tau^\pi$, $\mathcal{D}^*$ and $\mathcal{D}$.
7:     Update $\pi$ with PPO using the reward function $R_\theta$
8: end for
9: Output: Policy $\pi$
```

Appendix B. Experiment Details

B.1 P-I2L with Transition Dynamics Mismatch

In this appendix we explain more details of the experiment for Section 4:

B.1.1 Experimental Setup

We trained the demonstrator agent via PPO (Schulman et al., 2017) using ground truth rewards for 500 training steps (1M environmental steps) and checkpointed the policy for every 10 training steps. We used the PPO implementation from OpenAI baselines (Dhariwal et al., 2017) with the default hyperparameters. To construct the ranked demonstrations, we generated a trajectory of length 1000 per each policy checkpoint. For Hopper and HalfCheetah, we used worst 13 and 11 trajectories. For Walker, we used worst 6 trajectories generated by the policy checkpoints.

We created new environments by changing some properties of the original demonstration environments, in order to evaluate under transition dynamic mismatches between the demonstrator and imitator. First, we used the default environments HalfCheetah, Hopper and Walker in MuJoCo to collect the demonstrations. We then created GravityDouble/Heavy/HighFrictionBody environments by modifying the corresponding parameters of each environment for the imitator. GravityDouble has 2.0× gravity of the default environment. Heavy has 2.0× density and HighFrictionBody has 3.0× frictional coefficient on all the leg and tail joints of the default environment.

We extracted single ($N = 1$) best suboptimal trajectory from ranked demonstrations as an expert, and then trained the framework to surpass the demonstrator performance. The proposed framework consists of 4 components: the priority buffer, the policy, the discriminator and the critic. We fixed the capacity of the priority buffer to 10 episodes. The policy network has 3 fully connected layers of 64 units with Tanh nonlinearities. We train the discriminator network using the Adam optimizer (Kingma and Ba, 2015) with a learning rate of $3e$-4. The critic network has 4 fully connected layers of 64 units with Tanh nonlinearities. We train the critic network using the RMS-Prop optimizer with a learning rate of $5e$-5.

B.2 P-AIRL without Transition Dynamics Mismatch

In this appendix we explain more details of the experiment for Section 4:

B.2.1 Experimental Setup

To evaluate the P-AIRL method, we conducted the experiments in the default configuration environments for Hopper and Walker2D. We used the same ranked demonstrations collected in Section 4. We first extracted a single best suboptimal trajectory from ranked demonstrations as an expert, and then trained the framework. The policy network has 3 fully connected layers of 64 units with Tanh nonlinearities. We train the discriminator network using the Adam optimizer with a learning rate of 3e-4.