Enhancing Inverse Reinforcement Learning through Encoding Dynamic Information in Reward Shaping

Simon Sinong Zhan[*1] SINONGZHAN2028@U.NORTHWESTERN.EDU
Philip Wang[*1] PHILIPWANG2025@U.NORTHWESTERN.EDU
Qingyuan Wu[*2] QINGYUAN.WU@SOTON.AC.UK
Ruochen Jiao[1] RUOCHENJIAO2024@U.NORTHWESTERN.EDU
Yixuan Wang[1] YIXUANWANG2024@U.NORTHWESTERN.EDU
Chao Huang[2] CHAO.HUANG@SOTON.AC.UK
Qi Zhu[1] QZHU@NORTHWESTERN.EDU
[1] *Department of Electrical and Computer Engineering, Northwestern University, USA*
[2] *School of Electronics and Computer Science, University of Southampton, UK*

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

Adversarial-based inverse reinforcement learning (IRL) has shown promising results using reward shaping under deterministic settings, but it struggles in stochastic environments. To address this issue, we propose a novel maximum causal entropy based off-policy IRL method with transition-aware reward shaping framework. Our method integrates transition model estimation directly to learn stochastic-invariant rewards. We conduct a thorough theoretical analysis, establishing bounds on reward error and performance differences to validate the effectiveness of our method. The experimental results in continuous locomotion tasks (MuJoCo) show that our method can achieve superior performance in stochastic environments and competitive performance in deterministic environments, with significant improvement in sample efficiency, compared to existing baselines. Additionally, we extend our framework to high-dimensional vision-based tasks, where our method shows promising results on multiple stochastic Atari games. These results demonstrate that embedding transition awareness into reward learning is critical for robust IRL in realistic stochastic settings.

Keywords: Inverse Reinforcement Learning; Reinforcement Learning; Stochastic MDP

1. Introduction

Reinforcement learning (RL) has achieved considerable success across various domains, including board game (Schrittwieser et al., 2020), MOBA game (Berner et al., 2019), time-delayed system (Wu et al., 2024b,a), and cyber-physical systems (Wang et al., 2023a,b,c; Zhan et al., 2024). Despite these advances, RL highly depends on the quality of reward function design which demands expertise, intensive labour, and a great amount of time (Russell, 1998). To address this, imitation learning (IL) methods, such as Behavior Cloning (BC) (Torabi et al., 2018a) and Inverse Reinforcement Learning (IRL) (Arora and Doshi, 2021), leverage human or expert demonstrations to bypass the need for explicit reward functions. These methods aim to learn from the demonstrations to eventually match the distribution of expert behavior, and have shown great promise in applications like autonomous driving (Codevilla et al., 2018; Sun et al., 2018), legged locomotion (Peng et al., 2020; Ratliff et al., 2009), and planning tasks (Choudhury et al., 2018; Yin et al., 2022).

*. These authors contributed equally to this work

The notable approaches within IRL are Generative Imitation Learning (GIL) methods that build upon maximum entropy framework (Ziebart et al., 2008). These methods frame imitation learning as a maximum likelihood estimation problem on trajectory distributions, converting it into a Boltzmann distribution parameterized by rewards under **deterministic** environment settings (Wu et al., 2024a). This closely mirrors the distribution approximation found in generative models (Finn et al., 2016a; Swamy et al., 2021). Thus, model-free deep IL approaches often follow generative model structures, such as GANs (Ho and Ermon, 2016; Fu et al., 2017) or diffusion models (Reuss et al., 2023; Wu et al., 2024c), and require extensive sampling for distribution/score function matching in on-policy fashion (Orsini et al., 2021). Model-based IL frameworks have also emerged, where model-based framework is designed to provide estimation for gradient and planning, leading to innovative combinations such as gradient-based IRL with model predictive control (MPC) (Das et al., 2021) and end-to-end differentiable IRL frameworks for complex robotics tasks (Baram et al., 2016, 2017; Sun et al., 2021; Rafailov et al., 2021). However, these approaches primarily address deterministic settings and struggle when applied to stochastic environments.

Existing IRL methods, rooted in their maximum entropy nature, mostly exclusively focus on learning "deterministic" reward techniques. These methods, however, face significant performance degeneration in stochastic environments, leading to **risk-seeking behavior** and increased data requirements (Ziebart et al., 2010). For example, an agent trained under the deterministic Markov Decision Process (MDP) might aim to imitate expert behavior by seeking high rewards, yet fail to account for the low probability of some transitions in stochastic MDP settings. This happens because, in stochastic environments, the assumption of maximum entropy no longer holds. The trajectory distributions are not aligned with a Boltzmann distribution solely parameterized by **deterministic** rewards. In this case, the dynamic information must also be incorporated into the formulation. There are two likely solutions: One is massive sampling to cover all possible outcomes, which is computationally expensive in large state action spaces (Devlin and Kudenko, 2011; Gupta et al., 2022); The other is changing from maximum entropy framework to maximum causal entropy framework, estimating the dynamics information, and integrating it into the reward design, making the reward "stochastic". Traditional reward design is usually based on state only $R(s_t)$ (Torabi et al., 2018b), state-action pair $R(s_t, a_t)$ (Blondé and Kalousis, 2019), or transition tuple $R(s_t, a_t, s_{t+1})$ (Fu et al., 2017), where the information inputted can be thought as a **deterministic** sample piece under the **stochastic** setting. The challenge in **stochastic** environments calls for a different perspective of rewards – stochastic rewards absorbing the transition information. More detailed literature survey has been has been investigated in Appendix A.

Inspired by this idea, we propose a novel maximum causal entropy off-policy model-based adversarial IRL framework with a specifically tailored transition-aware reward shaping approach to elevate performance in stochastic environments while remaining competitive in deterministic settings. In contrast to existing methods, our approach leverages the predictive power of the estimated transition model to shape rewards, represented as $\hat{R}(s_t, a_t, \hat{\mathcal{T}})$. This also enables us to guide policy optimization and reduce dependency on costly real-world interactions in a model-based fashion. As part of our analysis, we provide a theoretical guarantee on the optimal behavior for policies induced by our reward shaping and derive a bound on the performance difference with respect to the transition model learning errors. Empirically, we demonstrate that this integration significantly enhances sample efficiency and policy performance in both settings, providing a comprehensive solution to the limitations of existing IRL methods in uncertain environments. **Contributions of this work** include:

- A transition-aware reward shaping formulation $\hat{R}(s_t, a_t, \hat{\mathcal{T}})$ with model estimation $\hat{\mathcal{T}}$ for stochastic MDPs, which provides the optimal policy invariance guarantee.
- A novel model-based off-policy IRL framework rooted in **maximum causal entropy** theory that seamlessly incorporates transition model training, adversarial reward learning with model estimation and forward model-based RL process, enhancing performance in stochastic environments, and sample efficiency.
- Theoretical analysis on reward learning and performance difference under transition model learning errors within our framework.
- Empirical validation showing our approach's performance improvements in stochastic environments as well as significant gains in sample efficiency and comparable performance in deterministic environments.

We begin by introducing the necessary preliminaries on Markov Decision Processes (MDPs) and inverse reinforcement learning (IRL). We then present our model-enhanced reward shaping method, along with its theoretical guarantee. Next, we describe the full model-enhanced IRL framework, including its derivation from the maximum causal entropy objective, and provide theoretical analysis on the reward error bound and performance difference bound. Experimental results are reported on a range of MuJoCo and Atari benchmarks, demonstrating the effectiveness of our approach. We conclude with a summary of our findings.

2. Preliminaries

MDP. RL usually considers a Markov Decision Process (MDP) $\mathcal{M}$ (Puterman, 2014) denoted as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, R, \rho_0 \rangle$. ρ_0 is the initial distribution of the state. $s \in \mathcal{S}, a \in \mathcal{A}$ stands for the state and action space respectively. $\mathcal{T}$ stands for the transition dynamic such that $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$. $\gamma \in (0, 1)$ is the discounted factor, R stands for reward function such that $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $\|R\|_\infty \leq R_{\max}$. The undiscounted visitation distribution of trajectory τ with policy π is given by $p(\tau) = \rho_0 \prod_{t=0}^{T-1} \mathcal{T}(s_{t+1}|s_t, a_t)\pi(a_t|s_t)$. The objective function of RL is defined as $\max \mathbb{E}_{\tau \sim p(\tau)} \left[\sum_{t=0}^{T} \gamma^t R(\tau) - H(\pi) \right]$, where H is the log likelihood of the policy. We introduce Soft Value Iteration for bellmen update (Haarnoja et al., 2018), where Q^{soft} and V^{soft} denotes the soft Q function and Value function respectively:

$$V^{soft}(s_t) = \log \sum_{a_t \in \mathcal{A}} \exp Q^{soft}(s_t, a_t), \quad Q^{soft}(s_t, a_t) = R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}} \left[V^{soft}(s_{t+1}) | s_t, a_t \right],$$

$$\pi(a_t|s_t) = \exp\left(Q^{soft}(s_t, a_t) - V^{soft}(s_t)\right), \quad A^{soft}(s_t, a_t) = Q^{soft}(s_t, a_t) - V^{soft}(s_t)$$

Inverse RL. In the IRL setting, we usually consider the MDP without reward as $\mathcal{M}'$ where R is also unknown. We denote the data buffer $\mathcal{D}_{exp}$ which collects trajectories from an expert policy π^E. We consider a reward function $R_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, where θ is the reward parameter. An IRL problem can be defined as a pair $\mathcal{B} = (\mathcal{M}', \pi^E)$. A reward function R_θ is feasible for $\mathcal{B}$ if π^E is an optimal policy for the MDP $\mathcal{M}' \cup R_\theta$, and we denote the set of feasible rewards as $\mathcal{R}_\mathcal{B}$. Using the maximize likelihood estimation framework, we can formulate the IRL as the following maximum causal entropy (MCE) problem, $\arg\max_\theta \mathbb{E}_{\tau \sim \mathcal{D}_{exp}} \log p_\theta(\tau)$, where $Q^{soft}_{R_\theta}$ and $V^{soft}_{R_\theta}$ are based on R_θ and $p_\theta(\tau) \propto \rho_0 \prod_{t=0}^{T-1} \mathcal{T}(s_{t+1}|s_t, a_t) \exp(Q^{soft}_{R_\theta}(s_t, a_t) - V^{soft}_{R_\theta}(s_t))$ (Ziebart et al., 2010). Under **deterministic** MDP, the above problem can be simplified as maximum entropy (ME) problem, where $p_\theta(\tau) \propto \frac{1}{Z_\theta} \exp \sum_{t=0}^{T-1} R_\theta(s_t, a_t)$ and Z_θ is the temperature factor of the Boltzmann Distribution (Ziebart et al., 2008).

Table 1: This table summarizes different reward formulations and their dynamic properties. **Components** refer to the input pairs for the reward functions. **Reward Shaping** indicates whether additional physical potential information is included (**X** means none). **Dynamics** specifies if transitions are considered in the reward function.

Methods	Components	Reward Shaping	Dynamic Information
AIRL(State Only) (Fu et al., 2017)	s_t	$R(s_t)$ + constant	X
DAC (Kostrikov et al., 2018)	s_t, a_t	X	X
SAM (Blondé and Kalousis, 2019)	s_t, a_t	X	X
SQIL (Reddy et al., 2019)	s_t, a_t	binary	X
GAIfO (Torabi et al., 2018b)	s_t, s_{t+1}	X	single sample
Ours	$s_t, a_t, \mathcal{T}$	$R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}[\phi(s_{t+1})\|s_t, a_t] - \phi(s_t)$	transition model

3. Transition-Aware Reward Shaping

In this section, we illustrate the advantages of involving transition dynamics into the reward shaping, especially in stochastic MDP settings. Existing work has proposed various formulations and definitions (Table 1), but few considers transition dynamic information in the reward shaping. Defining rewards solely based on states, $R^s(s_t)$, offers limited utility in environments where actions are critical. Even though the state-action pair-based rewards $R^{sa}(s_t, a_t)$ can capture the missing information of the action taken, it fails to consider future information, the successive state s_{t+1}. Transition tuple-based rewards $R^{tuple}(s_t, a_t, s_{t+1})$ incorporate dynamic information in a sampling-based way, which requires abundant data to learn the underlying relationship of two consecutive states, potentially raising the sample efficiency issue in the stochastic environment with the huge state space. To address this issue, we propose transition-aware reward shaping $\hat{R}(s_t, a_t, \mathcal{T})$, which explicitly infuses dynamic information $\mathcal{T}$ on the potential function, thus significantly improving sample efficiency. Specifically, our reward shaping is defined as

$$\hat{R}(s_t, a_t, \mathcal{T}) = R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}\left[\phi(s_{t+1})|s_t, a_t\right] - \phi(s_t), \tag{1}$$

where ϕ is a state-only potential function, $\mathcal{T}$ is the dynamics. Another insight of the above reward shaping is to resemble the advantage function with the soft value function as the optimal potential function, which we will elaborate on next section. With the given reward shaping $\hat{R}$, it is crucial to show that it induces the same optimal behavior as the original reward R. We formally define this policy invariance property as follows.

