DIFFUSION REWARDS GUIDED ADVERSARIAL IMITATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Imitation learning aims to learn a policy from observing expert demonstrations without access to reward signals from environments. Generative adversarial imitation learning (GAIL) formulates imitation learning as adversarial learning, employing a generator policy learning to imitate expert behaviors and discriminator learning to distinguish the expert demonstrations from agent trajectories. Despite its encouraging results, GAIL training is often brittle and unstable. Inspired by the recent dominance of diffusion models in generative modeling, this work proposes diffusion rewards guided adversarial imitation learning (DRAIL), which integrates a diffusion model into GAIL, aiming to yield more precise and smoother rewards for policy learning. Specifically, we propose a diffusion discriminative classifier to construct an enhanced discriminator; then, we design diffusion rewards based on the classifier's output for policy learning. We conduct extensive experiments in navigation, manipulation, and locomotion, verifying DRAIL's effectiveness compared to prior imitation learning methods. Moreover, additional experimental results demonstrate the generalizability and data efficiency of DRAIL. Visualized learned reward functions of GAIL and DRAIL suggest that DRAIL can produce more precise and smoother rewards.

1 INTRODUCTION

Imitation learning, or learning from demonstration (Schaal, 1997; Hussein et al., 2017; Osa et al., 2018), aims to learn an agent policy by observing and mimicking the behavior demonstrated in expert demonstrations. Various imitation learning methods (Zhao et al., 2023; Swamy et al., 2023) have enabled deploying reliable and robust learned policies in a variety of tasks involving sequential decision-making, especially in scenarios where formulating a reward function is intricate or uncertain (Christiano et al., 2017; Leike et al., 2018; Lee et al., 2019), or when learning in a trial-and-error manner is expensive or unsafe (Garcıa & Fernández, 2015; Gu et al., 2022).

Among various directions in imitation learning, generative adversarial imitation learning (GAIL) (Ho & Ermon, 2016) has received increasing attention. GAIL learns a generator policy to imitate expert behaviors through reinforcement learning and a discriminator to differentiate between the expert and the generator's state-action pair distributions, resembling the idea of generative adversarial networks (GANs; Goodfellow et al., 2014). Despite its encouraging results, GAIL training can often be brittle and unstable. To address this issue, significant efforts have been put into improving GAIL's sample efficiency, scalability, robustness, and generalizability by modifying loss functions (Fu et al., 2018), designing policy learning algorithms (Kostrikov et al., 2019a), and exploring similarity measures of distributions (Fu et al., 2018; Arjovsky et al., 2017; Dadashi et al., 2020).

Inspired by the recent dominance of diffusion models in generative modeling (Ho et al., 2020), this work explores incorporating diffusion models into GAIL to provide more precise and smoother reward functions for policy learning. An intuitive implementation involves learning a diffusion model to reconstruct a real/fake label conditioning on a state-action pair. However, using such a diffusion model to compute a reward for a single policy step requires undergoing an iterative generation process, taking hundreds of reverse diffusion steps, which is highly time-consuming and thus impractical for reinforcement learning policies.

In this work, we propose a diffusion discriminative classifier, which learns to classify a state-action pair into expert demonstrations or agent trajectories with merely two reverse diffusion steps. Then,

we leverage the proposed diffusion discriminative classifier to devise diffusion rewards, which reward agent behaviors that closely align with expert demonstrations. Putting them together, we present **D**iffusion **R**ewards Guided **A**dversarial **I**mitation **L**earning (**DRAIL**), a novel adversarial imitation learning framework that can efficiently and effectively produce reliable policies replicating the behaviors of experts.

We extensively compare our proposed framework DRAIL with behavioral cloning methods, *e.g.*, Diffusion Policy (Pearce et al., 2023; Chi et al., 2023), and AIL methods, *e.g.*, GAIL (Ho & Ermon, 2016) and WAIL (Arjovsky et al., 2017), in diverse continuous control domains, including navigation, robot arm manipulation, and locomotion. The experimental results show that our proposed framework consistently outperforms the baselines or achieves competitive performance. We evaluate the generalizability of the policies learned by different methods, demonstrating the superiority of DRAIL. Moreover, we vary the amounts of available expert data, and the results suggest that DRAIL is more data-efficient than BC and GAIL. We visualize the reward functions learned by GAIL and DRAIL, which shows that DRAIL captures more precise and smoother rewards.

2 Related Work

Imitation learning enables agents to learn from expert demonstration to acquire complex behaviors without explicit reward functions. Its application spans various domains, including robotics (Schaal, 1997; Zhao et al., 2023), autonomous driving (Ly & Akhloufi, 2020), and game AI (Harmer et al., 2018).

Behavioral Cloning (BC). BC (Pomerleau, 1989; Torabi et al., 2018) imitates an expert policy through supervised learning and is widely used for its simplicity and effectiveness across various domains. Despite its benefits, BC struggles to generalize to states not covered in expert demonstrations because of compounding error (Ross et al., 2011; Florence et al., 2022). While some methods aim to restrict the agent from deviating (Ross et al., 2011; Zhao et al., 2023), the challenge of generalization persists.

Inverse Reinforcement Learning (IRL). IRL methods (Ng & Russell, 2000) aim at inferring a reward function that could best explain the demonstrated behavior and subsequently learn a policy using the inferred reward function. Nevertheless, the problem of inferring reward functions is ill-posed since different reward functions could induce the same demonstrated behavior. Therefore, IRL methods often impose constraints on reward functions or policies to ensure the optimality and uniqueness of the demonstrated behavior (Ng & Russell, 2000; Abbeel & Ng, 2004; Syed et al., 2008; Ziebart et al., 2008). Yet, these constraints could potentially restrict the generalizability of learned policies.

Adversarial Imitation Learning (AIL). Instead of inferring underlying reward functions, AIL methods aim to directly match the state-action distributions of an agent and an expert through adversarial training. Generative adversarial imitation learning (GAIL; Ho & Ermon, 2016) and its extensions (Torabi et al., 2019; Kostrikov et al., 2019b; Zolna et al., 2021; Jena et al., 2021) train a generator policy to imitate expert behaviors and a discriminator to differentiate between the expert and the generator's state-action pair distributions, which resembles the idea of generative adversarial networks (GAN; Goodfellow et al., 2014).

Due to its simplicity and effectiveness, GAIL has been widely applied to various domains (Aytar et al., 2018; Ravichandar et al., 2020; Kiran et al., 2021). Over the past years, researchers and practitioners have proposed numerous improvements to enhance GAIL's sample efficiency, scalability, and robustness (Orsini et al., 2021), including modifications to discriminator's loss function (Fu et al., 2018), extensions to off-policy RL algorithms (Kostrikov et al., 2019a), and exploration of various similarity measures (Fu et al., 2018; Arjovsky et al., 2017; Dadashi et al., 2020). In this work, to improve GAIL's generalization capabilities, we propose to use the diffusion model as a discriminator in GAIL by designing an objective.

3 Preliminaries

This work proposes a novel adversarial imitation learning framework that integrates a diffusion model into generative adversarial imitation learning. Hence, this section presents background on these two topics.

3.1 GENERATIVE ADVERSARIAL IMITATION LEARNING (GAIL)

GAIL (Ho & Ermon, 2016) establishes a connection between generative adversarial network (GAN) (Goodfellow et al., 2014) and imitation learning. GAIL employs a generator, G_θ, that acts as a policy π_θ, mapping a state to an action. The generator aims to produce a state-action distribution (ρ_{π_θ}) which closely resembles the expert state-action distribution ρ_{π_E}. On the other hand, discriminator D_ω functions as a binary classifier, attempting to differentiate the state-action distribution of the generator (ρ_{π_θ}) from the expert's (ρ_{π_E}). The optimization equation of GAIL can be formulated using the Jensen-Shannon divergence, which is equivalent to the minimax equation of GAN. The minimax optimization of GAIL can be derived as follows:

$$\min_\theta \max_\omega \mathbb{E}_{\mathbf{x}\sim\rho_{\pi_\theta}}[\log D_\omega(\mathbf{x})] + \mathbb{E}_{\mathbf{x}\sim\rho_{\pi_E}}[\log(1 - D_\omega(\mathbf{x}))], \tag{1}$$

where ρ_{π_θ} and ρ_{π_E} are the state-action distribution from an agent π_θ and expert policy π_E respectively. The loss function for the discriminator is stated as $-(\mathbb{E}_{\mathbf{x}\sim\rho_{\pi_\theta}}[\log D_\omega(\mathbf{x})]+\mathbb{E}_{\mathbf{x}\sim\rho_{\pi_E}}[\log(1-D_\omega(\mathbf{x}))])$. For a given state, the generator tries to take expert-like action; the discriminator takes state-action pairs as input and computes the probability of the input originating from an expert. Then the generator uses a reward function $-\mathbb{E}_{\mathbf{x}\sim\rho_{\pi_\theta}}[\log D_\omega(\mathbf{x})]$ or $-\mathbb{E}_{\mathbf{x}\sim\rho_{\pi_\theta}}[\log D_\omega(\mathbf{x})] + \lambda H(\pi_\theta)$ to optimize its network parameters, where entropy term H is a policy regularizer controlled by $\lambda \geq 0$.

