2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

EDAIL: Adversarial Imitation Learning via Exploration-Driven Data Augmentation

Pengcheng Li[1], Qiang Fang[1,*], Xin Xu[1]

***Abstract*— Adversarial Imitation Learning (AIL) is a prominent paradigm in imitation learning that enables policy acquisition from expert demonstrations without relying on manually crafted reward functions. Although AIL has achieved promising results in certain scenarios, many existing methods suffer from mode collapse and training instability when expert demonstrations are limited. Given that agent–environment interactions are often abundant, we focus on effectively leveraging such interaction data to address the above challenges. In this paper, we propose a novel adversarial imitation learning framework called Exploration-Driven Adversarial Imitation Learning (EDAIL). First, we introduce exploratory policies that augment the discriminator's training data with high-confidence state-action pairs generated by the agent, thereby improving coverage of the solution space under sparse expert data. Second, we design an asymmetric surrogate reward function that shifts the reward-penalty boundary to mitigate discriminator bias caused by class imbalance, enabling more reliable policy optimization. We evaluate our method on six simulated tasks, including robotic manipulation, locomotion, and navigation, using only 1% and 10% of the datasets employed in prior baselines as expert demonstrations. Experimental results show that our method outperforms the baselines, demonstrating both the effectiveness and robustness of our method. In particular, it achieves a success rate of 84.67% on the FetchPush task using only 1% of expert demonstrations, representing an absolute improvement of 19.27 points over the state-of-the-art method. Our code will be available at `https://github.com/lipengcheng-nudt/EDAIL`.**

I. INTRODUCTION

In recent years, Reinforcement Learning (RL) has made remarkable advances in domains such as robotics [1], autonomous driving [2], and Unmanned Aerial Vehicles (UAVs) [3]. RL typically optimizes policies by maximizing a manually designed reward function. However, in complex environments, designing an informative and efficient reward function can be extremely challenging, often requiring extensive domain knowledge and fine-tuning.

Imitation Learning (IL) provides an alternative approach, aiming to learn policies directly from expert demonstrations, thereby bypassing the need for explicit reward specification [4], [5], [6]. Among IL methods, Adversarial Imitation Learning (AIL) [7], [8] has emerged as a dominant class of algorithms. AIL frameworks generally consist of two components: a discriminator that distinguishes expert behavior from agent behavior to infer a reward signal, and a policy network that learns by maximizing this inferred reward. Recent efforts have enhanced AIL by modifying reward structures [7], incorporating diffusion processes [9], or improving similarity metrics [10].

Despite these advances, the learned policy may suffer from poor generalization due to narrow coverage of the solution space when limited expert demonstrations are available. This can cause the policy to learn only a subset of the expert's behaviors, potentially leading to mode collapse. Additionally, class imbalance between expert and agent trajectories can bias the discriminator toward negative samples, resulting in unreliable surrogate rewards, which may cause suboptimal policy updates and training instability.

To alleviate these issues, we propose a novel AIL framework named Exploration-Driven Adversarial Imitation Learning (EDAIL). Specifically, during the training process, the agent obtains a considerable amount of agent-generated data through interactions with the environment. Instead of treating all the agent data as negative samples, we select a subset of high-confidence state-action pairs from this data as exploratory policies to supplement expert demonstrations for training the discriminator. This helps mitigate the limitations of sparse demonstrations and improves the representation of the feasible behavior space. Moreover, we introduce an asymmetric surrogate reward function that shifts the reward–penalty boundary, thereby compensating for the discriminator's bias caused by class imbalance and providing more accurate guidance for policy optimization.

In summary, our contributions are as follows:

- We propose exploratory policies to supplement expert demonstrations, which provide the discriminator with more diverse and comprehensive training data and thus enhance its generalization ability.
- We introduce an asymmetric surrogate reward function that allows accurate policy optimization in the presence of discriminator bias.
- By combining the above two components, we propose EDAIL and validate it across six benchmark tasks. Our method outperforms the baselines in terms of success rate and average return, achieving both faster convergence and smaller variance.

II. RELATED WORK

A. Behavior Cloning

Behavior Cloning (BC) is a foundational imitation learning approach that directly mimics expert demonstrations via supervised learning [4]. DAgger [11] improves upon BC by

This work was supported by the National Natural Science Foundation of China (62533021).

[1]College of Intelligence Science and Technology, National University of Defense Technology, Changsha, Hunan, China. lipengcheng24a@nudt.edu.cn, qiangfang@nudt.edu.cn, xinxu@nudt.edu.cn
*corresponding author

