Adversarial Inverse Reinforcement Learning with Self-attention Dynamics Model

Jiankai Sun[1], Lantao Yu[2], Pinqian Dong[3], Bo Lu[1], and Bolei Zhou[1]

***Abstract*—In many real-world applications where specifying a proper reward function is difficult, it is desirable to learn policies from expert demonstrations. Adversarial Inverse Reinforcement Learning (AIRL) is one of the most common approaches for learning from demonstrations. However, due to the stochastic policy, current computation graph of AIRL is no longer end-to-end differentiable like Generative Adversarial Networks (GANs), resulting in the need for high-variance gradient estimation methods and large sample size. In this work, we propose the Model-based Adversarial Inverse Reinforcement Learning (MAIRL), an end-to-end model-based policy optimization method with self-attention. By adopting the self-attention dynamics model to make the computation graph end-to-end differentiable, MAIRL has the low variance for policy optimization. We evaluate our approach thoroughly on various control tasks. The experimental results show that our approach not only learns near-optimal rewards and policies that match expert behavior but also outperforms previous inverse reinforcement learning algorithms in real robot experiments. Code is available at https://decisionforce.github.io/MAIRL/.**

***Index Terms*—Imitation Learning, Learning from Demonstration**

I. Introduction

REINFORCEMENT learning (RL) has emerged as a promising tool for solving complex decision-making tasks from the predefined reward functions [1]. However, defining a reward function that induces the desired behavior can be challenging for many robotic applications such as dexterous manipulation [2] and autonomous driving [3]–[5]. To address the above problem, Inverse Reinforcement Learning (IRL) [6] is proposed to learn reward function from expert demonstrations. The learned reward function is considered to be more generalizable than the learned policy and might achieve better performance in transfer learning [7].

Recently, inspired by GANs, AIRL methods in the model-free setting are proposed to directly learn the policy resulting from the full Markov decision process [8]. The issue of the AIRL rises when training stochastic policies, where the backpropagation flow of gradients from the discriminator to the generator is broken down by the stochasticity of the policy. Hence, the adversarial training framework is no longer end-to-end differentiable, which is different from the GAN training. Thus the high-variance gradient estimator has to be used to address the non-differentiable training in AIRL, making the training unstable and prone to sub-optimal policies.

One intuitive way to improve AIRL is introducing a predictive dynamics model to make the computation graph fully differentiable. The strength of this model-based approach is that it enables updating the policies using the exact gradient from the recovered reward signal, resulting in a lower training variance. However, one typical drawback of the model-based approaches is the approximation error of the parameterized dynamics model. In many complex robotic control applications, learning accurate dynamics of a system from observations remains challenging. Recently, self-attention models bring breakthroughs in natural language processing [9] and computer vision [10] due to its capability of effectively modeling temporal information over a long time horizon. Meanwhile, self-attention dynamics model avoids compressing the whole past into a fixed-size hidden state and does not suffer from vanishing or exploding gradients in the same way as RNNs. Thus our work adopts the self-attention in the dynamics model to capture the transition information with a long time horizon. We show that the proposed self-attention dynamics model consistently matches or outperforms the traditional recurrent architectures (*e.g.*, GRUs), as well as the Feedforward Neural Networks (FNNs), in a range of tasks.

In this work, we integrate Maximum Entropy IRL [11], model-based reinforcement learning, and self-attention [9], into a unified graphical model that bridges the gap between reward inference and learning from demonstrations. Fig. 1 illustrates the proposed method. Experimental results show significant improvements over the existing methods. We summarize our contributions as follows:

1 We introduce a novel framework, Model-based Adversarial Inverse Reinforcement Learning (MAIRL), to make the previous AIRL pipeline end-to-end differentiable. It updates policy using the exact gradient rather than high-variance gradient estimation, leading to more accurate reward function.
2 To address the approximation error in the dynamics model, we propose to use self-attention to enhance the model's ability to process transition information over long time horizons.
3 Extensive experimental results show that the proposed MAIRL framework can recover the reward function for

Manuscript received: October, 16, 2020; Revised January, 6, 2021; Accepted January, 25, 2021.

This paper was recommended for publication by Editor Dana Kulic upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by InnoHK CPII Grant.

[1]J. Sun, B. Lu and B. Zhou are with The Chinese University of Hong Kong, Hong Kong, China. J. Sun and B. Zhou are also affiliated with the Centre for Perceptual and Interactive Intelligence, Hong Kong, China. Email: sj019@ie.cuhk.edu.hk

[2]L. Yu is with Computer Science Department, Stanford University, USA.

[3]P. Dong is with State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan, China.

Digital Object Identifier (DOI): see top of this page.

transfer learning. It can also achieve state-of-the-art performance on the AIRL benchmark. We further deploy the proposed method to a real robotic platform for the tasks of learning from demonstrations.