3.2 DIFFUSION MODELS

Diffusion models have demonstrated state-of-the-art performance on a wide range of tasks (Sohl-Dickstein et al., 2015; Nichol & Dhariwal, 2021; Dhariwal & Nichol, 2021) In this work, we employ a Denoising Diffusion Probabilistic Model (DDPM) (Ho et al., 2020) to model expert state-action pairs and adapt it to condition on binary classification labels (real and fake).

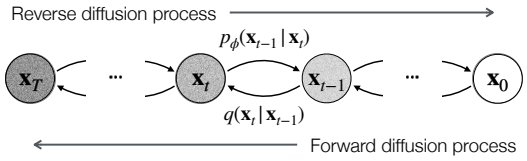


Figure 1: **Denoising Diffusion Probabilistic Model.** Latent variables $\mathbf{x}_1, ..., \mathbf{x}_T$ are produced from the data point $\mathbf{x}_0$ via the forward diffusion process, i.e., gradually adding noises to the latent variables. The diffusion model ϕ learns to reverse the process by denoising the noisy data to reconstruct the original data point $\mathbf{x}_0$.

DDPM introduces a gradual adding of noise to data samples (*i.e.*, concatenated state-action pairs) following a variance schedule $\beta_1, \ldots, \beta_t$ until achieving an isotropic Gaussian distribution (refer to as the *forward diffusion process*). This process is mathematically expressed as:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{2}$$

One of the forward process properties allows direct closed-form sampling of data $\mathbf{x}_t$ at any arbitrary time step t:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, \bar{\alpha}_t\mathbf{I}) \tag{3}$$

where $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. Consequently, $\mathbf{x}_t$ is expressed as a linear combination of $\mathbf{x}_0$ and noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$:

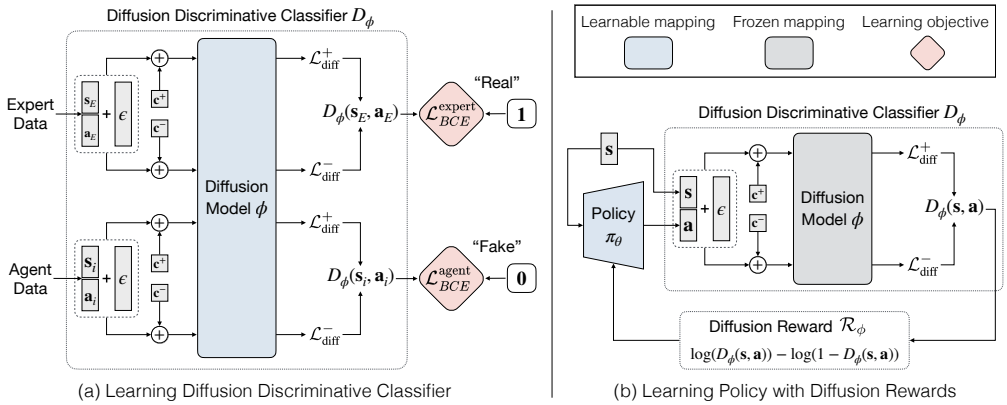
$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \tag{4}$$

The diffusion model ϕ is trained to predict the noise $\boldsymbol{\epsilon}_\phi(\mathbf{x}_t, t)$ applied to the original data, guided by the loss function

$$\mathcal{L}_{\text{DM}} = \mathbb{E}_{t\sim T}\left[\|\boldsymbol{\epsilon}_\phi(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) - \boldsymbol{\epsilon}\|^2\right] \tag{5}$$

This process facilitates the model's ability to reconstruct original data samples through the *reverse diffusion process*, as illustrated in Figure 1. In essence, DDPM learns to discern a state-action distribution by effectively denoising noisy sampled data.

This paper adapts the original training setup to learn the forward and reverse processes in a conditional manner. Specifically, the forward process captures the data distribution $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y})$, enabling the model to sample data based on a conditional classifier $\mathbf{y}$. Conversely, modeling the conditional data distribution $p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y})$ allows the generation of samples with attributes corresponding to the condition $\mathbf{y}$.



(a) Learning Diffusion Discriminative Classifier (b) Learning Policy with Diffusion Rewards

Figure 2: **Diffusion Rewards Guided Adversarial Imitation Learning.** Our proposed framework DRAIL incorporates a diffusion model into GAIL. **(a)** We propose a diffusion discriminative classifier that learns to distinguish expert data $(\mathbf{s}_E, \mathbf{a}_E)$ from agent data $(\mathbf{s}_i, \mathbf{a}_i)$ using a diffusion model ϕ. This classifier learns to denoise expert and agent state-action pairs concatenated with a real label $\mathbf{c}^+$ or a fake label $\mathbf{c}^-$ by optimizing the binary cross-entropy losses $\mathcal{L}_{BCE}^{\text{expert}}$ and $\mathcal{L}_{BCE}^{\text{agent}}$. **(b)** The π_θ learns to maximize the diffusion reward $\mathcal{R}_\phi$ computed based on the output of the diffusion discriminative classifier D_ϕ that takes the state-action pairs from the policy as input.

4 Approach

We propose a novel adversarial imitation learning framework incorporating diffusion models into the generative adversarial imitation learning (GAIL) framework, illustrated in Figure 2. Specifically, we employ a diffusion model to construct an enhanced discriminator, diffusion discriminative classifier, to provide more precise and smoother rewards for policy learning. We initiate our discussion by exploring a naive integration of the diffusion model, which directly reconstructs the reward values from Gaussian noise conditioned on the state-action pairs. However, as detailed in Section 4.1, we identify inherent issues with this naive approach. Subsequently, in Section 4.2, we introduce our proposed method that employs a conditional diffusion model to construct a diffusion discriminative classifier, which can provide diffusion rewards for policy learning. Finally, the overall algorithm of our method is outlined in Section 4.3.

4.1 Reward Reconstruction with Diffusion Model

The conditional diffusion model is widely employed for data generation, involving sampling data from the trained diffusion model. An intuitive approach to incorporating a conditional diffusion model as a GAIL discriminator by training it to denoise a real/fake label conditioned on expert or agent state-action pairs. Then, we can reward the policy based on the reconstructed realness from sampled Gaussian noise conditioned on the state-action pairs from the policy. Specifically, the diffusion model $p_\phi(R_{t-1}|R_t, \mathbf{s}, \mathbf{a})$ learns to denoise each time step t by conditioning on respective state-action pairs $(\mathbf{s}, \mathbf{a})$ and restoring desired reward value $R_0 \in \{0, 1\}$, *i.e.*, 1 for expert state-action pairs and 0 for agent state-action pairs, from isotropic Gaussian through a reverse diffusion process. The loss function $\mathcal{L}_R$ is the expected squared difference between the original noise and the predicted one.

$$\mathcal{L}_R(\mathbf{s}, \mathbf{a}) = \mathbb{E}_{t\sim T}\left[\|\boldsymbol{\epsilon}_\phi(R_t, t|\mathbf{s}, \mathbf{a}) - \boldsymbol{\epsilon}\|^2\right] = \mathbb{E}_{t\sim T}\left[\|\hat{\boldsymbol{\epsilon}}_\phi(R_0, \boldsymbol{\epsilon}, t|\mathbf{s}, \mathbf{a}) - \boldsymbol{\epsilon}\|^2\right] \tag{6}$$

To specify further, the loss functions for expert $\mathcal{L}_E$ and agent $\mathcal{L}_A$ state-action pairs are defined as:

$$\mathcal{L}_E = \mathbb{E}_{t\sim T,(\mathbf{s},\mathbf{a})\sim D_E}\left[\|\hat{\boldsymbol{\epsilon}}_\phi(1, \boldsymbol{\epsilon}, t|\mathbf{s}, \mathbf{a}) - \boldsymbol{\epsilon}\|^2\right], \mathcal{L}_A = \mathbb{E}_{t\sim T,(\mathbf{s},\mathbf{a})\sim D_A}\left[\|\hat{\boldsymbol{\epsilon}}_\phi(0, \boldsymbol{\epsilon}, t|\mathbf{s}, \mathbf{a}) - \boldsymbol{\epsilon}\|^2\right]. \tag{7}$$

Following training, the trained diffusion model can be employed to sample the reward value by progressively restoring the data sample from Gaussian noise through a reverse diffusion process conditioned on the provided state-action pair. The resultant reconstructed reward value subsequently serves as a training signal for the policy.