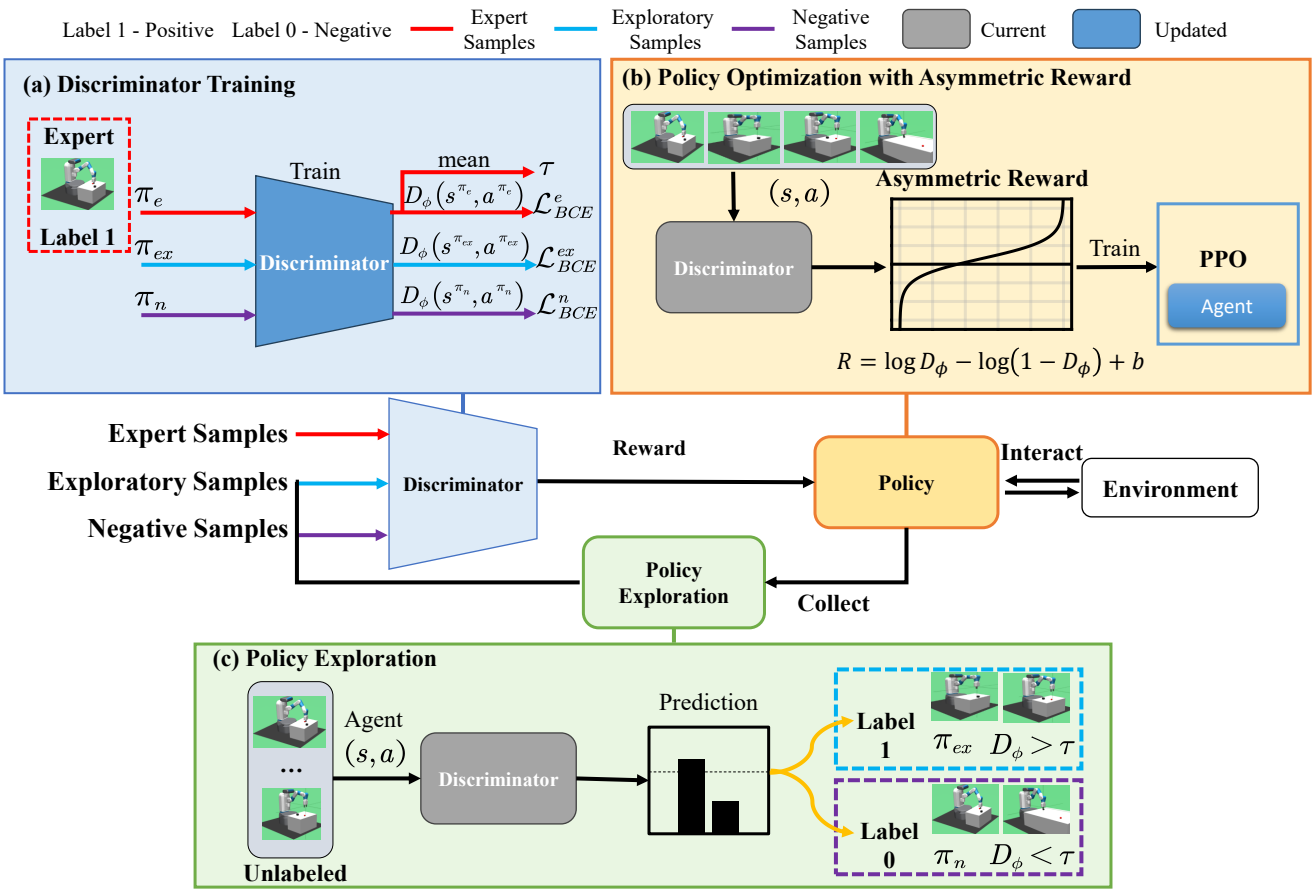


Fig. 1. **EDAIL.** Overview of our proposed pipeline. (a) In discriminator training, both expert and exploratory samples are used as positive samples. (b) The agent interacts with the environment to collect state–action pairs. An asymmetric surrogate reward function then computes rewards based on the discriminator's output, and the PPO algorithm is used to maximize these rewards. (c) The collected state–action pairs are categorized into exploratory and negative samples based on the discriminator's confidence scores, and used in the next iteration of discriminator training.

querying the expert for actions when the agent encounters unseen states during rollouts. Diffusion Policy [12], [13] reformulates policy learning as a conditional diffusion process, generating action sequences conditioned on observed states. BESO [14] adopts a score-based diffusion model to generate policies with fewer denoising steps. Other works focus on learning from imperfect demonstrations through importance-sampling-based techniques [15] and self-supervised methods [16]. While simple and computationally efficient, BC suffers from compounding errors when encountering states not covered in expert demonstrations [4].

B. Inverse Reinforcement Learning

Inverse Reinforcement Learning (IRL) [17], aims to infer the expert's goals or reward function from observed behaviors, rather than directly learning a policy. MEIRL [18] models the demonstrated behavior as a probability distribution over expert trajectories, constrained by the maximum entropy principle, to recover a nonlinear reward function. f-IRL [19] learns the reward function by leveraging the gradient of the f-divergence, enabling the recovery of a stable reward from the expert state distribution. AGA-MEIRL [20] learns reward functions using updated mixed expert demonstrations to address the issue of gradient explosion. However, different reward functions may induce the same agent behavior. To resolve this, some IRL approaches [21], [22] typically impose additional constraints on the reward function or policy to ensure uniqueness, which may in turn limit the generalization ability of the learned policy.

AIL, as one of the approaches within IRL, learns policies in an adversarial manner. GAIL [6] employs a discriminator to infer a reward function from expert demonstrations, which then guides the generator in learning policies via RL. AIRL [7] learns more generalizable policies by modifying the form of the surrogate reward function. DiffAIL [9] and DRAIL [23] enhance the discriminator by leveraging the distribution modeling capability of diffusion models. Other efforts focus on refining the similarity metrics between expert and agent trajectories, such as the Wasserstein distance [24] and the f-divergence [25], with the goal of improving discriminator training stability and sample efficiency. However, mode collapse and training instability are common issues in AIL. To address these issues, SMLING [26] proposes a non-adversarial inverse reinforcement learning approach by learning a score function. Unlike AIL, which alternates between training the discriminator and the policy, SMLING

updates the score function and the policy separately during training, which may prevent both from reaching optimal performance simultaneously. Our method falls within the domain of AIL, and mitigates mode collapse and training instability through exploratory policies and an asymmetric reward function.

III. METHOD

In this section, we present our proposed method. As shown in Fig. 1, we first describe how exploratory policies are leveraged to complement expert policies, enabling learning from a limited number of expert demonstrations. We then introduce a surrogate reward function designed for class imbalance, which provides more accurate guidance signals for forward RL.