Definition 1 *(Memarian et al., 2021) Let R and $\hat{R}$ be two reward functions. We say they induce the same soft optimal policy under transition dynamics $\mathcal{T}$ if, for all states $s \in \mathcal{S}$ and actions $a \in \mathcal{A}$: $A_R^{soft}(s_t, a_t) = A_{\hat{R}}^{soft}(s_t, a_t)$.*

With the above definition, we can transfer the proof of policy invariant property of our designed reward shaping (Equation 1) to showing the equivalence of soft advantage functions, which is proved in the following theorem. The detailed proof can be found in Appendix A.

Theorem 2 (Policy Invariance) *Let R and $\hat{R}$ be two reward functions. R and $\hat{R}$ induce the same soft optimal policy under all transition dynamics $\mathcal{T}$ if $\hat{R}(s_t, a_t, \mathcal{T}) = R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}[\phi(s_{t+1})|s_t, a_t] - \phi(s_t)$ for some potential-shaping function $\phi : \mathcal{S} \to \mathbb{R}$.*

Theorem 2 implies that the optimal policy induced from our transition-aware reward shaping $\hat{R}$ (Equation 1) is equivalent to the optimal policy trained by the ground-truth reward function R under the soft value iteration fashion.

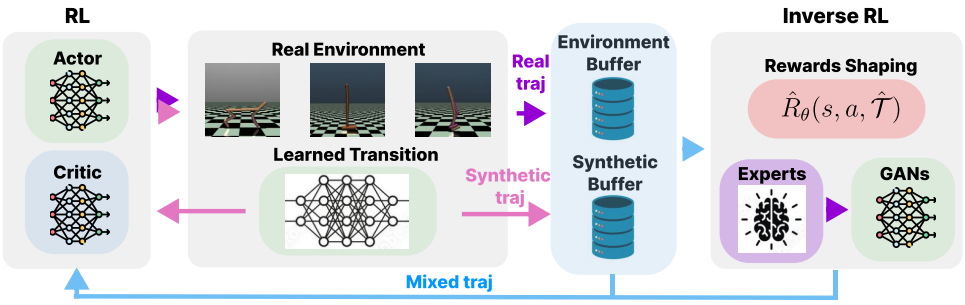

Figure 1: Framework overview of Model-Enhanced Adversarial IRL. Different color arrows stand for different sample flows. Purple stands for real environmental interaction samples, pink stands for synthetic samples generated from learned transition model, and blue stands for mixed of both.

4. Model Enhanced IRL

In this section, we first elaborate on the adversarial training of our reward shaping (Equation 2) and present the theoretical insight (Equation 3) of the equivalence between cross-entropy training loss of adversarial reward shaping formulation and maximum log-likelihood loss of original maximum causal entropy IRL problem. Then, we showcase our practical algorithm framework with trajectory generation, transition model learning, and policy optimization in the loop, as shown in Figure 1. Furthermore, we theoretically investigate the reward function bound (Theorem 4) and performance difference bound (Theorem 5) under the transition model learning error.

4.1. Adversarial Formulation of Reward Shaping

In this section, we connect the reward shaping in the adversarial training framework with rewards learning objective under the MCE framework. Inspired by GANs (Goodfellow et al., 2014), the idea behind the adversarial framework is to train a binary discriminator $D(s_t, a_t, s_{t+1})$ or $D(s_t, a_t)$ to distinguish state-action-transition samples from an expert and those generated by imitator policy following the original ME setting. However, as mentioned above, we only take in state-action pair and transition function to define our reward function which also extends to our discriminator as:

$$D_\theta(s_t, a_t, \mathcal{T}) = \frac{\exp\{f_\theta(s_t, a_t, \mathcal{T})\}}{\exp\{f_\theta(s_t, a_t, \mathcal{T})\} + \pi(a_t|s_t)}, \tag{2}$$

where $f_\theta(s_t, a_t, \mathcal{T}) = R_\theta(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}\left[\phi_\theta(s_{t+1})|s_t, a_t\right] - \phi_\theta(s_t)$ resembles the reward shaping defined above. The loss function for the training discriminator is defined below.

$$\mathcal{L}_{disc} = -\mathbb{E}_{\mathcal{D}_{exp}}\left[\log D_\theta(s, a, \mathcal{T})\right] - \mathbb{E}_\pi\left[\log(1 - D_\theta(s, a, \mathcal{T}))\right]. \tag{3}$$

We bridge this adversarial formulation with the original MCE IRL problem. In the following proposition, we give a sketch of the proof to show the connection between the objective function of the discriminator and MCE IRL. Proof details can be found in Appendix B.

Proposition 3 *Consider an undiscounted MDP. Suppose f_θ and π at the current iteration are the soft-optimal advantage function and policy for reward function R_θ. Minimising the cross-entropy loss of the discriminator under generator π is equivalent to maximising the log-likelihood under Maximum Causal Entropy IRL.*

With the above proposition, we can construct an intuition that f_θ^* should be equal to $\hat{R}_\theta$ the reward shaping we introduced early and resemble the soft advantage function. To extract rewards to represent reward used for policy optimization, we use $\log(D_\theta(s,a,\mathcal{T})) - \log(1 - D_\theta(s,a,\mathcal{T}))$, which resembles the entropy-regularized reward shaping $f_\theta(s,a,\mathcal{T}) - \log \pi(a|s)$.

4.2. Algorithm Framework

In this section, we present the overall framework of Model-Enhanced IRL and illustrate how transition model training is incorporated into the learning loop. We use ensemble gaussian dynamic models to adopt the transition (Yu et al., 2020, 2021; Rigter et al., 2022). The transition model is updated in each policy optimization iteration similar as model-based RL approaches (Janner et al., 2019; Hansen et al., 2022; Zhan et al., 2024). At each iteration, the updated transition model is utilized for reward learning and synthetic data generation in eval mode, which is stored in the synthetic trajectory replay buffer. Unlike AIRL and GAIL, our framework operates in an **off-policy** fashion, where samples used for both discriminator and policy update are drawn from a combination of the environmental replay buffer and the synthetic replay buffer. An overview of our framework is shown in Figure 1, and detailed algorithmic steps and parameters are provided in Algorithm 1.

Sample Efficiency: To improve sample efficiency, we leverage the estimated transition model to generate H-steps synthetic trajectories data alongside real interaction data, facilitating policy optimization. Given that the estimated transition model is inaccurate at the beginning, we employ a dynamic ratio between real and synthetic data to prevent the model from being misled by unlikely synthetic transitions (Janner et al., 2019; Zhan et al., 2024). Specifically, early-stage generated trajectories are not stored persistently, unlike real interactions which are fully stored in the off-policy environmental replay buffer. To maintain training stability, we use a synthetic replay buffer with a size that gradually increases as training progresses, ensuring a balanced inclusion of synthetic data over time. The growth rates of the data ratio and buffer size are adjusted based on the complexity of the transition model learning process and can be fine-tuned via hyper-parameters. Details can be found in Appendix F.

Distribution Shift: To mitigate distribution shift (Lee et al., 2020; Lin et al., 2020) during training, we employ a strategy involving the learned transition model. Typically, during interaction, the real state s_t is used as input to the actor, and the resulting action a_t is applied in the environment. To incorporate the transition model, we predict a synthetic state $\hat{s}_t$ from previous s_{t-1} and a_{t-1}. This generated $\hat{s}_t$ is then fed into the actor to produce action $\hat{a}_t$. The actions a_t and $\hat{a}_t$ are mixed and applied to the environment with a certain ratio, and the resulting pairs (s_t, a_t) or $(s_t, \hat{a}_t)$ are stored in the environmental replay buffer. This approach helps balance the exploration of real and model-predicted dynamics, reducing the impact of distributional discrepancies.

4.3. Performance Analysis

In this section, we analyze the optimal performance bound in the presence of transition model learning errors. Our results show that as the transition model error approaches zero, the performance difference at the optimal point vanishes at the same time. The learned transition model $\hat{\mathcal{T}}$ persists in some errors compared with the ground-true transition dynamic. In this section, we investigate how this error will affect performance of our method. extending original representation, we define an IRL problem as $\mathfrak{B} = (\mathcal{M}', \pi^E)$, where $\mathcal{M}'$ is a MDP without R and π^E is an optimal expert policy. We

denote $\mathcal{R}_{\mathfrak{B}}$ as the set of feasible rewards set for $\mathfrak{B}$. Since under our case $\mathcal{T}$ is approximated by $\hat{\mathcal{T}}$, we have another IRL problem defined as $\hat{\mathfrak{B}} = (\hat{\mathcal{M}}', \pi^E)$ where $\hat{\mathcal{M}}'$ has the same state and action space, discount factor, and initial distribution but an estimated transition model $\hat{\mathcal{T}}$. For notation, we use D_{TV} to denote the total variation distance, $\|\cdot\|$ to represent the infinity norm (with ∞ omitted for simplicity), $|\mathcal{S}|$ to denote the cardinality of the state space, and $V^{\pi^*}_{\mathcal{M}' \cup R}$ to represent the value function of policy π^* under the MDP $\mathcal{M}'$ with reward R, and vice versa.

Assumption 1 (Transition Model Error) *Since the transition model is trained through a supervised fashion, we can use a PAC generalization bound (Shalev-Shwartz and Ben-David, 2014) for sample error. Therefore, we assume that the total variation distance between $\mathcal{T}$ and $\hat{\mathcal{T}}$ is bounded by $\epsilon_{\mathcal{T}}$ through $[0, T]$: $\max_t \mathbb{E}_{s \sim \pi_{D,t}} \left[D_{TV}(\mathcal{T}(s'|s,a) | \hat{\mathcal{T}}(s'|s,a)) \right] \leq \epsilon_{\mathcal{T}}$, which is a common assumption adopted in literature (Janner et al., 2019; Sikchi et al., 2022).*

Next, based on the assumed total visitation bound on transition models (Assumption 1), we aim to propagate this bound to the reward learning process through our model-infused reward shaping. Specifically, the error bound arises from using the approximated transition model $\hat{\mathcal{T}}$ in our reward shaping, instead of the true but inaccessible transition model $\mathcal{T}$.

Theorem 4 (Reward Function Error Bound) *Let $\mathfrak{B} = (\mathcal{M}', \pi^*)$ and $\hat{\mathfrak{B}} = (\hat{\mathcal{M}}', \pi^*)$ be two IRL problems with transition functions $\mathcal{T}$ and $\hat{\mathcal{T}}$ respectively, then for any $R^E \in \mathcal{R}_{\mathfrak{B}}$ there is a corresponding $\hat{R}^E \in \mathcal{R}_{\hat{\mathfrak{B}}}$ such that*

$$\|R^E - \hat{R}^E\| \leq \frac{\gamma}{1-\gamma} |\mathcal{S}| \epsilon_{\mathcal{T}} R_{\max}$$

Proof of Theorem 4 can be found in Equation 10. With rewards bound above, we can extend the bound to the final performance, which represents the value functions difference brought up by estimated transition model error under RL setting.

Theorem 5 (Performance Difference Bound) *The performance difference between the optimal policies (π^* and $\hat{\pi}^*$) in corresponding MDPs ($\mathcal{M}' \cup R$ and $\hat{\mathcal{M}}' \cup \hat{R}$) can be bounded as follows:*

$$\|V^{\pi^*}_{\mathcal{M}' \cup R^E} - V^{\hat{\pi}^*}_{\hat{\mathcal{M}}' \cup \hat{R}^E}\| \leq \epsilon_{\mathcal{T}} \left[\frac{\gamma}{(1-\gamma)^2} R_{\max} + \frac{1+\gamma}{(1-\gamma)^2} R_{\max} |\mathcal{S}| \right].$$

The detailed proof of Theorem 5 is presented in Equation 13. The above theorem highlights the relationship between the performance difference and the transition model error, also implying that a perfectly-learned transition model ($\epsilon_{\mathcal{T}} \to 0$) could make the performance difference negligible. We also extend an ablation study to empirically validate this performance bound in ablation studies in subsection D.2, where we can observe a *linear-kind* tendency illustrating in above theorem.