Nevertheless, the sampling process is time-consuming as each sample requires T iterations to restore the actual reward value, and RL often requires millions of samples for training, resulting in a billion-

level overall training scale. Consequently, it is impractical to intuitively integrate the diffusion model into the GAIL framework by directly reconstructing reward values.

4.2 DIFFUSION DISCRIMINATIVE CLASSIFIER

Our goal is to yield a diffusion model reward given an agent state-action pair without going through the whole diffusion generation process. To this end, we proposed to leverage the diffusion model ϕ to predict the noise injected into the input state-action pairs, which requires only a single denoising step, instead of direct reward reconstruction. Our key insight is that the diffusion loss, *i.e.*, the difference between predicted noise and input noise, can indicate how well the state-action pair fits the expert demonstration, effectively acting as the reward value for a given state-action pair. We formulate the diffusion loss $\mathcal{L}_{\text{diff}}$ as follows:

$$\mathcal{L}_{\text{diff}}(\mathbf{s}, \mathbf{a}, \mathbf{c}) = \mathbb{E}_{t\sim T}\left[\|\hat{\boldsymbol{\epsilon}}_\phi(\mathbf{s}, \mathbf{a}, \boldsymbol{\epsilon}, t|\mathbf{c}) - \boldsymbol{\epsilon}\|^2\right] \tag{8}$$

Here, $\mathbf{c} \in \{\mathbf{c}^+, \mathbf{c}^-\}$, real label $\mathbf{c}^+$ represents the condition for fitting expert demonstration while fake label $\mathbf{c}^-$ represents the opposite. We implement $\mathbf{c}^+$ as $\mathbf{1}$ and $\mathbf{c}^-$ as $\mathbf{0}$. Then, we use $\mathcal{L}_{\text{diff}}(\mathbf{s}, \mathbf{a}, \mathbf{c}^+)$ and $\mathcal{L}_{\text{diff}}(\mathbf{s}, \mathbf{a}, \mathbf{c}^-)$ to indicate how well the input state-action pair fits expert or agent demonstration[1]. More specifically, when the input data is sampled from expert demonstration, $\mathcal{L}^+_{\text{diff}}$ should be close to 0, and $\mathcal{L}^-_{\text{diff}}$ should be as large as possible. On the contrary, when the input data is sampled from agent storage, $\mathcal{L}^+_{\text{diff}}$ should be as large as possible, and $\mathcal{L}^-_{\text{diff}}$ should close to 0.

While $\mathcal{L}_{\text{diff}}$ can indicate the "realness" or the "fakeness" of a state-action pair to some extent, optimizing a policy using rewards with this wide value range $[0, \infty)$ can be difficult (Henderson et al., 2018). To devise a discriminative classifier with a bounded output range of $[0, 1]$ given the diffusion model's output $\mathcal{L}_{\text{diff}}$, we construct a *Diffusion Discriminative Classifier* $D_\phi : \mathcal{S} \times \mathcal{A} \in \mathbb{R}$ (as shown in Figure 2) that integrates $\mathcal{L}^+_{\text{diff}}$ and $\mathcal{L}^-_{\text{diff}}$ to compute the "realness" of a state-action pair within the bounded range:

$$D_\phi(\mathbf{s}, \mathbf{a}) = \frac{e^{-\mathcal{L}_{\text{diff}}(\mathbf{s},\mathbf{a},\mathbf{c}^+)}}{e^{-\mathcal{L}_{\text{diff}}(\mathbf{s},\mathbf{a},\mathbf{c}^+)} + e^{-\mathcal{L}_{\text{diff}}(\mathbf{s},\mathbf{a},\mathbf{c}^-)}} \tag{9}$$

$$= \frac{1}{1 + e^{-[\mathcal{L}_{\text{diff}}(\mathbf{s},\mathbf{a},\mathbf{c}^-) - \mathcal{L}_{\text{diff}}(\mathbf{s},\mathbf{a},\mathbf{c}^+)]}} = \sigma(\mathcal{L}_{\text{diff}}(\mathbf{s}, \mathbf{a}, \mathbf{c}^-) - \mathcal{L}_{\text{diff}}(\mathbf{s}, \mathbf{a}, \mathbf{c}^+)), \tag{10}$$

where $\sigma(x) = 1/(1 + e^{-x})$ denotes the sigmoid function. The design of our diffusion discriminative classifier aligns with the GAIL discriminator (Ho & Ermon, 2016), which is proven effective in providing learning signals to the policy.

Consequently, we can optimize our proposed D_ϕ as follows:

$$\max_{D\in\mathbb{R}^{\mathcal{S}\times\mathcal{A}}} \mathbb{E}_{(\mathbf{s},\mathbf{a})\in\tau_E}\left[\log(D_\phi(\mathbf{s}, \mathbf{a}))\right] + \mathbb{E}_{(\mathbf{s},\mathbf{a})\in\tau_i}\left[\log(1 - D_\phi(\mathbf{s}, \mathbf{a}))\right] \tag{11}$$

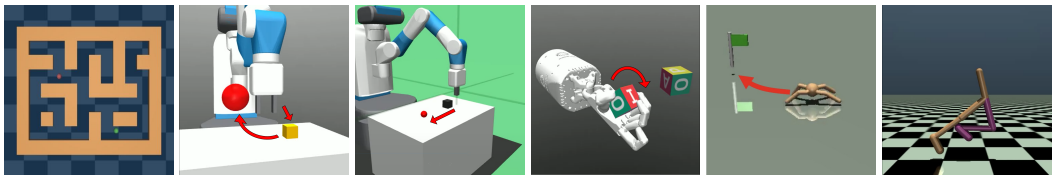
where τ_E and τ_i represent the expert trajectories and agent trajectories at training step i.

Intuitively, the discriminator D_ϕ is trained to predict a value closer to 1 when the input state-action pairs are sampled from expert demonstration (*i.e.*, trained to minimize $\mathcal{L}^+_{\text{diff}}$ and maximize $\mathcal{L}^-_{\text{diff}}$), and 0 if the input state-action pairs are obtained from the agent online interaction (*i.e.*, trained to minimize $\mathcal{L}^-_{\text{diff}}$ and maximize $\mathcal{L}^+_{\text{diff}}$).

Algorithm 1 Diffusion Rewards Guided Adversarial Imitation Learning (DRAIL)

1: **Input:** Expert trajectories τ_E, initial policy parameters θ_0, and initial diffusion discriminator parameters ϕ_0
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample agent trajectories $\tau_i \sim \pi_{\theta_i}$
4: Compute the output of diffusion discriminative classifier D_ϕ (Eq. 9) and the loss function $\mathcal{L}$ (Eq. 12)
5: Update the diffusion model $\phi_{i+1} \leftarrow \phi_i$ using $\nabla\mathcal{L}$
6: Compute the diffusion reward $\mathcal{R}_\phi(\mathbf{s}, \mathbf{a})$ with Eq. 13
7: Update the policy $\theta_{i+1} \leftarrow \theta_i$ with any RL algorithm with respect to reward $\mathcal{R}_\phi$
8: **end for**

[1]For simplicity, we will use the notations $\mathcal{L}^+_{\text{diff}}$ and $\mathcal{L}^-_{\text{diff}}$ to represent $\mathcal{L}_{\text{diff}}(\mathbf{s}, \mathbf{a}, \mathbf{c}^+)$ and $\mathcal{L}_{\text{diff}}(\mathbf{s}, \mathbf{a}, \mathbf{c}^-)$, respectively, in the rest of the paper.



(a) MAZE (b) FETCHPICK (c) FETCHPUSH (d) HANDROTATE (e) ANTREACH (f) WALKER

Figure 3: **Environments & Tasks. (a) MAZE**: A point-mass agent (green) within a 2D maze is trained to move from its initial position to reach the goal (red). **(b)-(c) FETCHPICK and FETCHPUSH**: The manipulation tasks is implemented with a 7-DoF Fetch robotics arm. FETCHPICK and FETCHPUSH require picking up or pushing an object to a target location (red). **(d) HANDROTATE**: For this dexterous manipulation task, a Shadow Dexterous Hand is employed to in-hand rotate a block to achieve a target orientation. **(e) ANTREACH**: This task is training a quadruped ant to reach a goal randomly positioned along the perimeter of a half-circle with a radius of 5 m. **(f) WALKER**: This locomotion task requires training a bipedal walker policy to achieve the highest possible walking speed while maintaining balance.