A. Preliminaries on Adversarial Imitation Learning

The objective of AIL is to learn an optimal policy π_θ such that its state-action distribution closely matches that of the expert policy π_E. AIL consists of two core components: a discriminator (D_ϕ) and a generator. The discriminator is typically a binary classifier trained to distinguish between state-action pairs originating from the expert policy π_E and those generated by the agent policy π_θ. The generator, often instantiated as a RL algorithm, optimizes the policy by maximizing a surrogate reward function provided by the discriminator. This RL component can be an off-policy algorithm such as SAC [27], or an on-policy algorithm like PPO [28]. Formally, the objective can be written as:

$$\begin{aligned}\min_\theta \max_\phi & \mathbb{E}_{x\sim\rho_{\pi_\theta}}[\log D_\phi(x)] \\ & + \mathbb{E}_{x\sim\rho_{\pi_e}}[\log(1 - D_\phi(x))].\end{aligned} \tag{1}$$

Unlike traditional reinforcement learning with predefined reward functions, AIL provides a surrogate reward function by measuring the difference between the learned policy and the expert policy. In practice, the surrogate reward in GAIL is often defined as $R(s,a) = \log D_\phi(s,a)$ or $R(s,a) = -\log\big(1 - D_\phi(s,a)\big)$, both of which aim to minimize the JS divergence. A more common and effective approach is to optimize the KL divergence using:

$$R(s,a) = \log D_\phi(s,a) - \log\big(1 - D_\phi(s,a)\big). \tag{2}$$

where the surrogate reward gives a reward to policies with a discriminator confidence greater than 0.5, and applies a penalty to those with confidence below 0.5.

B. Learning from the Exploratory Policies

In traditional AIL, all state–action pairs obtained through RL interaction with the environment are treated as negative samples during discriminator training. However, We argue that even if the current policy fails to accomplish the overall task objective, some of its behaviors may still be correct. For example, in AntReach, a quadruped robot may fail to reach the target location using the current policy, yet still exhibit a correct walking posture.

During policy optimization, the agent continually interacts with the environment, generating a sequence of state–action pairs. The discriminator D_ϕ assigns each pair a confidence score indicating the likelihood of it being an expert-level behavior. To exploit potentially useful behaviors produced by the agent policy π_θ, we retain those pairs with confidence scores exceeding a threshold τ and categorize them as part of the exploratory policy π_{ex}, defined as:

$$\pi_{\text{ex}} = \{(s,a) \mid D_\phi(s^{\pi_\theta}, a^{\pi_\theta}) > \tau\}. \tag{3}$$

where $(s^{\pi_\theta}, a^{\pi_\theta})$ denotes a state–action pair generated by the agent's policy, and D_ϕ represents the output of the discriminator. The confidence threshold τ is set to the mean discriminator output over expert demonstrations:

$$\tau = \text{mean}\left[D_\phi(s^{\pi_e}, a^{\pi_e})\right]. \tag{4}$$

In the early training stages, a relatively lower threshold allows more samples to participate in training, thereby accelerating learning. To avoid excessively low initial thresholds, we apply a lower bound via clipping. As training progresses, a higher threshold is automatically adopted, prioritizing the selection of higher-quality samples. State–action pairs with confidence below τ are classified as negative samples π_n, rather than labeling all agent-generated samples as negative.

In subsequent discriminator updates, both exploratory policies (π_{ex}) and expert policies (π_e) are treated as positive samples. Accordingly, the discriminator loss is modified as:

$$\begin{aligned}\mathcal{L}(\phi) = & \mathcal{L}_{BCE}(-\log(D_\phi(s^{\pi_{ex}}, a^{\pi_{ex}}))) \\ & + \mathcal{L}_{BCE}(-\log(D_\phi(s^{\pi_e}, a^{\pi_e}))) \\ & + \mathcal{L}_{BCE}(-\log(1 - D_\phi(s^{\pi_n}, a^{\pi_n}))).\end{aligned} \tag{5}$$

where $\mathcal{L}_{\text{BCE}}$ denotes the binary cross-entropy loss.

C. Asymmetric Surrogate Reward Function

When expert demonstrations are scarce, the discriminator tends to be biased toward negative samples due to class imbalance, leading the surrogate reward to assign penalties to expert-like behaviors incorrectly. To alleviate this issue, we design an asymmetric surrogate reward function that shifts the reward–penalty decision boundary.

In DRAIL, the discriminator output is computed as:

$$D_\phi(s,a) = \frac{e^{-\mathcal{L}_{\text{diff}}(s,a,c^+)}}{e^{-\mathcal{L}_{\text{diff}}(s,a,c^+)} + e^{-\mathcal{L}_{\text{diff}}(s,a,c^-)}}. \tag{6}$$

where $-\mathcal{L}_{\text{diff}}(s,a,c^+)$ and $-\mathcal{L}_{\text{diff}}(s,a,c^-)$ denote the negative diffusion losses that measure the similarity between (s,a) and expert policies or non-expert policies, respectively. For brevity, we denote them as $\mathcal{L}^+_{\text{diff}}$ and $\mathcal{L}^-_{\text{diff}}$, respectively. The original DRAIL surrogate reward is defined as $R(s,a) = \log\big(D_\phi(s,a)\big) - \log\big(1 - D_\phi(s,a)\big)$. A positive reward ($R > 0$) occurs when:

$$-\mathcal{L}^+_{\text{diff}} > -\mathcal{L}^-_{\text{diff}}. \tag{7}$$

indicating that (s,a) is more similar to the expert policies and should therefore be rewarded.

However, due to the imbalance between negative samples and expert demonstrations,the discriminator tends to overfit to the limited expert data. Consequently, for an expert-level (s,a) not present in the training set, the discriminator may

still produce a correct $-\mathcal{L}^{-}_{\text{diff}}$ but underestimate $-\mathcal{L}^{+}_{\text{diff}}$. This would cause $-\mathcal{L}^{+}_{\text{diff}} < -\mathcal{L}^{-}_{\text{diff}}$ leading to the expert-like pair being incorrectly penalized.

To mitigate this issue, we introduce an asymmetric surrogate reward with a bias term:

$$R(s,a) = \log D_\phi(s,a) - \log\big(1 - D_\phi(s,a)\big) + b. \tag{8}$$

where b is a tunable hyperparameter that controls the decision boundary shift. This modification preserves the relative reward scaling while altering the threshold between reward and penalty.