5. Experiments

We evaluate our Model-Enhanced IRL algorithm on the MuJoCo and Atari`100k` benchmarks (Todorov et al., 2012; Bellemare et al., 2013; Kaiser et al., 2019) with an NVIDIA RTX 4090 GPU and Intel Core-i9 13900K CPU. Our algorithm is tested against other Adversarial Imitation Learning (AIL) methods, including on-policy algorithms GAIL (Ho and Ermon, 2016), AIRL (Fu et al., 2017), and the off-policy method Discriminator Actor-Critic (DAC) (Kostrikov et al., 2018) across multiple

Table 2: Best performance of expert and all algorithms in **stochastic** MuJoCo Environments under conditions of different numbers of expert trajectories provided (10, 100, and 1000). AIRL and GAIL are trained with `10M` environmental steps. DAC and Ours are trained with `1M` environmental steps.

Environment	# Expert Trajactories	Expert	GAIL	AIRL	DAC	Ours
InvertedPendulum-v4	10	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$
	100	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$
	1000	$986.09_{\pm 95.97}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$
InvertedDoublePendulum-v4	10	$131.3_{\pm 77.0}$	$155.0_{\pm 58.1}$	$163.0_{\pm 48.6}$	$100.6_{\pm 11.8}$	$193.4_{\pm 15.5}$
	100	$108.0_{\pm 43.2}$	$167.2_{\pm 26.6}$	$151.2_{\pm 28.6}$	$94.5_{\pm 9.9}$	$198.1_{\pm 76.3}$
	1000	$140.44_{\pm 76.62}$	$189.5_{\pm 28.8}$	$150.2_{\pm 18.3}$	$105.6_{\pm 20.4}$	$182.2_{\pm 29.6}$
Hopper-v4	10	$1786.0_{\pm 803.0}$	$1266.9_{\pm 366.2}$	$2092.2_{\pm 57.4}$	$1000.4_{\pm 5.3}$	$2408.4_{\pm 641.7}$
	100	$1489.6_{\pm 659.6}$	$2385.9_{\pm 350.0}$	$2789.9_{\pm 30.8}$	$993.1_{\pm 10.5}$	$2820.9_{\pm 89.8}$
	1000	$1516.0_{\pm 692.6}$	$2746.5_{\pm 270.9}$	$2744.3_{\pm 37.4}$	$2007.1_{\pm 719.7}$	$2858.8_{\pm 76.9}$
HalfCheetah-v4	10	$1567.4_{\pm 74.1}$	$368.5_{\pm 53.7}$	$463.9_{\pm 61.2}$	$9.5_{\pm 457.2}$	$888.6_{\pm 67.3}$
	100	$1120.5_{\pm 67.5}$	$398.1_{\pm 123.5}$	$556.0_{\pm 12.8}$	$615.9_{\pm 250.5}$	$1108.3_{\pm 13.9}$
	1000	$1113.5_{\pm 76.1}$	$735.6_{\pm 44.0}$	$708.7_{\pm 14.5}$	$1046.4_{\pm 13.9}$	$1162.8_{\pm 62.2}$
Walker2d-v4	10	$3109.4_{\pm 1031.5}$	$1262.8_{\pm 396.3}$	$1170.5_{\pm 484.0}$	$101.4_{\pm 149.1}$	$2509.0_{\pm 860.0}$
	100	$3295.4_{\pm 704.0}$	$956.4_{\pm 313.2}$	$1740.7_{\pm 609.8}$	$416.1_{\pm 243.2}$	$3311.0_{\pm 157.2}$
	1000	$3268.9_{\pm 746.1}$	$1430.6_{\pm 489.8}$	$3051.3_{\pm 210.5}$	$3531.3_{\pm 105.3}$	$3497.8_{\pm 51.7}$

MuJoCo environments under both stochastic and deterministic settings. We use Proximal Policy Optimization (PPO) (Schulman et al., 2017) for both GAIL and AIRL, and Soft Actor-Critic (SAC) (Haarnoja et al., 2018) for DAC as the forward RL optimization respectively. In addition, we apply our transition-aware reward shaping framework to the high-dimensional vision-based Atari`100k` task and compare it with CNN-AIRL (Tucker et al., 2018). Complete learning curves of all methods are provided in Supplementary Material. The experiments are designed to highlight the key advantages of our framework:

- **Performance in Stochastic Environments:** In stochastic settings, our method significantly outperforms other approaches. This enhanced ability to learn under uncertainty is attributed to our framework's effectiveness in leveraging model-based predictiton capability.
- **Sample Efficiency** Our method can consistently reach expert performance with fewer training steps even under limited expert demonstrations when most of the baselines fail to extract reward signals.
- **Performance in Deterministic Environments:** We demonstrate that our method is on par with existing baselines' performances in deterministic settings.

MuJuCo Experiment Setup. We choose 5 different MuJoCo continuous control tasks listed in Table 2. To simulate stochastic dynamics, we introduce the agent-unknown random Gaussian noise with a mean of 0 and a standard deviation of 0.5 to the environmental interaction steps. All the expert trajectories are collected by an expert agent trained with standard SAC (Haarnoja et al., 2018) under deterministic or stochastic MuJoCo environments. Each algorithm is trained with 100k environmental steps and evaluated every 1k steps for `InvertedPendulum-v4` and `InvertedDoublePendulum-v4`. For `Hopper-v4`, `HalfCheetah-v4`, and `Walker2d-v4`, AIRL and GAIL are trained with 10M steps with evaluations every 100k steps, whereas DAC and our algorithm are trained with 1M environmental steps with evaluations every 10k steps. Each experiment is repeated with five random seeds, and we vary the number of expert demonstrations from 5 to 1000 to demonstrate robustness.

Stochastic MuJoCo. In Table 2, we present the performance of our method and baselines in stochastic MuJoCo environments with varying numbers of expert trajectories. **Our method consistently achieves the best performance across the majority of these environments, outperforming all baselines under different levels of expert trajectory availability**. In simple environments, such as `InvertedPendulum-v4`, the introduction of stochasticity and variations in expert trajectory have minimal impact on the final performance for both our method and the baselines. However, for more complex environments, the effect of stochasticity becomes more pronounced. Specifically, in `InvertedDoublePendulum-v4`, stochasticity notably degrades performance. Our method, however, maintains a competitive edge over all baselines, achieving better results with limited expert trajectories (10 and 100) and reaching similar performance to the baselines when more expert trajectories are available. In other environments with higher dimensional state space, our method substantially outperforms all baselines, especially when fewer expert trajectories are provided. These results indicate that our approach can effectively recover the reward function more closely from demonstrations in stochastic environments, resulting in significant performance improvement. To demonstrate our model's robustness to different levels of randomness and stochasticity, we further conduct an ablation study that can be found in Supplementary Material. Notably, the performance of DAC decreases significantly under stochastic settings, potentially due to DAC's ineffective reward formulation on state-action pairs, which also result in training instability shown in learning curves section in supplement.

Sample Efficiency. In Figure G, we display the sample efficiency across various environments with different numbers of expert trajectories. Based on results, **our method shows significant superiority in sample efficiency across all of the environments under stochastic settings. This advantage becomes more apparent with limited expert trajectories.** Specifically, for `Hopper-v4`, `HalfCheetah-v4`, and `Walker2d-v4` our method is the only approach capable of reaching expert-level performance consistently with limited number of expert trajectories. GAIL and AIRL both fail to reach the expert within `1M` environmental training steps. DAC was able to reach expert performance only when expert trajectories are sufficient, though it still suffers from sample inefficiency and training instability. For `InvertedPendulum-v4`, all methods can achieve expert-level performance except DAC, which exhibits instability with limited demonstrations. In `InvertedDoublePendulum-v4`, introducing stochasticity into the dynamics makes it challenging for all algorithms to achieve reasonable performance from noisy expert demonstrations while DAC completely fails to reach expert-level performance. We also observe a universal trend across all stochastic environments: as the number of expert trajectories increases, both the sample efficiency and performance of all methods improve accordingly. In order to determine the specific contributions of our model-based policy optimization and model-infused reward shaping, we investigate their individual impacts on sample efficiency as detailed in ablation study in Supplementary Material. The results indicate that both components contribute significantly to the overall performance, with the model-based policy optimization playing a crucial role in enhancing sample efficiency, especially in environments with complex dynamics and high-dimensional state spaces.

Deterministic MuJoCo. The performance of deterministic MuJoCo environments can be found in Table 3. For most tasks with deterministic dynamics, our method can achieve similar performance as the baselines and the expert. For `HalfCheetah-v4`, our method exceeds AIRL and GAIL, but fail to reach the similar level as DAC and expert. As the dynamic becomes complicated, ensemble dynamic model with our fixed hyper-parameters might not be able to fully capture the transition

Table 3: Best performance of expert and all algorithms in **deterministic** MuJoCo Environments with 1000 expert trajectories provided. DAC and our methods are trained for `1M` environmental steps. GAIL and AIRL are trained for `10M` environmental steps.

Environment	Expert	GAIL	AIRL	DAC	Ours
InvertedPendulum-v4	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$
InvertedDoublePendulum-v4	$9356.7_{\pm 0.2}$	$9324.4_{\pm 0.4}$	$355.3_{\pm 76.3}$	$9359.8_{\pm 0.1}$	$9359.8_{\pm 0.1}$
Walker2d-v4	$4520.7_{\pm 648.44}$	$3387.0_{\pm 617.8}$	$3623.6_{\pm 189.6}$	$4655.3_{\pm 126.4}$	$4396.7_{\pm 147.4}$
Hopper-v4	$3262.8_{\pm 314.4}$	$3420.8_{\pm 77.9}$	$3385.8_{\pm 50.5}$	$3481.6_{\pm 94.6}$	$3506.6_{\pm 23.5}$
HalfCheetah-v4	$13498.6_{\pm 710.9}$	$3502.6_{\pm 202.3}$	$3237.8_{\pm 85.5}$	$10102.2_{\pm 297.6}$	$6509.8_{\pm 177.7}$

info leading to high compounding transition model error, which can cause the performance deficit. Our theoretical analysis supports this finding, and we will explore the efficacy of different dynamic model structures or more effective hyper-parameters setting for future works. Generally, **our method shows competitive performance with the baselines in the deterministic environments.**

Stochastic Atari Task. In addition to the continuous MuJoCo locomotion suite, we further evaluate our reward shaping framework on two high-dimensional visual observation based Atari 2600 task, built on the Arcade Learning Enivonrment (ALE) simulator (Bellemare et al., 2013). The environment is naturally stochastic with the built-in sticky actions. We employ the standard Atari`100k` evaluation protocol where the algorithms are trained for 400k steps, 100k interactions considering action repeat (Kaiser et al., 2019). We obtain the expert trajectories by training Dreamer-v2 (Hafner et al., 2020) for `400k` steps and sample 10 trajectories afterwards. In order to capture the high-dimensional observation information, we substitute original ensemble model with the Recurrent State-Space Model (RSSM) to encode the dynamics. Table 4 shows our method comparing to CNN-AIRL (Tucker et al., 2018), an adapted IRL framework that preprocesses the image input through CNN then use it for recovering rewards. From the result, our method can recover expert behavior to some extent under the high-dimensional stochastic environment. More details of this experiment can be found in Supplementary Materials.

Table 4: Evaluation return after `400k` steps on visual Atari.

Method	SpaceInvaders	BattleZone
Expert	$445.5_{\pm 129.0}$	$22300.0_{\pm 3662.0}$
CNN-AIRL	$148.0_{\pm 73.2}$	$1200.0_{\pm 748.3}$
Ours	$323.0_{\pm 181.5}$	$14400.0_{\pm 5607.1}$

6. Conclusion

In this paper, we presented a novel off-policy model-enhanced IRL framework starting from Maximum Causal Entropy theory by introducing transition-aware reward shaping, specifically designed to enhance performance in stochastic environments with significant sample efficiency improvement comparing to existing approaches and maintain competitive performance in deterministic setting. The theoretical analysis provides guarantees on the optimal policy invariance under the transition-aware reward shaping and highlights the relationship between performance difference and transition model's estimation error. Empirical evaluations on MuJoCo and Atari benchmark environments validate the effectiveness of our method. Future works will focus on extending to various generative structure for reward learning and exploring extensions of the framework to multi-agent and hierarchical reinforcement learning scenarios. Overall, our approach offers a promising direction for advancing model-based adversarial IRL, with the potential to scale to a broader range of real-world applications.

Acknowledgments

Simon Sinong Zhan and Qi Zhu acknowledge the support from HAMMER ERC EEC-2133630 and National Science Foundation grants 2324936, 2328973, 2328032. Qingyuan Wu and Chao Huang are supported by the grant EP/Y002644/1 under the EPSRC ECR International Collaboration Grants program, funded by the International Science Partnerships Fund (ISPF) and the UK Research and Innovation.

References

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297:103500, 2021.

Kai Arulkumaran and Dan Ogawa Lillrank. A pragmatic look at deep imitation learning. In *Asian Conference on Machine Learning*, pages 58–73. PMLR, 2024.