4.3 OVERALL ALGORITHM

Our proposed method adheres to the fundamental AIL framework, where the discriminator and policy are updated alternately. In the discriminator step, we compute the loss function $\mathcal{L}$ of the target function following Eq. 11:

$$\mathcal{L} = \mathbb{E}_{(\mathbf{s},\mathbf{a})\in\tau_E}\underbrace{[-\log(D_\phi(\mathbf{s},\mathbf{a}))]}_{\mathcal{L}_{BCE}^{\text{expert}}} + \mathbb{E}_{(\mathbf{s},\mathbf{a})\in\tau_i}\underbrace{[-\log(1-D_\phi(\mathbf{s},\mathbf{a}))]}_{\mathcal{L}_{BCE}^{\text{agent}}} \tag{12}$$

where $\mathcal{L}$ can be viewed as the sum of the expert binary cross-entropy loss $\mathcal{L}_{BCE}^{\text{expert}}$ and the agent binary cross-entropy loss $\mathcal{L}_{BCE}^{\text{agent}}$. We then update the diffusion discriminator parameters based on the gradient of $\mathcal{L}$.

In the policy step, we adopt the optimization objective proposed by Fu et al. (2018) as our diffusion reward signal for the policy to learn:

$$\mathcal{R}_\phi(\mathbf{s},\mathbf{a}) = \log(D_\phi(\mathbf{s},\mathbf{a})) - \log(1-D_\phi(\mathbf{s},\mathbf{a})). \tag{13}$$

We can optimize the policy using any RL algorithm to maximize the diffusion rewards provided by the diffusion discriminative classifier, bringing the policy closer to the expert policy. Specifically, we utilize PPO as our policy update algorithm. The algorithm is presented in Algorithm 1 and the overall framework is illustrated in Figure 2.

5 EXPERIMENTS

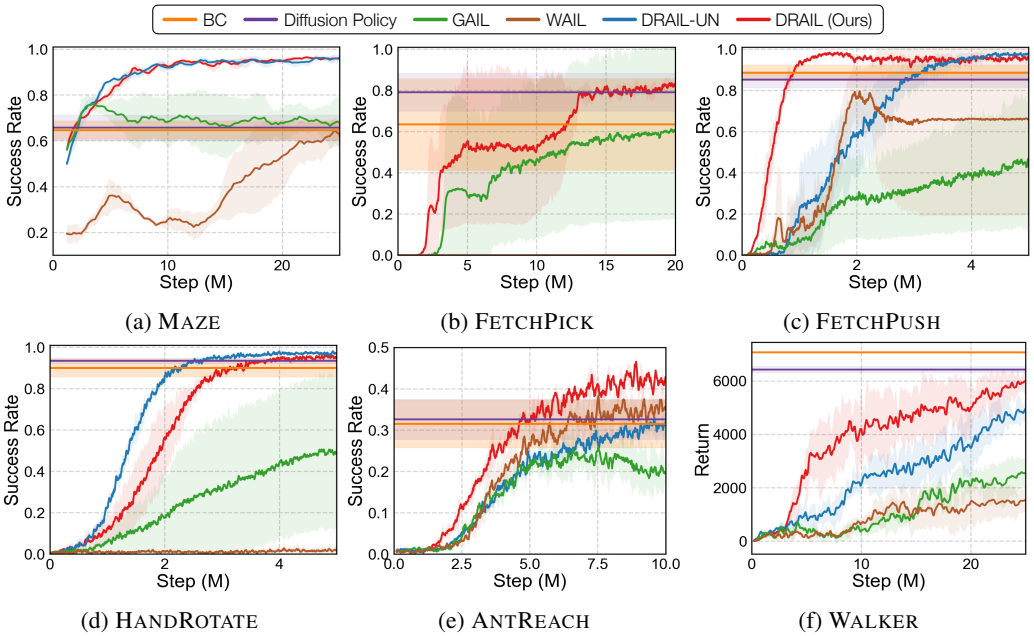
We extensively evaluate our proposed framework DRAIL in diverse continuous control domains, including navigation, robot arm manipulation, and locomotion. We also examine the generalizability and data efficiency of DRAIL in Section 5.4 and Section 5.5. The reward function learned by DRAIL is presented in Section 5.6.

5.1 EXPERIMENTAL SETUP

The environments, tasks, and expert demonstrations used in both the learning and evaluation phases are introduced in this section. Further details can be found in Section B.

MAZE. We evaluate our approach in the MAZE environment, which is introduced in Fu et al. (2020) (maze2d-medium-v2), as depicted in Figure 3a. In this task, a point-mass agent is trained to navigate from a randomly determined start location to the goal. The agent accomplishes the task by iteratively predicting its x and y acceleration. We utilize a controller to generate 100 demonstrations, comprising 18,525 transitions.

FETCHPICK and FETCHPUSH. Aiming at evaluating our approach in robot arm manipulation environments, we evaluate our approach in two 7-DoF Fetch tasks: FETCHPICK and FETCHPUSH, which are depicted in Figure 3b and Figure 3c, respectively. In FETCHPICK, the objective is to pick up an object from the table and lift it to a target location, while FETCHPUSH requires the arm to push



(a) MAZE (b) FETCHPICK (c) FETCHPUSH

(d) HANDROTATE (e) ANTREACH (f) WALKER

Figure 4: **Learning Efficiency.** We report success rates (MAZE, FETCHPICK, FETCHPUSH, HANDROTATE, ANTREACH) and return (WALKER), evaluated over three random seeds. Our method DRAIL learns more stably, faster, and achieves higher or competitive success rates than the best baseline over all environments.

an object to a designated location. The demonstrations utilized for these tasks are sourced from Lee et al. (2021). Each dataset consists of 10k transitions (303 trajectories for FETCHPICK and 185 trajectories for FETCHPUSH).

HANDROTATE. We further evaluate our approach in a challenging environment named HANDROTATE, which is introduced by Plappert et al. (2018). Here, a 24-DoF Shadow Dexterous Hand is tasked with learning to in-hand rotate a block to a target orientation, as depicted in Figure 3d. This environment features a high-dimensional state space (68D) and action space (20D). To generate our dataset, we collected 10k transitions (515 trajectories) using a SAC (Haarnoja et al., 2018) expert policy that underwent training for 10M environment steps.

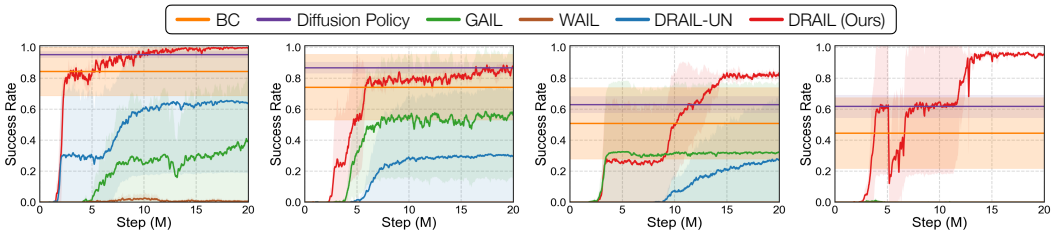
ANTREACH. The goal of ANTREACH is for a quadruped ant to reach a goal randomly positioned along the perimeter of a half-circle with a radius of 5 m, as depicted in Figure 3e. The 132D state representation includes joint angles, velocities, contact forces, and the goal position relative to the agent. We collect 1,000 demonstrations (25k transitions) using a pre-trained policy that underwent 40 million training steps.

WALKER. The objective of WALKER is to let a bipedal agent move at the highest speed possible while preserving its balance, as illustrated in Figure 3f. We utilize the demonstrations provided by Kostrikov (2018), which include 5 trajectories comprising 5k state-action pairs.

5.2 BASELINES AND VARIANTS

We compare our method DRAIL with the following baselines and variants of our approach.

- **Behavioral Cloning (BC)** trains a policy to mimic the actions of an expert by supervisedly learning a mapping from observed states to corresponding expert actions (Pomerleau, 1989; Torabi et al., 2018).
- **Diffusion Policy** represents a policy as a conditional diffusion model (Chi et al., 2023; Reuss et al., 2023). We implement this method according to Pearce et al. (2023). We include this baseline to compare learning a diffusion model as a *policy* (diffusion policy) or *reward function* (ours).



(a) FETCHPICK 1× (b) FETCHPICK 1.25× (c) FETCHPICK 1.75× (d) FETCHPICK 2.0×

Figure 5: **Generalization Experiments in FETCHPICK.** We present the performance of our proposed DRAIL and baselines in the FETCHPICK task, under varying levels of noise in initial states and goal locations. The evaluation spans three random seeds, and the training curve illustrates the success rate dynamics.