The new decision boundary for assigning a positive reward is given by:

$$\log D_\phi(s,a) + b > \log\big(1 - D_\phi(s,a)\big), \tag{9}$$

Substituting the discriminator's probabilistic form, we have:

$$\frac{e^{-\mathcal{L}^{+}_{\text{diff}}} \cdot e^{b}}{e^{-\mathcal{L}^{+}_{\text{diff}}} + e^{-\mathcal{L}^{-}_{\text{diff}}}} > \frac{e^{-\mathcal{L}^{-}_{\text{diff}}}}{e^{-\mathcal{L}^{+}_{\text{diff}}} + e^{-\mathcal{L}^{-}_{\text{diff}}}}, \tag{10}$$

This condition is equivalent to:

$$-\mathcal{L}^{+}_{\text{diff}} + b > -\mathcal{L}^{-}_{\text{diff}}. \tag{11}$$

In practice, this is equivalent to replacing $-\mathcal{L}^{+}_{\text{diff}}$ with $-\mathcal{L}^{+}_{\text{diff}} + b$ in order to adjust the similarity estimation of (s,a) to the expert policies. This adjustment partially corrects the discriminator's overfitting bias and provides more reliable guidance for RL towards expert-like behaviors.

The overall method is also summarized in Algorithm 1.

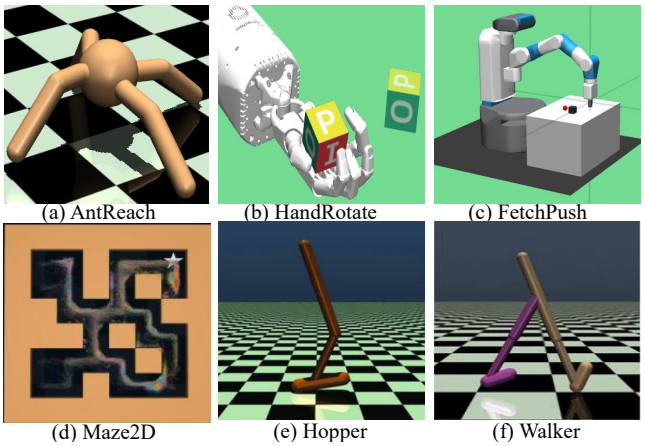
Algorithm 1 EDAIL

Input: expert demonstrations π_e, agent policy π_θ discriminator parameters ϕ, discriminator learning rate η_ϕ, total time steps T.

1: **for** $t = 0, 1, 2, \ldots, T$ **do**
2: Sample negative samples $(s^{\pi_n}, a^{\pi_n}) \sim \pi_n$
3: Sample from expert policies $(s^{\pi_e}, a^{\pi_e}) \sim \pi_e$ and exploratory policies $(s^{\pi_{ex}}, a^{\pi_{ex}}) \sim \pi_{ex}$
4: Calculate the output of the discriminator D_ϕ and compute the loss in (5).
5: Update the discriminator: $\phi \leftarrow \phi - \eta_\phi \nabla \mathcal{L}$ via gradient descent.
6: Collect state-action pairs (s,a) by interacting with the environment using policy π_θ
7: Calculate the confidence score D_ϕ and the surrogate reward function $R(s,a)$ (8).
8: Based on D_ϕ, divide state-action pairs (s,a) into π_{ex} and π_n (3).
9: Update the policy π_θ using the PPO with reward $R(s,a)$.
10: **end for**

IV. EXPERIMENT

In this section, we evaluate the performance of our model on robotic manipulation, navigation, and classic MuJoCo control tasks, comparing it against several state-of-the-art baselines.



(a) AntReach (b) HandRotate (c) FetchPush
(d) Maze2D (e) Hopper (f) Walker

Fig. 2. Six robotic tasks used in our experiments: (a) AntReach, (b) HandRotate, (c) FetchPush, (d) Maze2D, (e) Hopper, and (f) Walker.

A. Experimental Setup

We evaluate our method across six robotic tasks, as illustrated in Fig. 2. The AntReach, HandRotate, FetchPush, and Maze2D datasets are sourced from Lai et al. [23], while Hopper and Walker are sourced from Wang et al. [9].

AntReach: This is a locomotion and navigation task for a quadrupedal robot characterized by a high-dimensional state space (132 dimensions). The objective is for the ant robot to navigate to randomly assigned target positions. The dataset consists of 25,000 state-action pairs.

HandRotate: In this task, a 24-DoF dexterous hand must learn to rotate a cube held within it to a target orientation. This task features both a high-dimensional state space (68 dimensions) and a high-dimensional action space (20 dimensions). The dataset comprises 10,000 state-action pairs.

FetchPush: This task involves training an agent to push a randomly positioned black cube to a designated target location within a 3D environment using a 7-DoF robotic arm. The dataset contains 20,311 state-action pairs.

Maze2D: This is a classical two-dimensional navigation task that simulates the process of an agent planning a path within a constrained space to reach a target location. The task involves controlling a point-mass robot whose actions are represented as continuous displacement vectors, while the environment consists of a maze with obstacles. The dataset contains 18,525 state–action pairs.

Hopper: This task involves controlling a monoped robot to hop forward continuously. The robot must maintain momentum and balance on one leg. The dataset contains 40,000 state-action pairs.

Walker: In this task, a bipedal robot must continuously walk forward while maintaining balance. The dataset contains 40,000 state–action pairs.

Details: Our experimental setup is based on DRAIL. We employ a conditioned diffusion model to enhance the discriminator and adopt PPO as the forward RL algorithm. More specific hyperparameters are detailed in TABLE I. All experiments were conducted on a server equipped with an NVIDIA RTX 4090 GPU.