Nir Baram, Oron Anschel, and Shie Mannor. Model-based adversarial imitation learning. *arXiv preprint arXiv:1612.02179*, 2016.

Nir Baram, Oron Anschel, Itai Caspi, and Shie Mannor. End-to-end differentiable adversarial imitation learning. In *International Conference on Machine Learning*, pages 390–399. PMLR, 2017.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Lionel Blondé and Alexandros Kalousis. Sample-efficient imitation learning via generative adversarial nets. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3138–3148. PMLR, 2019.

Lionel Blondé, Pablo Strasser, and Alexandros Kalousis. Lipschitzness is all you need to tame off-policy generative adversarial imitation learning. *Machine Learning*, 111(4):1431–1521, 2022.

Alex J Chan and Mihaela van der Schaar. Scalable bayesian inverse reinforcement learning. *arXiv preprint arXiv:2102.06483*, 2021.

Jaedeug Choi and Kee-Eung Kim. Map inference for bayesian inverse reinforcement learning. *Advances in neural information processing systems*, 24, 2011.

Sanjiban Choudhury, Mohak Bhardwaj, Sankalp Arora, Ashish Kapoor, Gireeja Ranade, Sebastian Scherer, and Debadeepta Dey. Data-driven planning via imitation learning. *The International Journal of Robotics Research*, 37(13-14):1632–1672, 2018.

Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4693–4700. IEEE, 2018.

Neha Das, Sarah Bechtle, Todor Davchev, Dinesh Jayaraman, Akshara Rai, and Franziska Meier. Model-based inverse reinforcement learning from visual demonstrations. In *Conference on Robot Learning*, pages 1930–1942. PMLR, 2021.

Rati Devidze, Parameswaran Kamalaruban, and Adish Singla. Exploration-guided reward shaping for reinforcement learning under sparse rewards. *Advances in Neural Information Processing Systems*, 35:5829–5842, 2022.

Sam Devlin and Daniel Kudenko. Theoretical considerations of potential-based reward shaping for multi-agent systems. In *Tenth International Conference on Autonomous Agents and Multi-Agent Systems*, pages 225–232. ACM, 2011.

Sam Michael Devlin and Daniel Kudenko. Dynamic potential-based reward shaping. In *11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, pages 433–440. IFAAMAS, 2012.

Marco Dorigo and Marco Colombetti. Robot shaping: Developing autonomous agents through learning. *Artificial intelligence*, 71(2):321–370, 1994.

EclecticSheep, Davide Angioni, Federico Belotti, Refik Can Malli, and Michele Milesi. SheepRL, May 2023. URL https://github.com/Eclectic-Sheep/sheeprl/.

Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016a.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58. PMLR, 2016b.

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

Adam Gleave, Mohammad Taufeeque, Juan Rocamonde, Erik Jenner, Steven H. Wang, Sam Toyer, Maximilian Ernestus, Nora Belrose, Scott Emmons, and Stuart Russell. imitation: Clean imitation learning implementations. arXiv:2211.11972v1 [cs.LG], 2022. URL https://arxiv.org/abs/2211.11972.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Abhishek Gupta, Aldo Pacchiano, Yuexiang Zhai, Sham Kakade, and Sergey Levine. Unpacking reward shaping: Understanding the benefits of reward engineering on sample complexity. *Advances in Neural Information Processing Systems*, 35:15281–15295, 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 *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

Anna Harutyunyan, Sam Devlin, Peter Vrancx, and Ann Nowé. Expressing arbitrary reward functions as potential-based advice. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

Michael Herman, Tobias Gindele, Jörg Wagner, Felix Schmitt, and Wolfram Burgard. Inverse reinforcement learning with simultaneous estimation of rewards and dynamics. In *Artificial intelligence and statistics*, pages 102–110. PMLR, 2016.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Yujing Hu, Weixun Wang, Hangtian Jia, Yixiang Wang, Yingfeng Chen, Jianye Hao, Feng Wu, and Changjie Fan. Learning to utilize shaping rewards: A new approach of reward shaping. *Advances in Neural Information Processing Systems*, 33:15931–15941, 2020.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and JoẠ̃Go GM AraÃšjo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

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. *arXiv preprint arXiv:1809.02925*, 2018.

Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Addressing distribution shift in online reinforcement learning with offline datasets. 2020.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. *Advances in neural information processing systems*, 24, 2011.

Zichuan Lin, Garrett Thomas, Guangwen Yang, and Tengyu Ma. Model-based adversarial meta-reinforcement learning. *Advances in Neural Information Processing Systems*, 33:10161–10173, 2020.

Farzan Memarian, Wonjoon Goo, Rudolf Lioutikov, Scott Niekum, and Ufuk Topcu. Self-supervised online reward shaping in sparse-reward environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2369–2375. IEEE, 2021.

Alberto Maria Metelli, Giorgia Ramponi, Alessandro Concetti, and Marcello Restelli. Provably efficient learning of transferable rewards. In *International Conference on Machine Learning*, pages 7665–7676. PMLR, 2021.

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

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? *Advances in Neural Information Processing Systems*, 34: 14656–14668, 2021.

Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.

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

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Visual adversarial imitation learning using variational models. *Advances in Neural Information Processing Systems*, 34: 3016–3028, 2021.

Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *IJCAI*, volume 7, pages 2586–2591, 2007.

Jette Randløv and Preben Alstrøm. Learning to drive a bicycle using reinforcement learning and shaping. In *ICML*, volume 98, pages 463–471, 1998.

Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum margin planning. In *Proceedings of the 23rd international conference on Machine learning*, pages 729–736, 2006.

Nathan D Ratliff, David Silver, and J Andrew Bagnell. Learning to search: Functional gradient techniques for imitation learning. *Autonomous Robots*, 27:25–53, 2009.

Siddharth Reddy, Anca D Dragan, and Sergey Levine. Sqil: Imitation learning via reinforcement learning with sparse rewards. *arXiv preprint arXiv:1905.11108*, 2019.

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

Marc Rigter, Bruno Lacerda, and Nick Hawes. Rambo-rl: Robust adversarial model-based offline reinforcement learning. *Advances in neural information processing systems*, 35:16082–16097, 2022.

Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 101–103, 1998.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

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

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Harshit Sikchi, Wenxuan Zhou, and David Held. Learning off-policy with online planning. In *Conference on Robot Learning*, pages 1622–1633. PMLR, 2022.

Joar Max Viktor Skalse, Matthew Farrugia-Roberts, Stuart Russell, Alessandro Abate, and Adam Gleave. Invariance in policy optimisation and partial identifiability in reward learning. In *International Conference on Machine Learning*, pages 32033–32058. PMLR, 2023.

Jiankai Sun, Lantao Yu, Pinqian Dong, Bo Lu, and Bolei Zhou. Adversarial inverse reinforcement learning with self-attention dynamics model. *IEEE Robotics and Automation Letters*, 6(2):1880–1886, 2021.

Liting Sun, Cheng Peng, Wei Zhan, and Masayoshi Tomizuka. A fast integrated planning and control framework for autonomous driving via imitation learning. In *Dynamic Systems and Control Conference*, volume 51913, page V003T37A012. American Society of Mechanical Engineers, 2018.

Gokul Swamy, Sanjiban Choudhury, J Andrew Bagnell, and Steven Wu. Of moments and matching: A game-theoretic framework for closing the imitation gap. In *International Conference on Machine Learning*, pages 10022–10032. PMLR, 2021.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018a.

Faraz Torabi, Garrett Warnell, and Peter Stone. Generative adversarial imitation from observation. *arXiv preprint arXiv:1807.06158*, 2018b.

Aaron Tucker, Adam Gleave, and Stuart Russell. Inverse reinforcement learning for video games. *arXiv preprint arXiv:1810.10593*, 2018.

Luca Viano, Yu-Ting Huang, Parameswaran Kamalaruban, Adrian Weller, and Volkan Cevher. Robust inverse reinforcement learning under transition dynamics mismatch. *Advances in Neural Information Processing Systems*, 34:25917–25931, 2021.

Yixuan Wang, Ruochen Jiao, Chengtian Lang, Sinong Simon Zhan, Chao Huang, Zhaoran Wang, Zhuoran Yang, and Qi Zhu. Empowering autonomous driving with large language models: A safety perspective. *arXiv preprint arXiv:2312.00812*, 2023a.

Yixuan Wang, Simon Zhan, Zhilu Wang, Chao Huang, Zhaoran Wang, Zhuoran Yang, and Qi Zhu. Joint differentiable optimization and verification for certified reinforcement learning. In *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, pages 132–141, 2023b.

Yixuan Wang, Simon Sinong Zhan, Ruochen Jiao, Zhilu Wang, Wanxin Jin, Zhuoran Yang, Zhaoran Wang, Chao Huang, and Qi Zhu. Enforcing hard constraints with soft barriers: Safe reinforcement learning in unknown stochastic environments. In *International Conference on Machine Learning*, pages 36593–36604. PMLR, 2023c.

Eric Wiewiora, Garrison W Cottrell, and Charles Elkan. Principled methods for advising reinforcement learning agents. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 792–799, 2003.

Qingyuan Wu, Simon Sinong Zhan, Yixuan Wang, Yuhui Wang, Chung-Wei Lin, Chen Lv, Qi Zhu, and Chao Huang. Variational delayed policy optimization. *arXiv preprint arXiv:2405.14226*, 2024a.

Qingyuan Wu, Simon Sinong Zhan, Yixuan Wang, Yuhui Wang, Chung-Wei Lin, Chen Lv, Qi Zhu, Jürgen Schmidhuber, and Chao Huang. Boosting reinforcement learning with strongly delayed feedback through auxiliary short delays. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 53973–53998. PMLR, 21–27 Jul 2024b. URL https://proceedings.mlr.press/v235/wu24af.html.

Runzhe Wu, Yiding Chen, Gokul Swamy, Kianté Brantley, and Wen Sun. Diffusing states and matching scores: A new framework for imitation learning. *arXiv preprint arXiv:2410.13855*, 2024c.

Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

Zhao-Heng Yin, Weirui Ye, Qifeng Chen, and Yang Gao. Planning for sample efficient imitation learning. *Advances in Neural Information Processing Systems*, 35:2577–2589, 2022.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

Sinong Zhan, Yixuan Wang, Qingyuan Wu, Ruochen Jiao, Chao Huang, and Qi Zhu. State-wise safe reinforcement learning with pixel observations. In *6th Annual Learning for Dynamics & Control Conference*, pages 1187–1201. PMLR, 2024.

Amy Zhang, Shagun Sodhani, Khimya Khetarpal, and Joelle Pineau. Learning robust state abstractions for hidden-parameter block mdps. *arXiv preprint arXiv:2007.07206*, 2020.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. Modeling interaction via the principle of maximum causal entropy. 2010.

Appendix A. Related Works

Generative IRL. Margin optimization based IRL methods (Ng et al., 2000; Abbeel and Ng, 2004; Ratliff et al., 2006) aim to learn reward functions that explain expert behavior better than other policies by a margin. Bayesian approaches were introduced with different prior assumptions on reward distributions, such as Boltzmann distributions (Ramachandran and Amir, 2007; Choi and Kim, 2011; Chan and van der Schaar, 2021) or Gaussian Processes (Levine et al., 2011). To avoid biases from maximum margin methods, (Ziebart et al., 2008, 2010) proposed a Lagrangian dual framework to cast the reward learning into a maximum likelihood problem with linear-weighted feature-based reward representation. Wulfmeier et al. (2015) extended the framework to nonlinear reward representations, and Finn et al. (2016b) combined importance sampling techniques to enable model-free estimation. Inspired by GANs, generative methods were introduced for policy and reward learning in IRL (Ho and Ermon, 2016; Fu et al., 2017; Torabi et al., 2018b). However, **these methods typically work with Maximum Entropy (ME) formulation yet suffer from sample inefficiency and stochasticity**. Although there have been efforts to combine generative methods with off-policy RL agents to improve sample efficiency (Kostrikov et al., 2018; Blondé and Kalousis, 2019; Blondé et al., 2022), few extend it to the model-based setting which might further the improvement, and none of these approaches addresses the rewards learning in stochastic MDP.

MBIRL. Integrating IRL with MBRL has also shown success. For example, Das et al. (2021) and Herman et al. (2016) presented a gradient-based IRL approach using different policy optimization methods with dynamic models for linear-weighted features reward learning. In Das et al. (2021), the dynamic model is used to pass forward/backward the gradient in order to update the IRL and policy optimization modules. Similarly, end-to-end differentiable adversarial IRL frameworks to various state spaces have also been explored (Baram et al., 2016, 2017; Sun et al., 2021; Rafailov et al., 2021), where dynamic model serves a similar role. Despite these advancements, existing methods rarely address the specific challenges posed by stochastic environments, which limit reward learning performance.