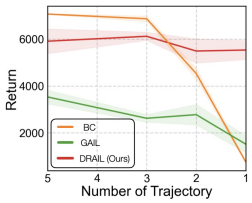
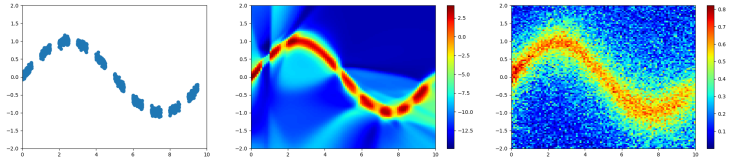


Figure 6: **Data Efficiency.** We compare BC, GAIL, and DRAIL using varying amounts of expert data in WALKER.



(a) Expert Distribution (b) GAIL Reward (c) DRAIL Reward

Figure 7: **Reward Function Visualization.** We present visualizations of the learned reward values by the discriminative classifier of GAIL and the diffusion discriminative classifier of our DRAIL. The target expert demonstration for imitation is depicted in **(a)**, which is a discontinuous sine function. The reward distributions of GAIL and our DRAIL are illustrated in **(b)** and **(c)**, respectively.

- **Generative Adversarial Imitation Learning (GAIL)** (Ho & Ermon, 2016) learns a policy from expert demonstrations by training a discriminator to distinguish between trajectories generated by the learned generator policy and those from expert demonstrations.
- **Wasserstein Adversarial Imitation Learning (WAIL)** extends GAIL by employing Wasserstein distance, aiming to capture smoother reward functions, proposed by Arjovsky et al. (2017).
- **DRAIL-Unnormalized (DRAIL-UN)** is a variant of DRAIL, which excludes the normalization term in Eq. 9. That said, we use $e^{-\mathcal{L}_{\text{diff}}} \in [0, 1]$ as the discrimination output. This variant resembles the idea proposed by Wang et al. (2023), a concurrent work that also integrates diffusion models into AIL.

5.3 EXPERIMENTAL RESULTS

We present the success rates (MAZE, FETCHPICK, FETCHPUSH, HANDROTATE, ANTREACH) and return (WALKER) of DRAIL and the baselines in Figure 4. Each task is trained with three different random seeds. More information on model architecture can be found in Section D and Section E describes training and evaluation details.

Overall, our method DRAIL consistently outperforms GAIL across all the environments, verifying the effectiveness of integrating our proposed diffusion discriminative classifier and diffusion rewards.

In MAZE, FETCHPUSH, and HANDROTATE tasks, our DRAIL performs competitively against our variant DRAIL-UN and outperforms the other baselines. These two diffusion model-based AIL algorithms exhibit superior performance compared to other baseline methods, particularly excelling in the environment of MAZE. In FETCHPICK, our DRAIL performs competitively against Diffusion Policy and outperforms the other baselines.

In the locomotion task, *i.e.*, WALKER. DRAIL surpasses all AIL baselines, yet under-performing compared to BC and Diffusion Policy. We hypothesize that WALKER requires less generalizability to unseen states and therefore BC can achieve the best performance with sufficient expert data. We discuss varying amounts of expert data in Section 5.5, which suggests that DRAIL outperforms BC when less expert data is available.

In the ANTREACH task, which blends locomotion and navigation, our method DRAIL also outperforms all baselines.

5.4 GENERALIZATION EXPERIMENTS

To examine the generalizability to states that are unseen from the expert demonstrations of different methods, we extend the FETCHPICK tasks following the setting proposed by Lee et al. (2021). Specifically, we evaluate policies learned by different methods by varying the noise injected into initiate states (*e.g.*, position and velocity of the robot arm) and the target block positions. We experiment with different noise levels, including $1\times$, $1.25\times$, $1.5\times$, $1.75\times$, and $2.0\times$, compared to the expert environment. That said, $1.5\times$ means the policy is evaluated in an environment with noises $1.5\times$ larger than those injected into expert data collection. Performing well in a high noise level setup requires the policy to generalize to unseen states.

The results of FETCHPICK under $1\times$, $1.25\times$, $1.75\times$, and $2.0\times$ noise level are presented in Figure 5. Across all noise levels, our proposed DRAIL outperforms all the baselines. In the $1.75\times$ noise level, DRAIL achieves a success rate of 83.97%, surpassing the best-performing baseline Diffusion Policy, which achieves only around a success rate of 62.93%. GAIL, on the other hand, experiences failure in 1 out of the 3 seeds, resulting in a high standard deviation (mean: 32.61, standard deviation: 56.49) despite our thorough exploration of various settings for its configuration.

We additionally conduct generalization experiments across all environments and present the results in Section A. Overall, our method DRAIL performs competitively against our variant DRAIL-UN and outperforms the other baselines, demonstrating our method's superior generalization ability.

5.5 DATA EFFICIENCY

To investigate the data efficiency of DRAIL, we vary the number of expert trajectories in WALKER and report the performance of BC, GAIL, and DRAIL. Specifically, we use 1, 2, 3, and 5 expert trajectories, each containing 1000 transitions, and present the results in Figure 6.

The result demonstrates the superior data efficiency of our method, maintaining a return value over 5000 even when trained with a single trajectory. In contrast, BC and GAIL suffer from significant performance drops when the amount of expert data is reduced.

5.6 REWARD FUNCTION VISUALIZATION

To analyze the learned reward functions, we design a SINE environment, where the expert state-action pairs form a discontinuous sine wave $a = \sin(20s\pi) + \mathcal{N}(0, 0.05)$, as shown in Figure 7a. We train GAIL and DRAIL to learn from this expert state-action distribution and visualize the discriminator output values D_ϕ to examine the learned reward function, as presented in Figure 7.

Figure 7b reveals that the GAIL discriminator exhibits excessive overfitting to the expert demonstration, rendering it ineffective in providing appropriate reward values when encountering unseen states. In contrast, Figure 7c shows that our proposed DRAIL generalizes better to the broader state-action distribution, yielding a more robust reward value, thereby enhancing the generalizability of learned policies. Furthermore, the predicted reward value of DRAIL gradually decreases as the state-action pairs deviate farther from the expert demonstration. This reward smoothness can guide the policy even when it deviates from the expert policy. In contrast, the reward distribution from GAIL is relatively narrow outside the expert demonstration, making it challenging to properly guide the policy if the predicted action does not align with the expert.

6 CONCLUSION

This work proposes a novel adversarial imitation learning framework that integrates a diffusion model into generative adversarial imitation learning. Specifically, we propose a diffusion discriminative classifier that employs a diffusion model to construct an enhanced discriminator, yielding more precise and smoother rewards. Then, we design diffusion rewards based on the classifier's output for policy learning. Extensive experiments in navigation, manipulation, and locomotion justify our proposed framework's effectiveness, sample efficiency, generalizability, and data efficiency.

BIBLIOGRAPHY

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of International Conference on Machine Learning*, 2004.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of International Conference on Machine Learning*, 2017.

Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando De Freitas. Playing hard exploration games by watching youtube. In *Proceedings of Neural Information Processing Systems*, 2018.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Proceedings of Neural Information Processing Systems*, 2017.

Robert Dadashi, Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal wasserstein imitation learning. *arXiv preprint arXiv:2006.04678*, 2020.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Proceedings of Neural Information Processing Systems*, 2021.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robotic Learning*, 2022.

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

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of Neural Information Processing Systems*, 2014.

Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois Knoll. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*, 2022.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of International Conference on Machine Learning*, 2018.

Jack Harmer, Linus Gisslén, Jorge del Val, Henrik Holst, Joakim Bergdahl, Tom Olsson, Kristoffer Sjöö, and Magnus Nordin. Imitation learning with concurrent actions in 3d games. In *IEEE Conference on Computational Intelligence and Games*, 2018.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2018.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of Neural Information Processing Systems*, 2020.

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys*, 2017.

Rohit Jena, Changliu Liu, and Katia Sycara. Augmenting gail with bc for sample efficient imitation learning. In *Conference on Robot Learning*, 2021.

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

B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. `https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail`, 2018.

Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *Proceedings of International Conference on Machine Learning*, 2019a.

Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *Proceedings of International Conference on Learning Representations*, 2019b.

Youngwoon Lee, Shao-Hua Sun, Sriram Somasundaram, Edward S. Hu, and Joseph J. Lim. Composing complex skills by learning transition policies. In *Proceedings of International Conference on Learning Representations*, 2019.