TABLE I

HYPERPARAMETER SETTINGS

Hparams	Ant	Hand	Push	Maze	Hopper	Walker
Batch Size	128	128	128	128	128	128
Shift Param b	0.4	0.4	0.4	0.4	3	3
D_ϕ Layer	5	3	5	5	5	5
D_ϕ Hidden Dim	1024	128	1024	128	1024	1024
D_ϕ LR	1e-3	1e-4	1e-3	1e-3	1e-4	1e-4
PPO Layer	3	3	3	3	3	3
PPO Hidden Dim	256	64	256	64	64	64
PPO LR	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
Discount λ	0.99	0.99	0.99	0.99	0.99	0.99
Env Steps	1e7	5e6	5e6	1e7	5e6	1e7

TABLE II

NUMBER OF DEMONSTRATIONS (1%, 10%, AND TOTAL) ACROSS ENVIRONMENTS. TOTAL OF DEMONSTRATIONS USED IN DRAIL [23] AND DIFFAIL [9].

	Demonstration Counts		
Environment	**1%**	**10%**	**Total**
AntReach	250	2500	25000
HandRotate	100	1000	10000
FetchPush	203	2031	20311
Maze2D	185	1852	18525
Hopper	400	4000	40000
Walker	400	4000	40000

B. Baselines

We compare our method against the following baseline approaches:

- **BC** [4] learns the policy through supervised learning. The objective is to minimize the difference between its predicted actions and the expert's actions in identical states.
- **GAIL** [6] consists of a discriminator and a generator. It learns a policy resembling the expert's by distinguishing between expert demonstrations and the generated policy.
- **DiffAIL** [9] employs an unconditional diffusion model as the discriminator, leveraging the strong distribution modeling capability of diffusion models to provide higher-quality rewards for policy learning.
- **DRAIL** [23] uses a conditional diffusion model as the discriminator, providing rewards by comparing the diffusion losses under two different class-conditioning labels.
- **SMILING** [26] utilizes score matching to replace the discriminator, proposing a non-adversarial training framework. This approach alleviates issues such as training instability, mode collapse, and optimization difficulties when expert demonstrations are limited.

C. Experimental Results

We evaluated our method on six benchmark tasks: AntReach, HandRotate, FetchPush, Maze2d, Hopper, and Walker, and compared it against five representative baselines: BC, GAIL, DiffAIL, DRAIL, and SMILING. For each dataset, we uniformly sampled 1% and 10% of the state–action pairs from the dataset to serve as expert demonstrations. The corresponding numbers of demonstrations are listed in Table II. Performance was measured using success rate for AntReach, HandRotate, FetchPush, and Maze2D, and average return for Hopper and Walker. All experiments were repeated five times with different random seeds. The results

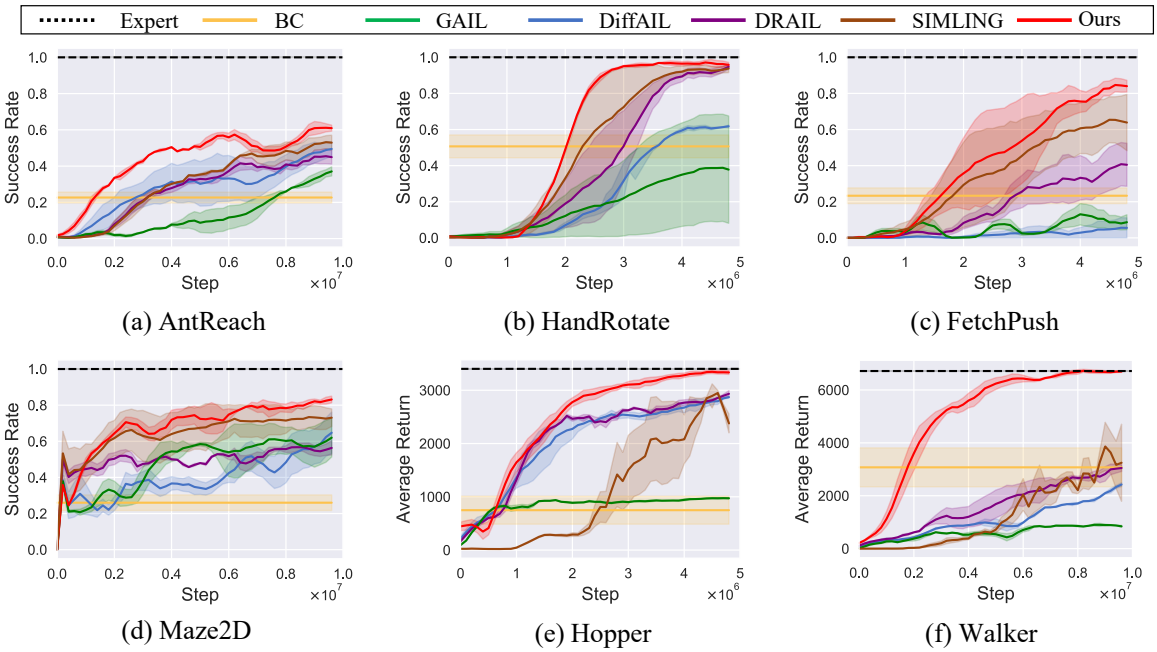


Fig. 3. Performance comparison on six tasks using **1%** of expert demonstrations (AntReach: 250, HandRotate: 100, FetchPush: 203, Maze2D: 185, Hopper: 400, Walker: 400), averaged over five random seeds. AntReach, HandRotate, FetchPush, and Maze2D are evaluated by success rate, while Hopper and Walker are evaluated by return. Our method outperforms the baseline methods and achieves faster convergence with lower variance.