Reward Shaping. Reward shaping (Dorigo and Colombetti, 1994; Randløv and Alstrøm, 1998) is a technique that enhances the original reward signal by adding additional domain information, making it easier for the agent to learn optimal behavior. This can be defines as $\hat{R} = R + F$, where F is the shaping function and $\hat{R}$ is the shaped reward function. Potential-based reward shaping (PBRS) (Ng et al., 2000) builds the potential function on states, $F(s, a, s') = \phi(s') - \phi(s)$, while ensuring the policy invariance property, which refers to inducing the same optimal behavior under different rewards R and $\hat{R}$. Nonetheless, there exists other variants on the inputs of the potential functions such as state-action (Wiewiora et al., 2003), state-time (Devlin and Kudenko, 2012), and value function (Harutyunyan et al., 2015) as potential function input. There are also some recent attempts of reward shaping without utilization of domain knowledge potential function to solve exploration under sparse rewards (Hu et al., 2020; Devidze et al., 2022; Gupta et al., 2022; Skalse et al., 2023).

Appendix A. Reward Shaping Soft Optimal Policy

Theorem 6 *Let R and $\hat{R}$ be two reward functions. R and $\hat{R}$ induce the same soft optimal policy under all transition dynamics $\mathcal{T}$ if $\hat{R}(s_t, a_t) = R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}[\phi(s_{t+1})|s_t, a_t] - \phi(s_t)$ for some potential-shaping function $\phi : \mathcal{S} \to \mathbb{R}$.*

Proof According to Soft VI (section 2), we can expand the representation of $Q^{soft}_{\hat{R}}(s_t, a_t)$ as follows.

$$
\begin{aligned}
Q^{soft}_{\hat{R}}(s_t, a_t) &= R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}\left[\phi(s_{t+1})|s_t, a_t\right] - \phi(s_t) + \gamma \mathbb{E}_{\mathcal{T}}\left[V^{soft}_{\hat{R}}(s_{t+1})|s_t, a_t\right], \\
Q^{soft}_{\hat{R}}(s_t, a_t) + \phi(s_t) &= R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}\left[V^{soft}_{\hat{R}}(s_{t+1}) + \phi(s_{t+1})|s_t, a_t\right], \\
Q^{soft}_{\hat{R}}(s_t, a_t) + \phi(s_t) &= R(s_t, a_t) + \gamma \mathbb{E}_{\mathcal{T}}\left[\log \sum_{a \in \mathcal{A}} \exp\left(Q^{soft}_{\hat{R}}(s_{t+1}, a) + \phi(s_{t+1})\right)|s_t, a_t\right].
\end{aligned}
$$

From above derivation, we can tell that $Q^{soft}_{\hat{R}}(s_t, a_t) + \phi(s_t)$ satisfy the soft bellmen update with original R. Thus, with simple induction, we can arrive that $Q^{soft}_{R}(s_t, a_t) = Q^{soft}_{\hat{R}}(s_t, a_t) + \phi(s_t)$. Then, we can derive the advantage function

$$
\begin{aligned}
A^{soft}_{\hat{R}}(s_t, a_t) &= Q^{soft}_{\hat{R}}(s, a) - V^{soft}_{\hat{R}}(s_t) \\
&= Q^{soft}_{\hat{R}}(s_t, a_t) - \log \sum_{a \in \mathcal{A}} \exp\left(Q^{soft}_{\hat{R}}(s_t, a_t)\right) \\
&= Q^{soft}_{\hat{R}}(s_t, a_t) + \phi(s_t) - \log \sum_{a \in \mathcal{A}} \exp\left(Q^{soft}_{\hat{R}}(s_t, a_t) + \phi(s_t)\right) \\
&= Q^{soft}_{R}(s_t, a_t) - \log \sum_{a \in \mathcal{A}} \exp\left(Q^{soft}_{R}(s_t, a_t)\right) \\
&= A^{soft}_{R}(s_t, a_t).
\end{aligned}
$$

■

Appendix B. Adversarial Reward Learning

Proposition 7 *Consider an undiscounted MDP. Suppose f_θ and π at current iteration are the soft-optimal advantage function and policy for reward function R_θ. Minimising the cross-entropy loss of the discriminator under generator π is equivalent to maximising the log-likelihood under Maximum Causal Entropy IRL.*

Proof

$$
\begin{aligned}
\mathcal{L}_{IRL}(\mathcal{D}_{exp}, \theta) &= \mathbb{E}_{\mathcal{D}_{exp}}[\log p_\theta(\tau)] \\
&= \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-1} \log \pi(a_t|s_t) + \log \rho_0 + \sum_{t=1}^{T} \log \mathcal{T}(s_{t+1}|s_t, a_t)\right] \\
&= \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-1} \left(Q^{soft}_{\theta}(s_t, a_t) - V^{soft}_{\theta}(s_t)\right)\right] + \text{constant}.
\end{aligned}
$$

Breaking down above equations with soft VI (Section 2), we can arrive the following.

$$\mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-1} R_\theta(s_t,a_t)\right] + \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-2} \mathbb{E}_{\mathcal{T}}\left[V_\theta^{soft}(s_{t+1})|s_t,a_t\right]\right] - \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-1} V_\theta^{soft}(s_t)\right]. \quad (4)$$

Next we will derive the gradient of the loss.

$$\nabla_\theta \mathcal{L}(\mathcal{D}_{exp},\theta) = \underbrace{\nabla_\theta \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-1} R_\theta(s_t,a_t)\right]}_{A} + \underbrace{\nabla_\theta \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-2}\left(\mathbb{E}_{\mathcal{T}}\left[V_\theta^{soft}(s_{t+1})|s_t,a_t\right]\right) - V_\theta^{soft}(s_{t+1})\right]}_{B} - \underbrace{\nabla_\theta V_\theta^{soft}(s_0)}_{C}. \quad (5)$$

Let's get explicit expression of each part.

$$\begin{aligned} A &= \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-1} \nabla_\theta R_\theta(s_t,a_t)\right] \\ C &= \nabla_\theta \log \sum_{a_t\in\mathcal{A}} \exp Q_\theta^{soft}(s_t,a_t) \\ &= \sum_{a_t\in\mathcal{A}} \pi(a_t|s_t)\nabla_\theta Q_\theta^{soft}(s_t,a_t) \\ &= \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} \nabla_\theta R_\theta(s_t,a_t)\right]. \end{aligned}$$

In our case, the transition function $\mathcal{T}$ is estimated by an approximation function $\hat{\mathcal{T}}$, which is updated with samples from $\mathcal{D}_{exp}$ and samples from off-policy buffer $\mathcal{D}_{env}$, thus we can drop the $\mathbb{E}_{\mathcal{T}}$ here. And B term will cancel out, ending up to 0. To summarize, the gradient of log MLE loss of MCE IRL is the following.

$$\nabla_\theta \mathcal{L}_{IRL}(\mathcal{D}_{exp},\theta) = \mathbb{E}_{\mathcal{D}_{exp}}\left[\sum_{t=0}^{T-1} \nabla_\theta R_\theta(s_t,a_t)\right] - \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} \nabla_\theta R_\theta(s_t,a_t)\right]. \quad (6)$$

Next, we will start to derive the gradient of cross-entropy discriminator training loss. Remember the discriminator loss is defined in Eq 3.

$$\begin{aligned} \log D_\theta(s,a,\mathcal{T}) &= f_\theta(s,a,\mathcal{T}) - \log(\exp\{f_\theta(s,a,\mathcal{T})\} + \pi(a|s)), \\ \log(1-D_\theta(s,a,\mathcal{T})) &= \log\pi(a|s) - \log(\exp\{f_\theta(s,a,\mathcal{T})\} + \pi(a|s)). \end{aligned}$$

Then, the gradient of each term is as follow:

$$\begin{aligned} \nabla_\theta \log D_\theta(s,a,\mathcal{T}) &= \nabla_\theta f_\theta(s,a,\mathcal{T}) - \frac{\exp\{f_\theta(s,a,\mathcal{T})\}\nabla_\theta f_\theta(s,a,\mathcal{T})}{\exp\{f_\theta(s,a,\mathcal{T})\} + \pi(a|s)}, \\ \nabla_\theta \log(1-D_\theta(s,a,\mathcal{T})) &= -\frac{\exp\{f_\theta(s,a,\mathcal{T})\}\nabla_\theta f_\theta(s,a,\mathcal{T})}{\exp\{f_\theta(s,a,\mathcal{T})\} + \pi(a|s)}. \end{aligned}$$

Since π is trained by using f_θ as shaped reward, from soft VI we can derive that $\pi^*_{f_\theta}(a|s) = \exp A^{soft}_{f_\theta}(s,a)$. By assumption, we assume that f_θ is the advantage function of R_θ, $f_\theta(s,a) = A^{soft}_{R_\theta}(s,a)$. From Theorem 2, we know that $A^{soft}_{R_\theta}(s,a) = A^{soft}_{f_\theta}(s,a)$, which also implies that $\pi^*_{f_\theta} = \pi^*_{R_\theta}$. Then, we can deduce the gradient of the loss of discriminator.

$$\begin{aligned}
-\nabla_\theta \mathcal{L}_{disc} &= \mathbb{E}_{\mathcal{D}_{exp}}\left[\nabla_\theta \log D_\theta(s,a,\mathcal{T})\right] + \mathbb{E}_\pi\left[\nabla_\theta \log(1 - D_\theta(s,a,\mathcal{T}))\right] \\
&= \mathbb{E}_{\mathcal{D}_{exp}}\left[\frac{1}{2}\nabla_\theta f_\theta(s,a,\mathcal{T})\right] - \mathbb{E}_\pi\left[\frac{1}{2}\nabla_\theta f_\theta(s,a,\mathcal{T})\right], \\
-2\nabla_\theta \mathcal{L}_{disc} &= \mathbb{E}_{\mathcal{D}_{exp}}\left[\nabla_\theta f_\theta(s,a,\mathcal{T})\right] - \mathbb{E}_\pi\left[\nabla_\theta f_\theta(s,a,\mathcal{T})\right].
\end{aligned}$$

■

Appendix C. Performance Gap Analysis

Lemma 8 (Implicit Feasible Reward Set (Ng et al., 2000)) *Let $\mathfrak{B} = (\mathcal{M}', \pi^*)$ be an IRL problem. Then $R \in \mathcal{R}_\mathfrak{B}$ if and only if for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ the following holds:*

$$\begin{aligned}
Q^{\pi^*}_{\mathcal{M}' \cup R}(s,a) - V^{\pi^*}_{\mathcal{M}' \cup R}(s) = 0 \quad &\text{if } \pi^*(a|s) > 0, \\
Q^{\pi^*}_{\mathcal{M}' \cup R}(s,a) - V^{\pi^*}_{\mathcal{M}' \cup R}(s) \leq 0 \quad &\text{if } \pi^*(a|s) = 0.
\end{aligned}$$

Combined with the traditional Value Iteration of RL, we can write out the explicit form of the reward function R.

Lemma 9 (Explicit Feasible Reward Function (Metelli et al., 2021)) *With the above lemma conditions, $R \in \mathcal{R}_\mathfrak{B}$ if and only if there exist $\xi \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}_{\geq 0}$ and value function $V \in \mathbb{R}^\mathcal{S}$ such that:*

$$R(s,a) = V(s) - \gamma \sum_{s' \in \mathcal{S}} \mathcal{T}(s'|s,a)V(s') - \xi(s,a)\mathbb{I}\{\pi^*(a|s) = 0\}. \tag{7}$$

With Equation 7, we can derive the following error bound between $R \in \mathcal{R}^E_\mathfrak{B}$ and $\hat{R}^E \in \mathcal{R}_{\hat{\mathfrak{B}}}$.