Youngwoon Lee, Andrew Szot, Shao-Hua Sun, and Joseph J. Lim. Generalizable imitation learning from observation via inferring goal proximity. In *Proceedings of Neural Information Processing Systems*, 2021.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

Abdoulaye O Ly and Moulay Akhloufi. Learning to drive by imitation: An overview of deep behavior cloning methods. *IEEE Transactions on Intelligent Vehicles*, 2020.

Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of International Conference on Machine Learning*, 2000.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of International Conference on Machine Learning*, 2021.

Manu Orsini, Anton Raichuk, Léonard Hussenot, Damien Vincent, Robert Dadashi, Sertan Girgin, Matthieu Geist, Olivier Bachem, Olivier Pietquin, and Marcin Andrychowicz. What matters for adversarial imitation learning? In *Proceedings of Neural Information Processing Systems*, 2021.

Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, Jan Peters, et al. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 2018.

Tim Pearce, Tabish Rashid, Anssi Kanervisto, David Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating human behaviour with diffusion models. In *Proceedings of International Conference on Learning Representations*, 2023.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Proceedings of Neural Information Processing Systems*, 1989.

Harish Ravichandar, Athanasios S Polydoros, Sonia Chernova, and Aude Billard. Recent advances in robot learning from demonstration. *Annual review of control, robotics, and autonomous systems*, 2020.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, 2011.

Stefan Schaal. Learning from demonstration. In *Proceedings of Neural Information Processing Systems*, 1997.

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

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of International Conference on Machine Learning*, 2015.

Gokul Swamy, David Wu, Sanjiban Choudhury, Drew Bagnell, and Steven Wu. Inverse reinforcement learning without reinforcement learning. In *Proceedings of International Conference on Machine Learning*, 2023.

Umar Syed, Michael Bowling, and Robert E Schapire. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*, 2008.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of International Joint Conference on Artificial Intelligence*, 2018.

Faraz Torabi, Garrett Warnell, and Peter Stone. Generative adversarial imitation from observation. In *Proceedings of International Conference on Machine Learning*, 2019.

Bingzheng Wang, Yan Zhang, Teng Pang, Guoqiang Wu, and Yilong Yin. Diffail: Diffusion adversarial imitation learning. *arXiv preprint arXiv:2312.06348*, 2023.

Hsiang-Chun Wang, Shang-Fu Chen, Ming-Hao Hsu, Chun-Mao Lai, and Shao-Hua Sun. Diffusion model-augmented behavioral cloning. In *ICML Workshop: New Frontiers in Learning, Control, and Dynamical Systems*, 2024.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2008.

Konrad Zolna, Scott Reed, Alexander Novikov, Sergio Gomez Colmenarejo, David Budden, Serkan Cabi, Misha Denil, Nando de Freitas, and Ziyu Wang. Task-relevant adversarial imitation learning. In *Conference on Robotic Learning*, 2021.

APPENDIX

Table of Contents

A FULL RESULTS OF GENERALIZATION EXPERIMENTS

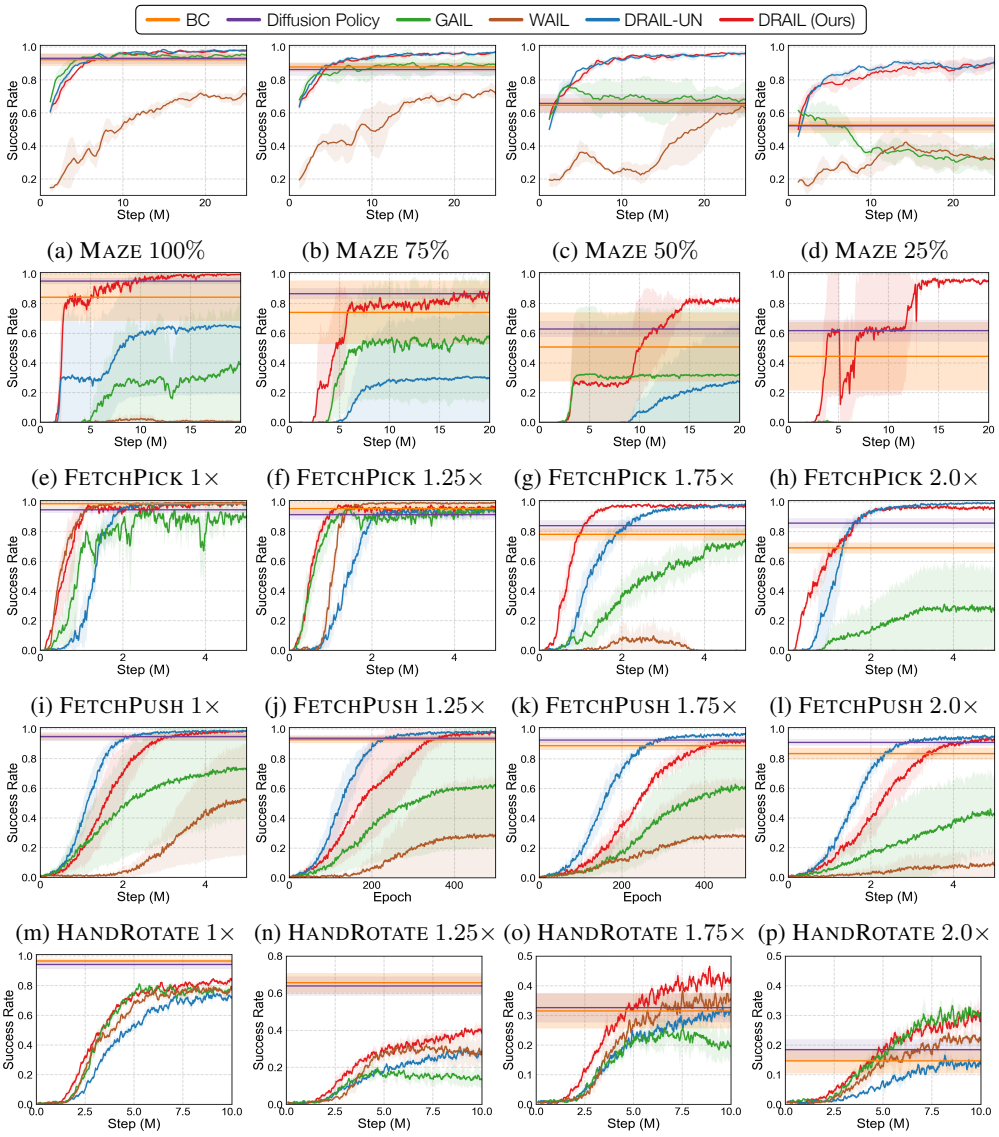
A.1 EXPERIMENT SETTINGS

To show our approach's better generalization capabilities, we extend the environment scenarios following the setting stated in Lee et al. (2021): (1) In MAZE main experiment, the initial and goal states of the expert dataset only constitute 50% of the potential initial and goal states. In the generalization experiment, we gather expert demonstrations from some lower and higher coverage: 25%, 75%, and 100%. (2) In FETCHPICK, FETCHPUSH, and HANDROTATE main experiments, the demonstrations are collected in a lower noise level setting, $1\times$. Yet, the agent is trained within an environment incorporating $1.5\times$ noise, which is 1.5 times larger noise than the collected expert demonstration, applied to the starting and target block positions. In the generalization experiment, we train agents in different noise levels: $1\times$, $1.25\times$, $1.5\times$, $1.75\times$, $2.0\times$. (3) In ANTREACH main experiment, 0.03 random noise is added to the initial pose during policy learning. In ANTREACH generalization experiment, we train agents in different noise levels: 0, 0.01, 0.05.

These generalization experiments simulate real-world conditions. For example, because of the expenses of demonstration collection, the demonstrations may inadequately cover the entire state space, as seen in setup (1). Similarly, in setups (2) and (3), demonstrations may be acquired under controlled conditions with minimal noise, whereas the agent operating in a real environment would face more significant noise variations not reflected in the demonstrations, resulting in a broader distribution of initial states.

A.2 EXPERIMENT RESULTS

MAZE. Our DRAIL outperforms baselines or performs competitively against DRAIL-UN across all demonstration coverages, as shown in Figure 8. Particularly, BC, WAIL, and GAIL's performance decline rapidly in the low coverage case. In contrast, diffusion model-based AIL algorithms demonstrate sustained performance, as shown in Figure 4a. This suggests that our method exhibits robust generalization, whereas BC and GAIL struggle with unseen states under limited demonstration coverage.