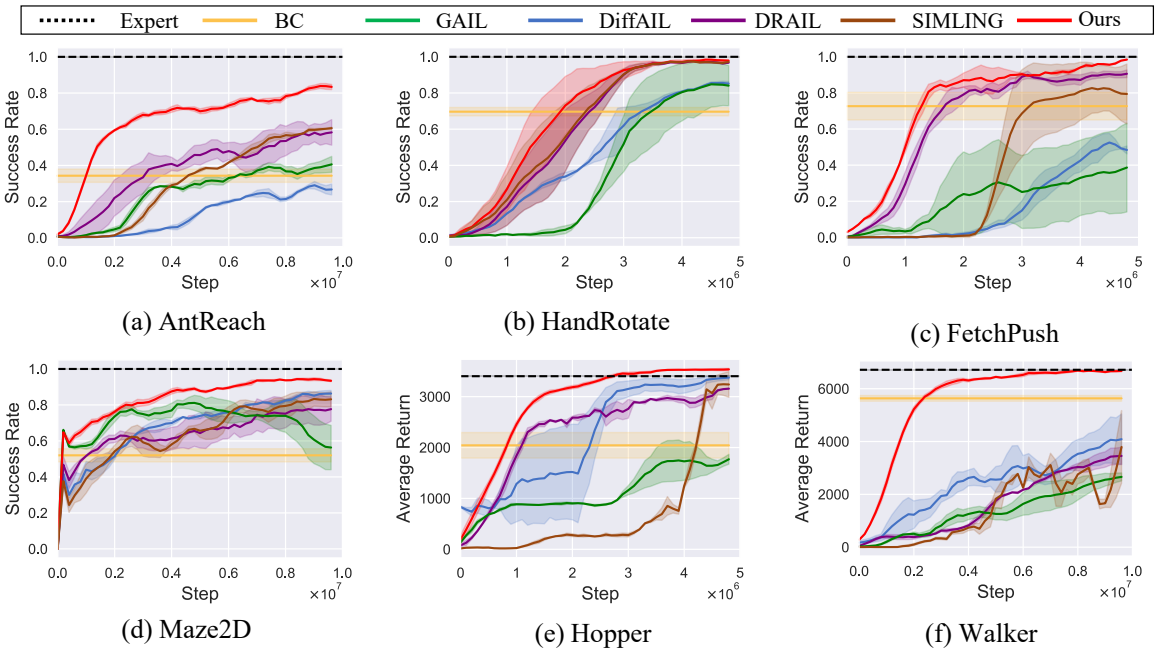


Fig. 4. Performance comparison on six tasks using **10%** of expert demonstrations (AntReach: 2500, HandRotate: 1000, FetchPush: 2031, Maze2D: 1852, Hopper: 4000, Walker: 4000), averaged over five random seeds. AntReach, HandRotate, FetchPush, and Maze2D are evaluated by success rate, while Hopper and Walker are evaluated by return. Our method outperforms the baseline methods and achieves faster convergence with lower variance.

are presented in Fig. 3 and Fig. 4. Since BC does not interact with the environment during training, its performance is shown as a horizontal line in the plots.

The 1% setting evaluates performance under limited data conditions. EDAIL achieved success rates of 61.10%, 97.13%, 84.67%, and 83.12% on AntReach, HandRotate, FetchPush, and Maze2D, respectively, consistently outperforming all baselines. On the MuJoCo locomotion benchmarks, our method achieved an average return of 3346.36 on Hopper and 6684.18 on Walker, once again outperforming competing approaches under limited expert demonstrations. Among these tasks, AntReach is particularly challenging due to its 132-dimensional state space; here, baseline methods such as BC, GAIL, DiffAIL, and DRAIL all achieved success rates below 50% with such limited demonstrations. The FetchPush task also proved difficult, as it requires precise 3D object manipulation, making it especially challenging for GAIL and DiffAIL to acquire effective policies in the low-data regime. On the Walker task, with only 2M timesteps, our method achieves a return comparable to that of DRAIL at 9M timesteps, indicating a significantly faster convergence.

EDAIL's performance improved as more data became available. With 10% of the expert data, our method achieved success rates of 83.88%, 98.40%, 98.47%, and 94.36% on AntReach, HandRotate, FetchPush, and Maze2D, respectively, along with average returns of 3536.49 on Hopper and 6685.38 on Walker. Moreover, it exhibited smaller variance on the AntReach, Maze2D, Hopper, and Walker tasks. Across all six tasks, our method not only outperformed the baselines but also converged faster, demonstrating superior sample

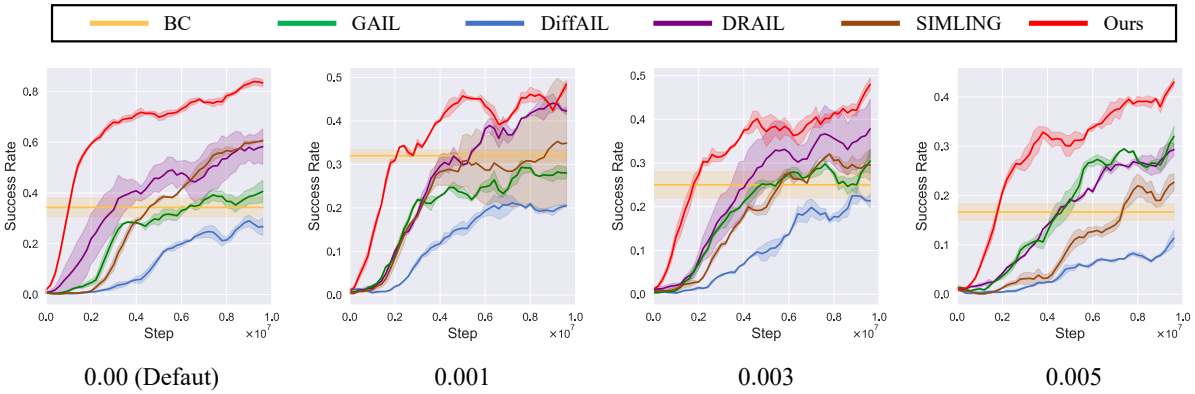


Fig. 5. Performance comparison on the AntReach task with 10% expert demonstrations under three noise levels of 0.01, 0.03, and 0.05. This demonstrates the generalization capability of our method under unseen conditions.