Theorem 10 (Reward Function Error Bound) *Let $\mathfrak{B} = (\mathcal{M}', \pi^*)$ and $\hat{\mathfrak{B}} = (\hat{\mathcal{M}}', \pi^*)$ be two IRL problems, then for any $R^E \in \mathcal{R}_\mathfrak{B}$ there is a corresponding $\hat{R}^E \in \mathcal{R}_{\hat{\mathfrak{B}}}$ such that*

$$\|R^E - \hat{R}^E\| \leq \frac{\gamma}{1-\gamma}|\mathcal{S}|\epsilon_\mathcal{T} R_{\max}. \tag{8}$$

Proof From Section 9, we can derive the following representations of R and $\hat{R}$ with the same set of V and ξ:

$$\begin{aligned}
R^E(s,a) &= V(s) - \gamma \sum_{s' \in \mathcal{S}} \mathcal{T}(s'|s,a)V(s') - \xi(s,a)\mathbb{I}\{\pi^*(a|s) = 0\}, \\
\hat{R}^E(s,a) &= V(s) - \gamma \sum_{s' \in \mathcal{S}} \hat{\mathcal{T}}(s'|s,a)V(s') - \xi(s,a)\mathbb{I}\{\pi^*(a|s) = 0\}.
\end{aligned}$$

The difference between R^E and $\hat{R}^E$ can be bounded as follows:

$$\|R^E - \hat{R}^E\| \leq \gamma \sum_{s' \in \mathcal{S}} D_{TV}\left(\mathcal{T}(s'|s,a) | \hat{\mathcal{T}}(s'|s,a)\right) \cdot \|V(s')\|.$$

Given that the total variation distance between the two dynamics is bounded by $\epsilon_{\mathcal{T}}$, and the reward function is bounded by $R_{\max}$, together with the definition of the value function, we have $\|V\|_\infty \leq \frac{R_{\max}}{1-\gamma}$. Substituting these bounds, we derive the following inequality:

$$\|R^E - \hat{R}^E\| \leq \frac{\gamma}{1-\gamma} |\mathcal{S}| \epsilon_{\mathcal{T}} R_{\max}.$$

■

Next, we will propagate this bound to the value functions of optimal policy regarding different reward functions R^E and $\hat{R}^E$. From the traditional Value iteration, we can write out the value function.

$$V^\pi_{\mathcal{M}' \cup R^E}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} \mathcal{T}(s'|s,a) \left[R^E(s,a) + \gamma V^\pi_{\mathcal{M}' \cup R^E}(s')\right]. \tag{9}$$

Lemma 11 (Value Function Error under same policy and different rewards and MDP) $||V^\pi_{\mathcal{M}' \cup R}(s) - V^\pi_{\hat{\mathcal{M}}' \cup \hat{R}}(s)||$*: the performance difference of the same policy in different MDPs.*

$$||V^\pi_{\mathcal{M}' \cup R^E}(s) - V^\pi_{\hat{\mathcal{M}}' \cup \hat{R}^E}(s)|| \leq \epsilon_{\mathcal{T}} \frac{1+\gamma}{(1-\gamma)^2} R_{\max} |\mathcal{S}|. \tag{10}$$

Proof

$$\begin{aligned}
&||V^\pi_{\mathcal{M}' \cup R^E}(s) - V^\pi_{\hat{\mathcal{M}}' \cup \hat{R}^E}(s)|| \\
&\leq \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} ||\mathcal{T}(s'|s,a)\left[R^E(s,a) + \gamma V^\pi_{\mathcal{M}' \cup R^E}(s')\right] - \hat{\mathcal{T}}(s'|s,a)\left[R^E(s,a) + \gamma V^\pi_{\mathcal{M}' \cup R^E}(s')\right] \\
&+ \hat{\mathcal{T}}(s'|s,a)\left[R^E(s,a) + \gamma V^\pi_{\mathcal{M}' \cup R^E}(s')\right] - \hat{\mathcal{T}}(s'|s,a)\left[\hat{R}^E(s,a) + \gamma V^\pi_{\hat{\mathcal{M}}' \cup \hat{R}^E}(s')\right]|| \\
&\leq \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} (\epsilon_{\mathcal{T}} \frac{R_{\max}}{1-\gamma} + \hat{\mathcal{T}}(s'|s,a) \epsilon_{\mathcal{T}} (\frac{\gamma R_{\max}}{1-\gamma} |\mathcal{S}| + \gamma ||V^\pi_{\mathcal{M}' \cup R^E}(s') - V^\pi_{\hat{\mathcal{M}}' \cup \hat{R}^E}(s')||)) \\
&= \sum_{a \in \mathcal{A}} \pi(a|s) (\epsilon_{\mathcal{T}} \frac{R_{\max}}{1-\gamma} |\mathcal{S}| + \epsilon_{\mathcal{T}} \frac{\gamma R_{\max}}{1-\gamma} |\mathcal{S}| + \gamma ||V^\pi_{\mathcal{M}' \cup R^E}(s') - V^\pi_{\hat{\mathcal{M}}' \cup \hat{R}^E}(s')||) \\
&\leq \epsilon_{\mathcal{T}} \frac{1+\gamma}{1-\gamma} R_{\max} |\mathcal{S}| + \gamma ||V^\pi_{\mathcal{M}' \cup R^E}(s') - V^\pi_{\hat{\mathcal{M}}' \cup \hat{R}^E}(s')|| \\
&\leq \epsilon_{\mathcal{T}} \frac{1+\gamma}{(1-\gamma)^2} R_{\max} |\mathcal{S}|.
\end{aligned} \tag{11}$$

■

Lemma 12 *Let* $\|V^{\pi_1}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s) - V^{\pi_2}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s)\|$ *denote the performance difference between different policies* π_1 *and* π_2 *in the same learned MDP (Viano et al., 2021; Zhang et al., 2020). The following inequality holds:*

$$\|V^{\pi_1}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s) - V^{\pi_2}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s)\| \leq \frac{\gamma}{(1-\gamma)^2}\epsilon_{\mathcal{T}} R_{\max}.$$

Theorem 13 (Performance Difference Bound) *The performance difference between the optimal policies (*π^* *and* $\hat{\pi}^*$*) in corresponding MDPs (*$\mathcal{M}' \cup R^E$ *and* $\hat{\mathcal{M}}' \cup \hat{R}^E$*) can be bounded as follows:*

$$\|V^{\pi^*}_{\mathcal{M}'\cup R^E} - V^{\hat{\pi}^*}_{\hat{\mathcal{M}}'\cup\hat{R}^E}\| \leq \epsilon_{\mathcal{T}}\left[\frac{\gamma}{(1-\gamma)^2}R_{\max} + \frac{1+\gamma}{(1-\gamma)^2}R_{\max}|\mathcal{S}|\right]. \tag{12}$$

Proof

$$\begin{aligned}
&||V^{\pi^*}_{\mathcal{M}'\cup R^E}(s) - V^{\hat{\pi}^*}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s)|| \\
&\leq ||V^{\pi^*}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s) - V^{\hat{\pi}^*}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s)|| + ||V^{\hat{\pi}^*}_{\mathcal{M}'\cup R^E}(s) - V^{\hat{\pi}^*}_{\hat{\mathcal{M}}'\cup\hat{R}^E}(s)|| \\
&= \epsilon_{\mathcal{T}}\frac{\gamma}{(1-\gamma)^2}R_{\max} + \epsilon_{\mathcal{T}}\frac{1+\gamma}{(1-\gamma)^2}R_{\max}|\mathcal{S}| \\
&= \epsilon_{\mathcal{T}}\left[\frac{\gamma}{(1-\gamma)^2}R_{\max} + \frac{1+\gamma}{(1-\gamma)^2}R_{\max}|\mathcal{S}|\right].
\end{aligned}$$

■

Appendix D. Ablation Studies

D.1. Robustness to stochasticity

In this study, we examine the robustness of our method across varying levels of stochasticity in the environment. Following the same setup as in our main experiments, we introduce an unknown Gaussian noise with different standard deviations in `InvertedPendulum-v4` to simulate increased stochasticity. As shown in Table 5 and Figure 2, our method consistently recovers expert-level performance despite the presence of stochastic disturbances. However, as the level of stochasticity increases, we observe that training stability decreases, as reflected in the increased variance in Figure 2.

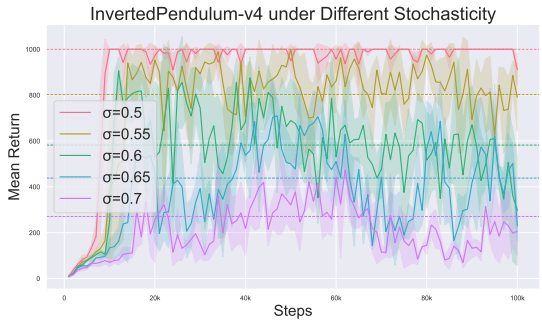

Figure 2: Training return diagram averaging across three seeds for different numbers of expert trajectories in `InvertedPendulum-v4`.

Std	Expert	Ours
0.5	$1000.0_{\pm 0.0}$	$1000.0_{\pm 0.0}$
0.55	$802.4_{\pm 305.8}$	$1000.0_{\pm 0.0}$
0.6	$582.1_{\pm 360.5}$	$906.3_{\pm 59.1}$
0.65	$438.2_{\pm 322.3}$	$709.9_{\pm 80.6}$
0.7	$270.7_{\pm 236.0}$	$472.7_{\pm 85.7}$

Table 5: Best performance of expert and our method in `InvertedPendulum-v4` environments with different Gaussian noises (standard deviations ranging from 0.5 to 0.7) for stochasticity under provided 100 expert trajectories.

D.2. Model estimation error and reward learning

In this study, we empirically evaluate the effect of dynamic model learning errors on our method's performance, extending the theoretical analysis presented in subsection 4.3. To isolate the impact of model errors specifically on reward learning, we use SAC on real trajectories for policy optimization, thereby removing any influence of model errors on trajectory generation that would typically affect model-based policy optimization. To quantify the relationship between model errors and performance, we standardize the ensemble model architecture as 2-layer MLPs with varying hidden layer dimensions from 8 to 256 to adjust model capacity. Our experiments are conducted in `HalfCheetah-v4` with random, policy-unknown Gaussian noise (mean 0 and standard deviation 0.5), as described in section 5. From Figure 4 and Figure 3, we observe the general trend that as modeling error decrease together with increasing capacity of the model structure, performances also increases, which is obvious when hidden dimension bumps up from 8 to 16 and 16 to 32. As transition model error narrows down, the performance improvement also becomes less obvious.

D.3. Does model-based trajectories generation help?

In this study, we empirically investigate the effectiveness of model-based policy optimization on our transition-aware reward shaping IRL framework. We compare three off-policy approaches namely DAC (Kostrikov et al., 2018), transition-aware reward shaping with pure SAC for policy optimization (labeled as $mbirl\_sac$), and our original transition-aware reward shaping with model-based technique

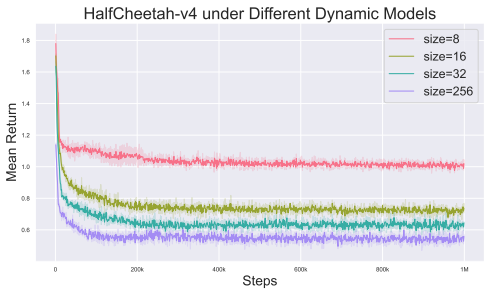

Figure 3: Transition model learning error diagram averaging across three seeds for 10 expert trajectories in `HalfCheetah-v4`.

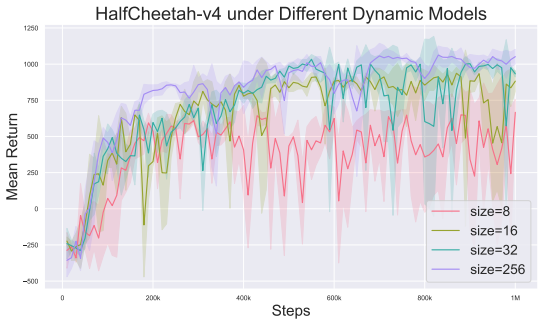

Figure 4: Training return diagram averaging across three seeds for 10 expert trajectories in `HalfCheetah-v4`.

for policy optimization. Noted that synthetic data is also not used in reward learning in $mbirl\_sac$ approach. We conduct the experiment in stochastic `Hopper-v4` with 1000 expert trajectories. From Figure 5 and Table 6, we can tell that both methods using transition-aware reward shaping have much better performance and sample efficiency compared to DAC. In terms of performance, both methods perform at a similar level. However, as the synthetic trajectories generation boosts the training process, our model-based method has better sample efficiency than the pure SAC-based method.

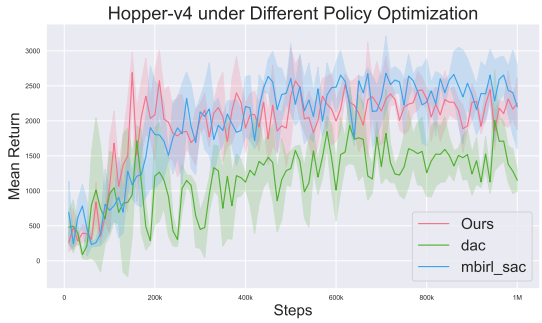

Figure 5: Performance diagram averaging across three seeds for different algorithms in `Hopper-v4` with 1000 expert trajectories provided. DAC is in green color; $mbirl\_sac$ is in blue; Our method is in red.

Method	Performance
DAC	$2007.1_{\pm 719.7}$
$mbirl\_sac$	$2694.5_{\pm 77.5}$
Ours	$2798.8_{\pm 82.9}$

Table 6: Best performance of three methods in stochastic `Hopper-v4` environment with under provided 1000 expert trajectories.