(a) MAZE 100% (b) MAZE 75% (c) MAZE 50% (d) MAZE 25%

(e) FETCHPICK 1× (f) FETCHPICK 1.25× (g) FETCHPICK 1.75× (h) FETCHPICK 2.0×

(i) FETCHPUSH 1× (j) FETCHPUSH 1.25× (k) FETCHPUSH 1.75× (l) FETCHPUSH 2.0×

(m) HANDROTATE 1× (n) HANDROTATE 1.25× (o) HANDROTATE 1.75× (p) HANDROTATE 2.0×

(q) ANTREACH 0.00 (r) ANTREACH 0.01 (s) ANTREACH 0.03 (t) ANTREACH 0.05

Figure 8: **Full Results of Generalization Experiments.** MAZE is evaluated with different coverages of state spaces in demonstrations, while FETCHPICK, FETCHPUSH, HANDROTATE, and ANTREACH environments are evaluated in environments of different noise levels. The number indicates the amount of additional noise in agent learning compared to that in the expert demonstrations, with more noise requiring harder generalization. The noise level rises from left to right.

FETCHPICK and FETCHPUSH. In FETCHPICK, our method outperforms all baselines across all noise levels, as shown in Figure 8. In the $1.75\times$ noise level, our method DRAIL achieves a success rate of 83.97, surpassing the best-performing baseline BC, which achieves only around 51%. GAIL, on the other hand, experiences failure in 1 out of the 3 seeds, resulting in a high standard deviation (mean: 32.61, standard deviation: 56.49) despite our thorough exploration of various settings for its configuration. In FETCHPUSH, our method DRAIL and its variant DRAIL-UN exhibit more robust results, in generalizing to unseen states compared to other baselines, as shown in Figure 8. This showcases that the diffusion reward guidance could provide better generalizability for the AIL framework. Moreover, our DRAIL is quite sample-efficient regarding interaction with the environment during training compared to other baselines and the variant in FETCHPICK and FETCHPUSH environment.

HANDROTATE. Our DRAIL outperforms baselines or performs competitively against DRAIL-UN across all noise levels, as illustrated in Figure 8. Specifically, our DRAIL and DRAIL-UN achieve a success rate of 95% at a noise level of 2.0, while GAIL and WAIL only reach approximately 42% and 8%, respectively. Notably, WAIL attains a success rate of 51% at a noise level of 1 but struggles to train as the noise level increases.

ANTREACH. Under a 0.03 noise level, the overall success rate of our method, the baselines, and the variant drop dramatically, as illustrated in Figure 4e. Despite this, our DRAIL maintains a higher success rate of 42%, surpassing other baselines that achieve success rates lower than 40%.

B ENVIRONMENT & TASK DETAILS

B.1 MAZE

Description. In a 2D maze environment, a point-maze agent learns to navigate from a starting location to a goal location. The agent achieves this by iteratively predicting its x and y velocity. The initial and final positions of the agent are randomly selected. The state space includes position, velocity, and goal position. The maximum episode length for this task is set at 400, and the episode terminates if the goal is reached earlier.

Expert Dataset. The expert dataset comprises 100 demonstrations, which includes $18,525$ transitions provided by Lee et al. (2021).

B.2 FETCHPUSH & FETCHPICK

Description. In the FETCHPUSH task, the agent is required to push an object to a specified target location. On the other hand, in the FETCHPICK task, the objective is to pick up an object from a table and move it to a target location.

According to the environment setups stated in Lee et al. (2021), the 16-dimensional state representation includes the angles of the robot joints, and the initial three dimensions of the action vector represent the intended relative position for the next time step. The first three dimensions of the action vector denote the intended relative position in the subsequent time step. In the case of FETCHPICK, an extra action dimension is incorporated to specify the distance between the two fingers of the gripper. The maximum episode length for this task is set at 50 for FETCHPICK and 60 for FETCHPUSH, and the episode terminates if the agent reaches the goal earlier.

Expert Dataset. The expert dataset for FETCHPICK comprises 303 trajectories, amounting to $10,000$ transitions provided by Lee et al. (2021).

B.3 HANDROTATE

Description. In the task HANDROTATE proposed by Plappert et al. (2018), a 24-DoF Shadow Dexterous Hand is designed to rotate a block in-hand to a specified target orientation. The 68D state representation includes the agent's joint angles, hand velocities, object poses, and target rotation. The 20D action vector represents the joint torque control of the 20 joints. Notably, HANDROTATE is challenging due to its high-dimensional state and action spaces. We follow the experimental setup outlined in Plappert et al. (2018) and Lee et al. (2021), where rotation is constrained to the z-axis, and

allowable initial and target z rotations are within $[-\frac{\pi}{12}, \frac{\pi}{12}]$ and $[\frac{\pi}{3}, \frac{2\pi}{3}]$, respectively. The maximum episode length for this task is set at 50, and the episode terminates if the hand reaches the goal earlier.

Expert Dataset. Aiming at collecting expert demonstrations, we train a SAC (Haarnoja et al., 2018) policy utilizing dense rewards for $10M$ environment steps. The dense reward assigned at each time step t is $R(s_t, a_t) = d_t - d_{t+1}$, where d_t and d_{t+1} denotes the angles (in radians) between the current and desired block orientations before and after executing the actions. Following the training phase, the SAC expert policy attains a success rate of 59.48%. Subsequently, we extract 515 successful trajectories (equivalent to 10k transitions) from this policy to construct our expert dataset for the HANDROTATE task.

B.4 ANTREACH

Description. The ANTREACH task features a four-leg ant robot reaching a randomly assigned target position located within a range of half-circle with a radius of 5 meters. The task's state is represented by a 132-dimension vector, including joint angles, velocities, and the relative position of the ant towards the goal. Expert data collection for this task is devoid of any added noise, while random noise is introduced during the training and inference phases. Consequently, the policy is required to learn to generalize to states not present in the expert demonstrations. The maximum episode length for this task is set at 50, and the episode terminates if the ant reaches the goal earlier.

Expert Dataset. The expert dataset comprises 10000 state-action pairs provided by Lee et al. (2021).

B.5 WALKER

Description. WALKER task involves guiding an agent to move towards the x-coordinate as fast as possible while maintaining balance. An episode terminates either when the agent experiences predefined unhealthy conditions in the environment or when the maximum episode length (1000) is reached. The agent's performance is evaluated over 100 episodes with three different random seeds. The return of an episode is the cumulative result of all timesteps within that episode. The 17D state includes joint angles, angular velocities of joints, and velocities of the x and z-coordinates of the top. The 6D action defines the torques that need to be applied to each joint of the walker avatar.

Expert Dataset. The expert dataset consists of 5 trajectories with $5k$ state-action pairs provided by Kostrikov (2018)

C SAMPLE EFFICIENCY AND DATA EFFICIENCY

To illustrate that utilizing the diffusion reward in our DRAIL provides better sample efficiency in terms of interacting with the environment and data efficiency in terms of expert demonstration compared to a simple MLP, we present the full training curve in (Figure 9). This experiment involves training our DRAIL, GAIL, and BC in the WALKER environment with different numbers of expert trajectories, as detailed in Section 5.5.

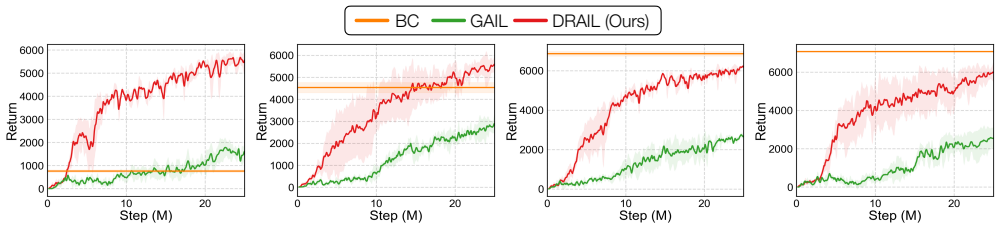
The results demonstrate that our DRAIL learns faster compared to the AIL baseline (GAIL), indicating superior sample efficiency in terms of environment interaction. Moreover, our method exhibits robustness when reducing the trajectory number from 5 to 1, with performance only decreasing from approximately 6000 to 5500. In contrast, BC's performance experiences a dramatic drop from 7000 to lower than 1000. We attribute GAIL's poor performance to sample inefficiency rather than the amount of expert data.

D MODEL ARCHITECTURE

This section presents the model architecture of all the experiments. Appendix D.1 describe the model architecture of all methods used in Section 5.3.

Table 1: **Model Architectures.** We report the architectures used for all the methods on all the tasks.