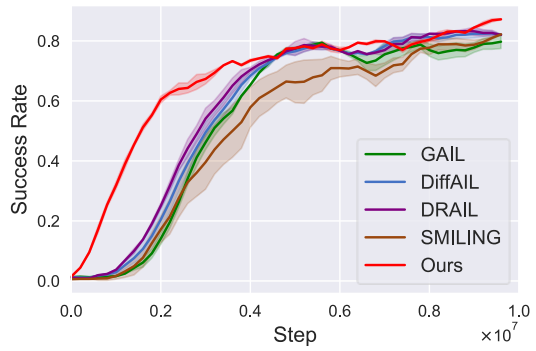


Fig. 6. Success rate comparison on the AntReach task using 100% of expert demonstrations. EDAIL maintains better final performance. Meanwhile, our method achieves the fastest convergence in the early stages of training, possibly due to the use of high-quality agent data through exploratory policies.

efficiency and robustness. Notably, on the Walker task, our method achieved the average return reached by DRAIL at 9M steps using only 1.7M environment steps. On the Hopper task, our method even exceeded the performance of the expert demonstrations in the dataset, which achieved an average return of 3402.00. This indicates that our discriminator, guided by the exploratory policies, effectively captured the underlying rationale of expert behavior from limited data, while the asymmetric surrogate reward provided accurate and stable guidance for policy optimization.

D. Ablation Study

In this section, we conducted a series of ablation experiments to examine the robustness and contribution of each component in our proposed method.

Generalization under Noisy Initial States. To evaluate the generalization capability of EDAIL, we performed experiments on the AntReach task using only 10% of the expert demonstration dataset. We augmented the environment by injecting noise into the robot's initial state, including its initial position, joint angles, and target position. The magnitude of the perturbation was controlled by the specified noise level, and we tested three levels: 0.01, 0.03, and 0.05. This setup creates more challenging evaluation scenarios and allows us to assess whether the learned policy can adapt to variations not encountered during training. The results, shown in Fig. 5, indicate that our method achieved success rates of 48.40%, 48.00%, and 43.04% under the three noise levels, compared to 44.07%, 37.80%, and 29.33% obtained by DRAIL. While all methods experienced performance degradation in these challenging settings, our method consistently maintained a clear advantage over the baselines. This suggests that it learns a generalizable policy rather than merely replicating observed demonstrations.

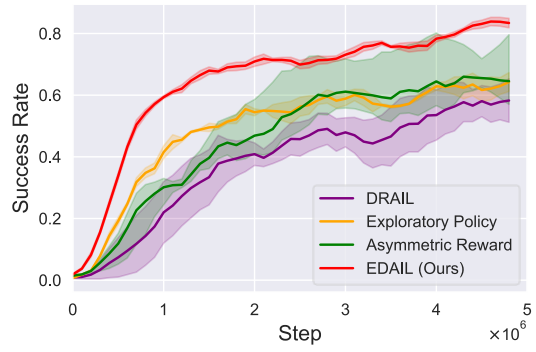


Fig. 7. Effect of two EDAIL components. Exploratory Policy achieves faster convergence with smaller variance, while Asymmetric Reward occasionally attains higher success rates. Each component individually surpasses DRAIL in success rate, and when combined, the overall performance achieves the best result.

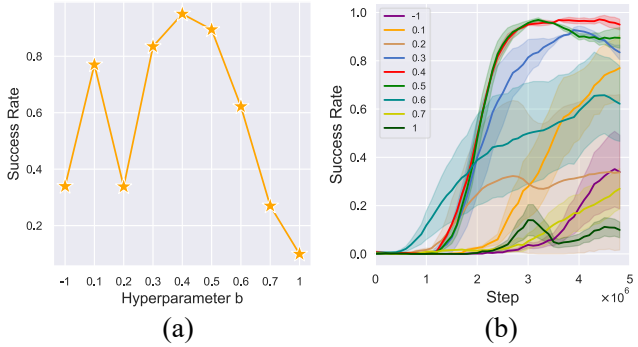


Fig. 8. Performance variation with different values of the shift parameter b in the asymmetric surrogate reward function.

Performance under Full-Data Regime. We further evaluated our method in the full-data regime on the AntReach task, as shown in Fig. 6. Although it is primarily designed for scenarios with limited expert demonstrations, it does not suffer from performance degradation when trained with the complete dataset. Using 100% of the expert demonstrations, the method also achieved better results and faster convergence compared to baseline methods.

Effect of Individual Components. We conducted ablation experiments to quantify the individual contributions of the two core components in EDAIL. In Exploratory Policy, the asymmetric surrogate reward was removed, retaining only the exploratory policies together with the standard surrogate reward function $r = \log D - \log(1 - D)$, so as to isolate the effect of the exploratory policies. In Asymmetric Reward, the asymmetric surrogate reward was adopted while the exploration strategy was removed, enabling assessment of its standalone impact on performance. The results in Fig. 7 show that Exploratory Policy achieves faster convergence and smaller variance; however, due to the inherent bias of the standard surrogate reward function, its performance plateaus at a success rate of 62.47%, leaving room for further improvement. Asymmetric Reward occasionally achieves higher success rates, but it often struggles to optimize effectively and exhibits high variance, with an overall success rate of 63.00%. Both components surpass DRAIL in success rate, and their combination in EDAIL yields the best overall performance of 83.88%.

Impact of Shift Parameter in Asymmetric Surrogate Reward. Based on the dynamics complexity of the task and the diversity of plausible behaviors, environments are typically divided into two categories. One category includes simpler tasks with easier dynamics and relatively concentrated behaviors. The other category includes more complex locomotion tasks with longer-horizon dependencies, where limited demonstrations can make the discriminator underestimate expert-like behaviors. For the simpler tasks, we use HandRotate with 1% expert demonstrations as a reference to illustrate the effect of b. Fig. 8 reports the mean and variance of success rates for different values of b. For Ant, Push, and Maze tasks, using the same setting $b = 0.4$ is sufficient to achieve task success rates higher than the baseline methods. For the more complex tasks, we use Walker as a reference and apply a unified setting $b = 3$ for both Walker and Hopper, which likewise results in higher average returns.