Appendix E. Algorithmic Framework

Algorithm 1 Model-Enhanced IRL (ME-IRL)

Input: Expert buffer $\mathcal{D}_{exp}$ (collected expert trajectories), total training steps N, starting step for training policy STARTING_STEP, horizon length H for synthetic trajectory generation.
Output: Learned policy π optimized to mimic expert behavior.
Initialize policy π (random or pre-trained), discriminator D_θ, buffers $\mathcal{D}_{env}$ (real environment samples), $\mathcal{D}_{gen}$ (synthetic samples), and transition model $\hat{\mathcal{T}}$ (random initialization).
for step t in {1, . . . , N} **do**
  Interact with the environment using policy π to collect new state-action pairs.
  Add the collected state-action pairs (s_t, a_t, s_{t+1}) to the real environment buffer $\mathcal{D}_{env}$.
  if $t <$ STARTING_STEP **then**
    Pre-train transition model: Train $\hat{\mathcal{T}}$ on batches of samples from $\mathcal{D}_{env}$ using maximum likelihood estimation (MLE) loss to stabilize the transition model early.
  else
    Step 1: Update Transition Model and Generate Synthetic Data
    Update the transition model $\hat{\mathcal{T}}$ using maximum likelihood loss (MLE) on $\mathcal{D}_{env}$.
    Use $\hat{\mathcal{T}}$ to generate H-step synthetic trajectories and store them in the synthetic buffer $\mathcal{D}_{gen}$.
    Step 2: Update Discriminator
    Sample mini-batches of state-action pairs from $\mathcal{D}_{exp}$ (expert buffer) and $\mathcal{D}_{env}$ (environment buffer) with varying ratio.
    Train the discriminator D_θ using cross-entropy loss to classify expert data (from $\mathcal{D}_{exp}$) versus policy-generated data (from $\mathcal{D}_{env}$) as described in Equation 3.
    Step 3: Policy Optimization with Mixed Data
    Sample state-action batches from $\mathcal{D}_{env}$ and $\mathcal{D}_{gen}$ with a varying ratio that increases the use of $\mathcal{D}_{gen}$ as $\hat{\mathcal{T}}$ becomes more accurate.
    Compute the reward $\hat{R}_\theta$ for the sampled state-action pairs using the discriminator.
    Update the policy π using Soft Actor-Critic (SAC) with the computed reward $\hat{R}_\theta$ as the optimization objective.
  end if
end for
Return: Optimized policy π.

Appendix F. Implementation Details

For our framework, we use two identical 2-layer Multi-Layer Perceptrons (MLPs) with 100 hidden units and ReLU activations for both the reward function R and the shaping potential function ϕ. To initialize the replay buffer for both **DAC** and ours, we collect 1,000 steps samples in `InvertedPendulum-v4` and `InvertedDoublePendulum-v4`, and 10,000 steps samples in `Hopper-v4`, `HalfCheetah-v4`, and `Walker2d-v4` with initial policy. During this pre-training phase, we also update the transition model at each step to mitigate divergence might happen at the beginning of the training. Additionally, the transition model is only trained using samples from real environment buffer $\mathcal{D}_{env}$ in policy optimization section before actor and critics updates during training phase. As discussed in Section 4, the size of the synthetic data buffer $\mathcal{D}_{\text{gen}}$ and the ratio of samples drawn from it increase as the model accuracy improves. Both parameters increase linearly with training steps, up to a maximum synthetic-to-real data ratio of 0.5 per training step and a maximum buffer size of 1 million samples in $\mathcal{D}_{\text{gen}}$. For consistency in comparisons, we used similar network structures and hyper-parameters for **AIRL, GAIL, and DAC** baselines, which we reference the implementations from Arulkumaran and Lillrank (2024) and Gleave et al. (2022). Detailed hyper-parameters for these networks are provided in the table below. For on-policy baselines **AIRL** and **GAIL**, the rollout length is set to 1,000 for `InvertedPendulum-v4` and `InvertedDoublePendulum-v4`, and 5,000 for `Hopper-v4`, `Walker2d-v4`, and `HalfCheetah-v4`. For the SAC and PPO policy optimization components, we reference implementations from the `CleanRL` repository (Huang et al., 2022). Our implementation of Dreamer-v2 and RSSM are based on SheepRL (EclecticSheep et al., 2023).

Table 7: Hyper-parameters table.

Hyper-parameter	Value
Seeds	0, 5, 10
Buffer Size	1M
Batch Size	128
Max Grad Norm	10
Starting Steps	1,000/10,000
Global Timesteps	100k/1M
Discount Factor	0.99
Model-based Policy Optimization	
Learning Rate for Actor	3e-4
Learning Rate for Critic	3e-4
Learning Rate for Model	3e-4
Network Layers	3
Policy Network Neurons	[64, 64]
Critic Network Neurons	[128, 128]
Model Network Neurons	[256, 256]
Ensemble Numbers	7
Elites Numbers	5
Activation	Tanh(Policy)/ReLU
Optimizer	Adam
Initial Entropy	$-\|\mathcal{A}\|$
Learning Rate for Entropy	3e-4
Train Frequency for Actor	1
Train Frequency for Critic	1
Train Frequency for Model	1
Synthetic and Real Data Mix Coef	0.5
Horizon(H)	4
Adversarial Discriminator	
Learning Rate	3e-4
R Network Neurons	[100, 100]
ϕ Network Neurons	[100, 100]
Optimizer	Adam
Loss	Binary Cross-Entropy

Appendix G. MuJuCo Graphical Results

Below are the testing return diagrams from stochastic MuJoCo Environments. Since AIRL and GAIL use distinct environmental training steps from DAC and our method, Figure 6 provides a clear comparison under `10M` landscape while the rest of the graphs show the sample efficiencies for all algorithms under `1M` landscape.

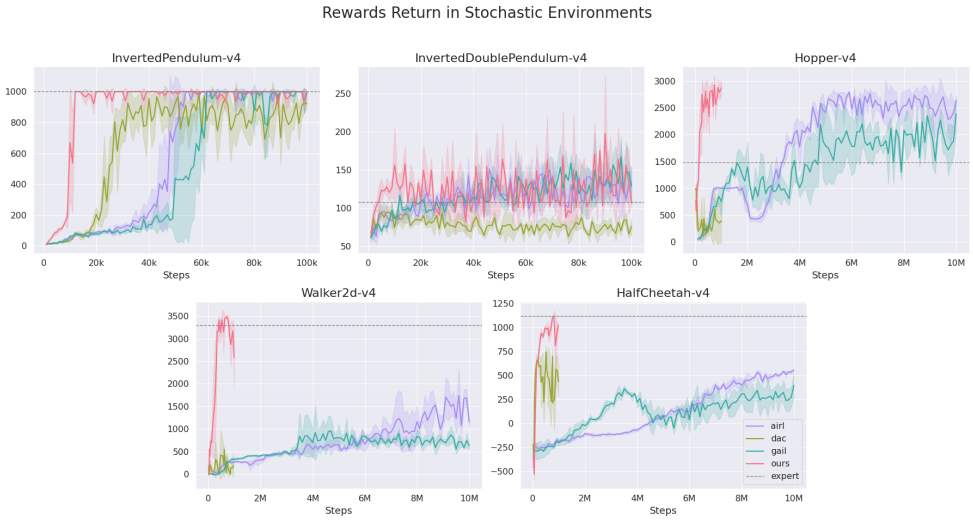

Figure 6: Training curves of all 4 methods in 5 different **stochastic** environments with 100 expert trajectories. For better comparison in sample efficiency, graph is presented under `10M` landscape.

Appendix H. Supplementary Material

H.1. Atari Experiment Details

In this section, we explain the implementation details for the Atari experiments. We follow SheepRL (Eclectic-Sheep et al., 2023) for our implementation for the expert policy (Dreamer-v2 (Hafner et al., 2020)) and the baseline IRL comparison (CNN-AIRL(Tucker et al., 2018)). As for our own algorithm, we replace the ensemble dynamic model with RSSM(Hafner et al., 2019) to capture the high-dimensional inputs from Atari environments. To address the discrete action space, we use categorical distributions as actor networks for all algorithms and change the policy update logic accordingly. Due to computation resource constraint, we test our algorithms on two distinct environments - `SpaceInvaders-v5` and `BattleZone-v5` instead of the full set of Atari2600 environments. All algorithms are trained for `400k` steps. We report the mean return for `SpaceInvaders-v5` in Table 4 and `BattleZone-v5` in Table 8. The hyper-parameters of the RSSM can be found in Table 9.

Method	Retern
Expert	$22300.0_{\pm 3662.0}$
CNN-AIRL	$1200.0_{\pm 748.3}$
Ours	$14400.0_{\pm 5607.1}$

Table 8: Mean return over 10 evaluation episodes after `400k` steps on `BattleZone-v5`.

Table 9: RSSM hyper-parameters for Atari experiments.

Policy Optimization for discrete actor	
Discrete Latent Size	32
Stochastic Latent Size	32
KL Balancing α	0.8
Free Nats	1.0 (averaged)
KL Regulariser Scale	1.0
Encoder / Observation Model	
CNN Channel Multiplier	48
MLP Layers	4
Dense Units	400
Activation	ELU / ELU
Layer Norm	False
Recurrent Model	
Recurrent State Size	600
Dense Units	400
Activation	ELU
Layer Norm	True
Transition & Representation Models	
Hidden Size	600
Activation	ELU
Layer Norm	False
Discount Model	
Dense Units	400
Activation	ELU / ELU
Layer Norm	False

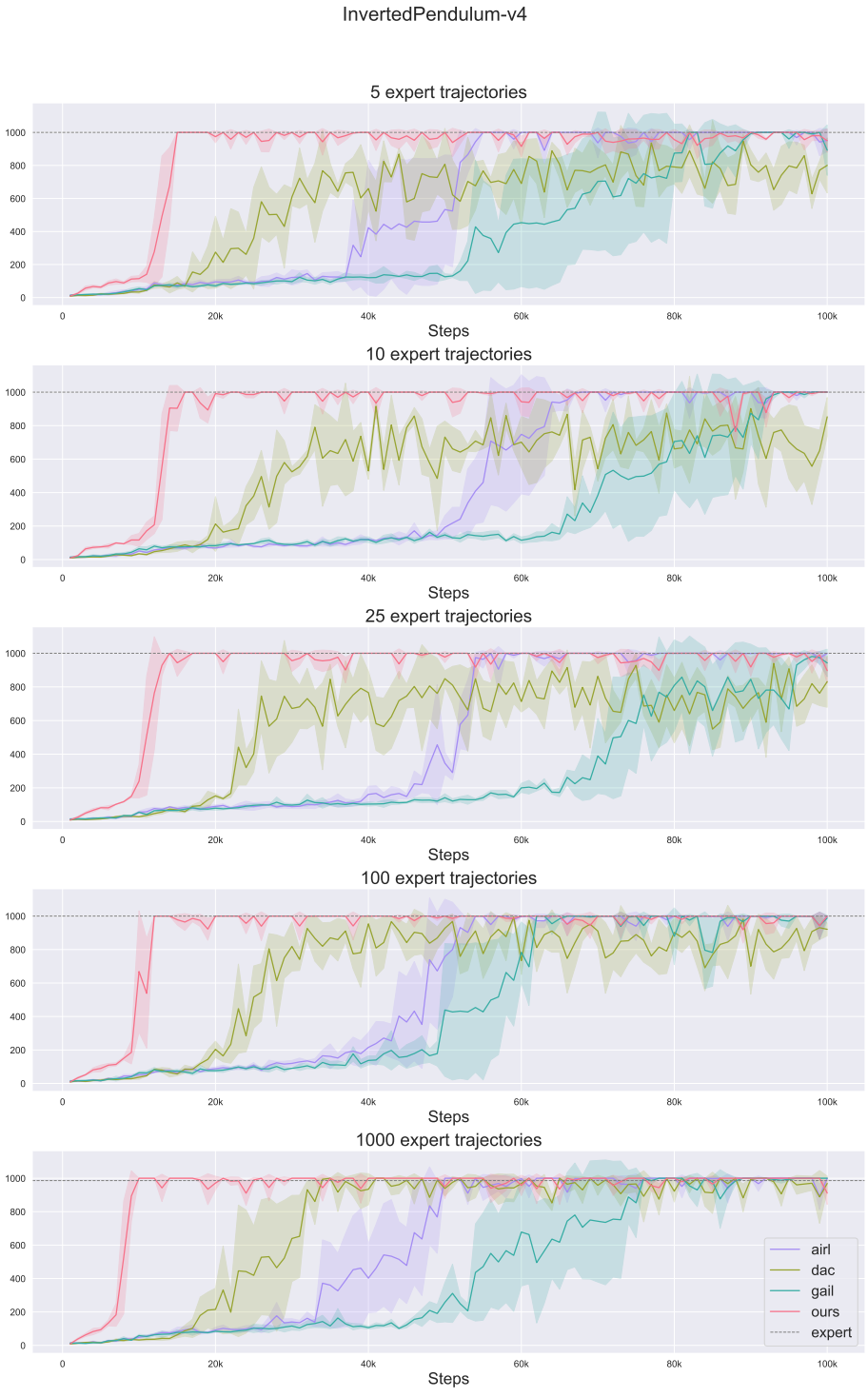

Figure 7: Training return diagram averaging across three seeds for different numbers of expert trajectories in Stochastic `InvertedPendulum-v4`.