Method	Models	Component	MAZE	FETCHPICK	FETCHPUSH	HANDROTATE	WALKER	ANTREACH
BC	Policy π	# Layers	3	4	3	4	3	3
		Input Dim.	6	16	16	68	17	132
		Hidden Dim.	256	256	256	512	256	256
		Output Dim.	2	4	3	20	6	8
Diffusion Policy	Policy π	# Layers	5	5	5	5	7	6
		Input Dim.	8	20	19	88	23	140
		Hidden Dim.	256	1200	1200	2100	1024	1200
		Output Dim.	2	4	3	20	6	8
GAIL	Discriminator D	# Layers	3	4	5	4	5	5
		Input Dim.	8	20	19	88	23	140
		Hidden Dim.	64	64	64	128	64	64
		Output Dim.	1	1	1	1	1	1
	Policy π	# Layers	3	3	3	3	3	3
		Input Dim.	6	16	16	68	17	132
		Hidden Dim.	64	64	256	64	256	256
		Output Dim.	2	4	3	20	6	8
WAIL	Discriminator D	# Layers	3	4	5	4	5	5
		Input Dim.	8	20	19	88	23	140
		Hidden Dim.	64	64	64	128	64	64
		Output Dim.	1	1	1	1	1	1
		Reg. Value ϵ	0	0	0.01	0	0.1	0.01
	Policy π	# Layers	3	3	3	3	3	3
		Input Dim.	6	16	16	68	17	132
		Hidden Dim.	64	64	256	64	256	256
		Output Dim.	2	4	3	20	6	8
DRAIL (Ours)	Diffusion Model ϕ	# Layers	5	4	5	3	5	5
		Input Dim.	18	30	29	98	33	150
		Hidden Dim.	128	128	1024	128	1024	1024
		Output Dim.	8	20	19	88	23	140
		Label Dim. $\lvert\mathbf{c}\rvert$	10	10	10	10	10	10
	Policy π	# Layers	3	3	3	3	3	3
		Input Dim.	6	16	16	68	17	132
		Hidden Dim.	64	64	256	64	256	256
		Output Dim.	2	4	3	20	6	8
DRAIL-UN	Diffusion Model ϕ	# Layers	5	4	5	3	5	5
		Input Dim.	8	20	19	88	23	140
		Hidden Dim.	128	128	1024	128	1024	1024
		Output Dim.	8	20	19	88	23	140
	Policy π	# Layers	3	3	3	3	3	3
		Input Dim.	6	16	16	68	17	132
		Hidden Dim.	64	64	256	64	256	256
		Output Dim.	2	4	3	20	6	8



(a) WALKER 1 trajectory (b) WALKER 2 trajectories (c) WALKER 3 trajectories (d) WALKER 5 trajectories

Figure 9: **Sample Efficiency and Data Efficiency.** We report the full training curves of BC, GAIL, and DRAIL using varying amounts of expert data in WALKER. Our DRAIL demonstrates superior sample efficiency in terms of environment interaction and data efficiency in terms of expert demonstration.

D.1 MODEL ARCHITECTURE OF DRAIL, DRAIL-UN, AND THE BASELINES

In Section 5.3, we conducted a comparative analysis between our proposed DRAIL and its variant, DRAIL-UN, along with several baseline approaches (BC, Diffusion Policy, GAIL, WAIL) across six diverse environments. We applied Multilayer Perceptron (MLP) for the policy of BC, the conditional diffusion model in Diffusion Policy, as well as the policy and the discriminator of GAIL and WAIL. For our proposed DRAIL and DRAIL-UN, MLPs were employed to the policy and diffusion model of the diffusion discriminative classifier. The activation functions used for the MLPs in the diffusion model were ReLU, while hyperbolic tangent was employed for the others. The total timestep T for all diffusion models in this paper is set to 1000 and the scheduler used for diffusion models is cosine scheduler (Nichol & Dhariwal, 2021). Further details regarding the parameters for the model architecture can be found in Table 1.

BC. We maintained a concise model for the policy of BC to prevent excessive overfitting to expert demonstrations. This precaution is taken to mitigate the potential adverse effects on performance when confronted with environments exhibiting higher levels of noise.

Diffusion Policy. Based on empirical results and Wang et al. (2024), the Diffusion Policy performs better when implemented with a deeper architecture. Consequently, we have chosen to set the policy's number of layers to 5.

GAIL. The detailed model architecture for GAIL is presented in Table 1.

WAIL. We set the ground transport cost and the type of regularization of WAIL as Euclidean distance and L_2-regularization. The regularization value ϵ is provided in Table 1.

DRAIL. The conditional diffusion model of the diffusion discriminative classifier in our DRAIL is constructed by concatenating either the real label $\mathbf{c}^+$ or the fake label $\mathbf{c}^-$ to the noisy state-action pairs as the input. The model then outputs the predicted noise applied to the state-action pairs. The dimensions of both $\mathbf{c}^+$ and $\mathbf{c}^-$ are reported in Table 1.

DRAIL-UN. In DRAIL-UN, the conditional diffusion model is not utilized as it only needs to consider the numerator of Equation (9). Consequently, the diffusion model takes only the noisy state-action pairs as input and outputs the predicted noise value.

E TRAINING DETAILS

E.1 TRAINING HYPERPARAMTERS

The hyperparameters employed for all methods across various tasks are outlined in Table 2. The Adam optimizer (Kingma & Ba, 2015) is utilized for all methods, with the exception of the discriminator in WAIL, for which RMSProp is employed. Linear learning rate decay is applied to all policy models.

Due to the potential impact of changing noise levels on the quality of agent data input for the discriminator, the delicate balance between the discriminator and the AIL method's policy may be

Table 2: **Hyperparameters.** This table presents the overview of the hyperparameters used for all the methods across various tasks.

Method	Hyperparameter	MAZE	FETCHPICK	FETCHPUSH	HANDROTATE	WALKER	ANTREACH
BC	Learning Rate	0.00005	0.0008	0.0002	0.0001	0.0001	0.001
	Batch Size	128	128	128	128	128	128
	# Epochs	2000	1000	1000	5000	1000	1000
Diffusion Policy	Learning Rate	0.0002	0.00001	0.0001	0.0001	0.0001	0.00001
	Batch Size	128	128	128	128	128	128
	# Epochs	20000	20000	10000	2000	5000	10000
GAIL	Discriminator Learning Rate	0.001	0.00001	0.000008	0.0001	0.0000005	0.0001
	Policy Learning Rate	0.0001	0.00005	0.0002	0.0001	0.0001	0.0001
	Environment Step	25000000	25000000	5000000	5000000	25000000	10000000
WAIL	Discriminator Learning Rate	0.00001	0.0001	0.00008	0.0001	0.0000008	0.00001
	Policy Learning Rate	0.00001	0.0005	0.0001	0.0001	0.0001	0.0001
	Environment Step	25000000	25000000	5000000	5000000	25000000	10000000
DRAIL (Ours)	Discriminator Learning Rate	0.001	0.0001	0.001	0.0001	0.0002	0.001
	Policy Learning Rate	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
	Environment Step	25000000	25000000	5000000	5000000	25000000	10000000
DRAIL-UN	Discriminator Learning Rate	0.001	0.0001	0.0001	0.0001	0.0001	0.0001
	Policy Learning Rate	0.0001	0.0001	0.00005	0.0001	0.0001	0.0001
	Environment Step	25000000	25000000	5000000	5000000	25000000	10000000

Table 3: **PPO training parameters.** This table reports the PPO training hyperparameters used for each task.

Hyperparameter	MAZE	FETCHPICK	FETCHPUSH	HANDROTATE	WALKER	ANTREACH
Clipping Range ϵ	0.2	0.2	0.2	0.2	0.2	0.2
Discount Factor γ	0.99	0.99	0.99	0.99	0.99	0.99
GAE Parameter λ	0.95	0.95	0.95	0.95	0.95	0.95
Value Function Coefficient	0.5	0.5	0.5	0.5	0.5	0.5
Entropy Coefficient	0.0001	0.0001	0.001	0.0001	0.001	0.001

disrupted. Therefore, we slightly adjusted the learning rate for the policy and the discriminator for different noise levels on each task. The reported parameters in Table 2 correspond to the noise levels presented in Figure 4.

E.2 REWARD FUNCTION DETAILS

As explained in Section 4.3, we adopt the optimization objective proposed by (Fu et al., 2018) as diffusion reward signal for the policy learning in our DRAIL. To maintain fairness in comparisons, we apply the same reward function to our variant DRAIL-UN and GAIL. For WAIL, we adhere to the approach outlined in the original paper, wherein the output of the discriminator directly serves as the reward function. For each environment, we set a consistent value for the environment interaction step across all AIL approaches.

In our experiments, we employ Proximal Policy Optimization (PPO) (Schulman et al., 2017), a widely used policy optimization method. We maintain all hyperparameters of PPO constant across methods for a given task, except the learning rate, which is adjusted for each method. The PPO hyperparameters for each task are presented in Table 3.