V. Conclusion

In this work, we proposed a novel AIL framework named EDAIL. First, we introduced exploratory policies that augment the expert dataset with high-confidence agent-generated samples to improve discriminator coverage and policy generalization. Second, we designed an asymmetric surrogate reward function that shifts the reward–penalty decision boundary, mitigating the risk of misjudging behavior quality under discriminator bias. Our method was evaluated on multiple robotic manipulation, navigation, and locomotion tasks, achieving superior performance compared with state-of-the-art baselines, particularly in scenarios with very limited expert demonstrations. However, the use of additional data increases storage requirements, which imposes greater demands on hardware resources. In future work, we plan to develop an adaptive strategy for selecting the shift parameter in the asymmetric surrogate reward, and to extend our framework to real-world robotic systems, aiming to further enhance its applicability in practical scenarios where expert data is scarce.

References

[1] Y. Wu, L. Pan, W. Wu, G. Wang, Y. Miao, F. Xu, and H. Wang, "RL-GSBridge: 3D Gaussian Splatting Based Real2Sim2Real Method for Robotic Manipulation Learning," in 2025 IEEE International Conference on Robotics and Automation (ICRA), pp. 192–198, 2025.

[2] X. Yao, J. Liu, X. Zhang, and X. Xu, "Receding-Horizon Reinforcement Learning for Time-Delayed Human–Machine Shared Control of Intelligent Vehicles," IEEE Transactions on Human-Machine Systems, pp. 1–10, Jan. 2025.

[3] C. Yan, C. Wang, X. Xiang, K. H. Low, X. Wang, X. Xu, and L. Shen, "Collision-Avoiding Flocking With Multiple Fixed-Wing UAVs in Obstacle-Cluttered Environments: A Task-Specific Curriculum- Based MADRL Approach," in IEEE Transactions on Neural Networks and Learning Systems, vol. 35, no. 8, pp. 10894-10908, Aug. 2024.

[4] T. V. Samak, C. V. Samak, and S. Kandhasamy, "Robust behavioral cloning for autonomous vehicles using end-to-end imitation learning," SAE International Journal of Connected and Automated Vehicles, vol. 4, no. 3, pp. 279–295, 2021.

[5] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," in Proceedings of the 39th International Conference on Machine Learning (ICML), 2022.

[6] J. Ho and S. Ermon, "Generative adversarial imitation learning," in Advances in Neural Information Processing Systems (NeurIPS), vol. 29, 2016.

[7] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," in Proceedings of the 6th International Conference on Learning Representations (ICLR), 2018.

[8] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson, "Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning," in Proceedings of the 7th International Conference on Learning Representations (ICLR), 2019.

[9] B. Wang, G. Wu, T. Pang, Y. Zhang, and Y. Yin, "DiffAIL: Diffusion adversarial imitation learning," in Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), vol. 38, no. 14, pp. 15447–15455, Mar. 2024.

[10] G. Freund, A. Gleave, and S. Levine, "A Coupled Flow Approach to Imitation Learning," in Proceedings of the 40th International Conference on Machine Learning, Jul. 2023, pp. 10357–10372.

[11] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), 2011, pp. 627–635.

[12] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in Proceedings of Robotics: Science and Systems (RSS), 2023.

[13] Z. Wang, J. J. Hunt, and M. Zhou, "Diffusion policies as an expressive policy class for offline reinforcement learning," in Proceedings of the International Conference on Learning Representations (ICLR), 2023.

[14] M. Reuss, M. Li, X. Jia, and R. Lioutikov, "Goal conditioned imitation learning using score-based diffusion policies," in Proceedings of the Robotics: Science and Systems (RSS), 2023.

[15] Z. Li, T. Xu, Z. Qin, Y. Yu, and Z.-Q. Luo, "Imitation learning from imperfection: Theoretical justifications and algorithms," Advances in Neural Information Processing Systems, vol. 36, 2024.

[16] K. Wu, N. Liu, Z. Zhao, D. Qiu, J. Li, Z. Che, Z. Xu, and J. Tang, "Learning From Imperfect Demonstrations With Self-Supervision for Robotic Manipulation," in 2025 IEEE International Conference on Robotics and Automation (ICRA), pp. 16899–16906, 2025.

[17] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in International Conference on Machine Learning (ICML), 2000.

[18] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, et al., "Maximum entropy inverse reinforcement learning," in Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), vol. 8, pp. 1433–1438, Chicago, IL, USA, 2008.

[19] T. Ni, H. Sikchi, Y. Wang, T. Gupta, L. Lee, and B. Eysenbach, "f-IRL: Inverse Reinforcement Learning via State Marginal Matching," in Conference on Robot Learning (CoRL), 2020.

[20] L. Song, D. Li, and X. Xu, "Adaptive generative adversarial maximum entropy inverse reinforcement learning," *Information Sciences*, vol. 695, p. 121712, 2025.

[21] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in International Conference on Machine Learning, 2004.

[22] Y. R. Kim and H. D. Choi, "CNN-based Apprenticeship Learning for Inverse Reinforcement Learning," in 2024 24th International Conference on Control, Automation and Systems (ICCAS), pp. 73–78, 2024.

[23] C. M. Lai, H. C. Wang, P. C. Hsieh, F. Wang, M. H. Chen, and S. H. Sun, "Diffusion-reward adversarial imitation learning," in Advances in Neural Information Processing Systems (NeurIPS), vol. 37, pp. 95456–95487, 2024.

[24] R. Dadashi, L. Hussenot, M. Geist, and O. Pietquin, "Primal Wasserstein Imitation Learning," in Proceedings of the International Conference on Learning Representations (ICLR), 2021.

[25] G. Swamy, D. Wu, S. Choudhury, D. Bagnell, and S. Wu, "Inverse reinforcement learning without reinforcement learning," in International Conference on Machine Learning, pp. 33299–33318, PMLR, 2023.

[26] R. Wu, Y. Chen, G. Swamy, K. Brantley, and W. Sun, "Diffusing states and matching scores: A new framework for imitation learning," in Proceedings of the 13th International Conference on Learning Representations (ICLR), 2025.

[27] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in Proceedings of the 35th International Conference on Machine Learning (ICML), 2018.

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