Figure 8: Training return diagram averaging across three seeds for different numbers of expert trajectories in Stochastic `InvertedDoublePendulum-v4`.

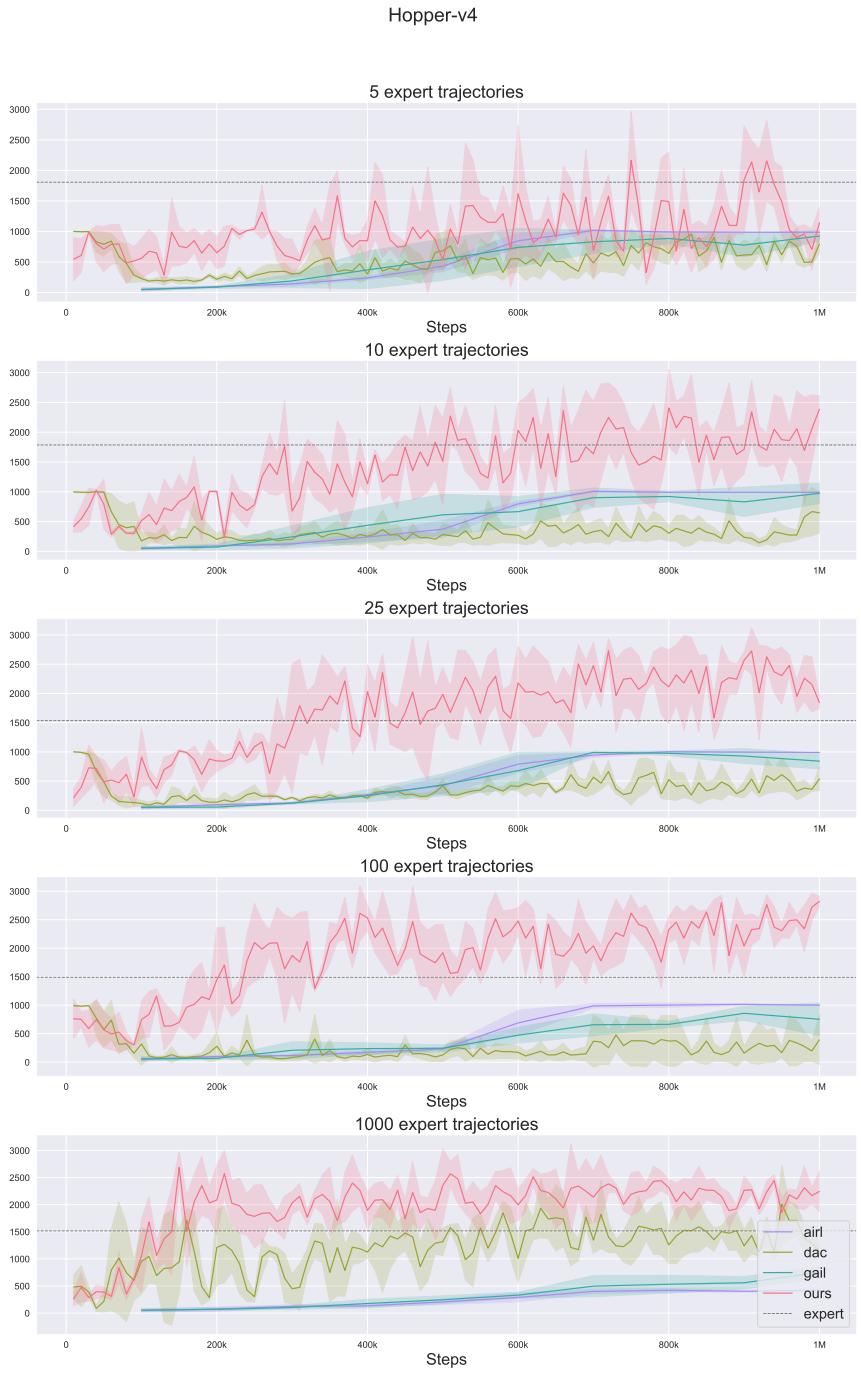

Figure 9: Training return diagram averaging across three seeds for different numbers of expert trajectories in Stochastic `Hopper-v4`.

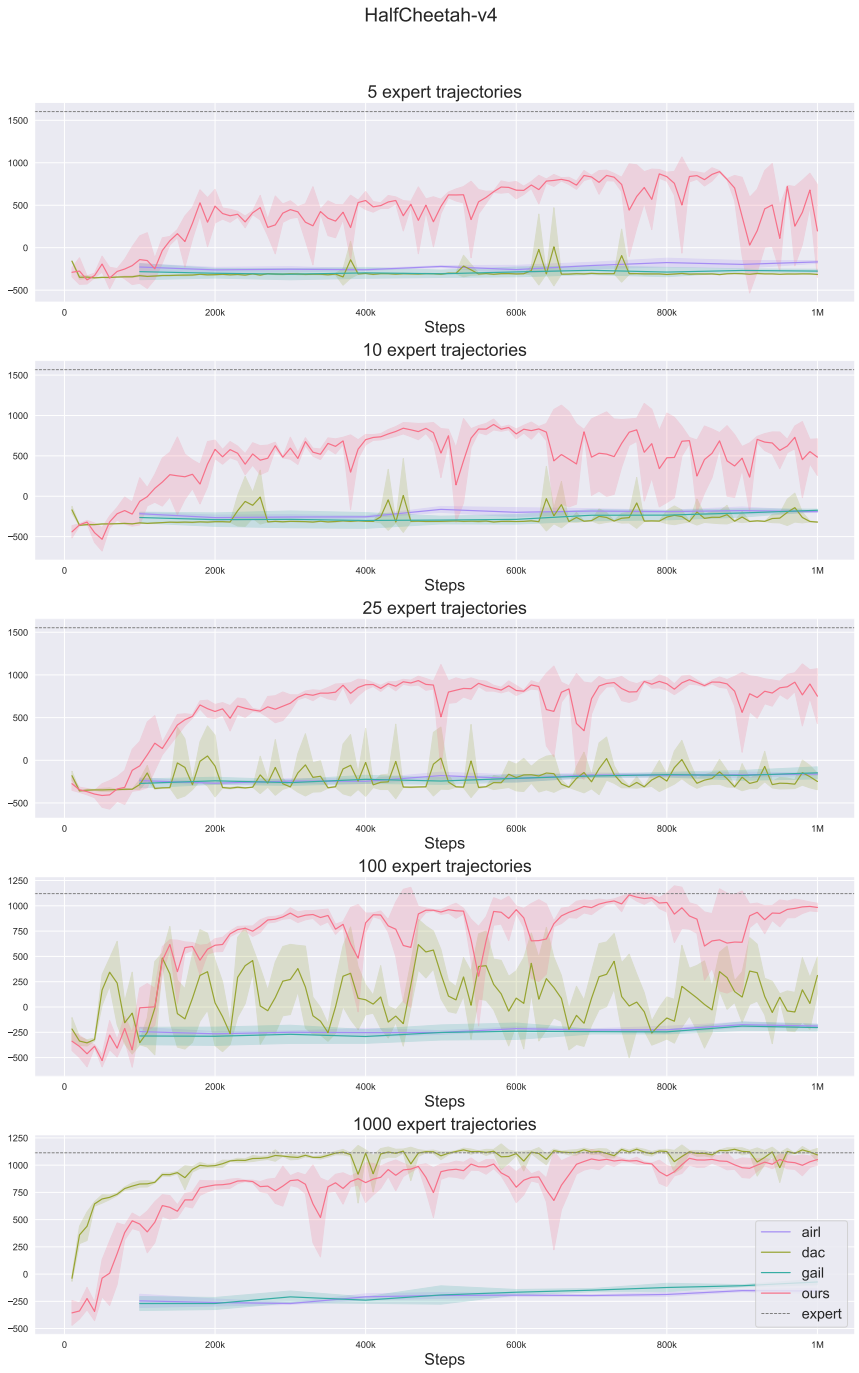

Figure 10: Training return diagram averaging across three seeds for different numbers of expert trajectories in Stochastic `HalfCheetah-v4`.

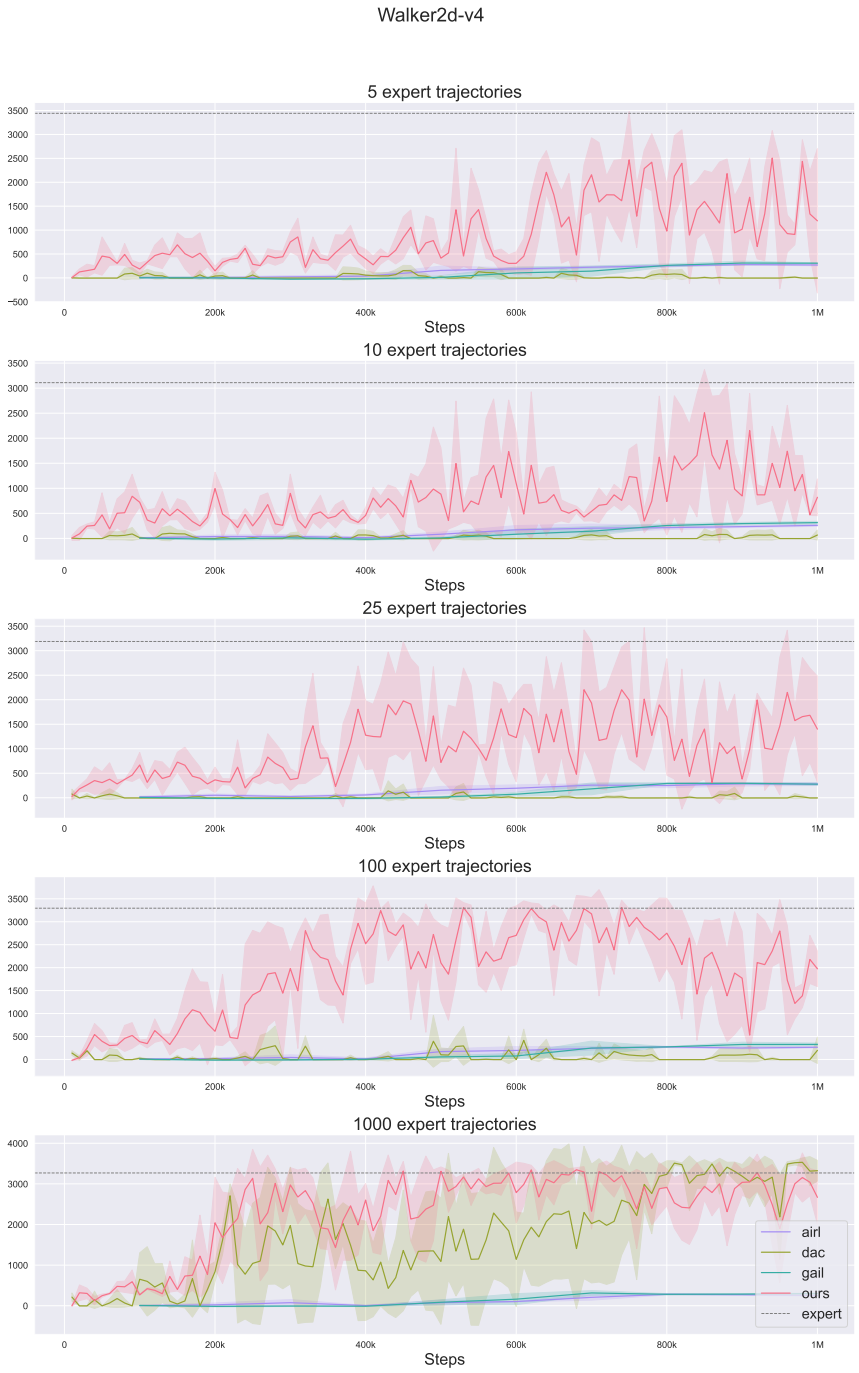

Figure 11: Training return diagram averaging across three seeds for different numbers of expert trajectories in Stochsatic `Walker2d-v4`.