II. Related Work

Adversarial Methods for Maximum-Entropy Inverse Reinforcement Learning. In contrast to reinforcement learning which relies on pre-defined reward functions, the goal of IRL is to learn the reward function r from the demonstrations so that RL policy can be further learned. Maximum-Entropy IRL [11] models the expert demonstrations as Boltzmann distribution using parametrized reward $r_\xi(\tau)$ as an energy function,

$$p_\xi(\tau) = \frac{1}{Z}\exp(r_\xi(\tau)), \tag{1}$$

where $r_\xi(\tau) = \sum_{i=0}^{T} r_\xi(s_t, a_t)$ is a commutative reward function parameterized by ξ over the given trajectory τ, and $Z = \int \exp(r_\xi(\tau))\mathrm{d}\tau$ is the partition function (normalization constant). Principal benefits of the Maximum Entropy paradigm include the ability to handle expert suboptimality as well as the stochasticity by operating on the distribution over possible trajectories. The main challenge in Maximum-Entropy IRL is that it is generally intractable to compute Z, an integral over the trajectory space. Recent adversarial methods for Maximum-Entropy IRL [8], [12] present Importance Sampling (IS) technique to approximate Z under unknown dynamics. AIRL [12] is a representative model-free adversarial IRL framework based on GAIL [8], maximum entropy IRL framework [11], and Guided Cost Learning (GCL) [2]. Nevertheless, the introduced gradient estimation technique makes AIRL suffer from a high variance of training.

Model-based Reinforcement Learning. As a promising candidate for real-world sequential decision-making problems, model-based reinforcement learning methods have many strengths, such as analytic gradient computation. There is an extensive literature on learning a dynamics model and using the learned model to train a policy via model-based planning. PILCO [13] uses a Gaussian process to model system dynamics. PETS [14] combines uncertainty-aware deep network dynamics models with sampling-based uncertainty propagation. World Models [15] learn latent dynamics in a two-stage process to evolve linear controllers in imagination. Dreamer [16] solves visual locomotion tasks by latent online planning. MGAIL [17] introduces a forward model in GAIL, but its discriminator is unsuitable as a reward since, at optimality, it outputs 0.5 uniformly across all states and actions, which is a less portable representation for transfer. Desired dynamics models should be able to capture long-term time dependencies. Developing internal models with FNNs, RNNs, or latent dynamics to reason about the future has been explored in the works mentioned above.

Different from previous work, we adopt self-attention [9] in the dynamics model. Compared with RNNs, the advantages of self-attention includes avoiding compressing the whole past into a fixed-size hidden state, less total computational complexity per layer, and more parallelizable computations. In the experiments, we will demonstrate the strengths of the self-attention dynamics model.

III. Background

Consider a Markov Decision Process (MDP) represented as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \rho_0, \gamma)$ with state space $\mathcal{S}$, action-space $\mathcal{A}$, dynamics $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, reward function $r(s, a)$, initial state distribution $\rho_0 : \mathcal{S} \to \mathbb{R}$, and discount factor $\gamma \in (0, 1)$. Let π be a stochastic policy that takes a state and outputs a distribution over actions. Let τ and τ_E denote trajectories (i.e., sequences of state-action pairs $(s_0, a_0, \cdots, s_T, a_T)$) generated by a policy π and an expert policy π_E, respectively, where T denotes the time horizon. Our method is built upon AIRL [12] and MGAIL [17], so we first introduce them briefly as follows.

A. Adversarial Inverse Reinforcement Learning (AIRL)

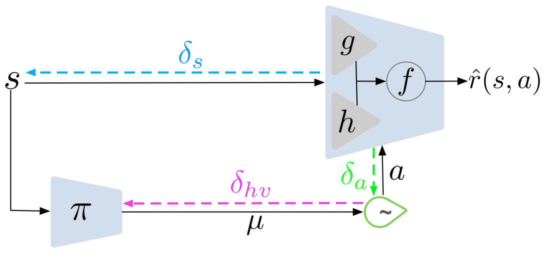


Fig. 2: Computation graph of the model-free AIRL

Inspired by GANs, AIRL [12] is a model-free adversarial learning framework that learns a generative model from the two-player adversarial training of the generator and discriminator.

The discriminator is trained to minimize the cross-entropy loss between expert demonstrations and generated samples:

$$\begin{aligned}\mathcal{L}(D) = &-\sum_{t=0}^{T}\mathbb{E}_{s^{(t)},a^{(t)}\sim\tau_E}[\log D(s^{(t)}, a^{(t)})]\\ &-\sum_{t=0}^{T}\mathbb{E}_{s'^{(t)},a'^{(t)}\sim\mathcal{D}}[1 - \log D(s'^{(t)}, a'^{(t)})],\end{aligned} \tag{2}$$

where D is the discriminator that performs the binary classification to distinguish between samples generated by π and π_E, $\mathcal{D}$ is the experience buffer of π, τ_E are expert trajectories generated by π_E. In contrast to GAIL [8], which can not recover the reward function, AIRL can recover the reward function along with the policy by imposing the following form on the discriminator (for the sake of simplicity, we ignore the parameter notations):

$$D(s, a) = \frac{\exp\{f(s, a)\}}{\exp\{f(s, a)\} + \pi(a|s)}, \tag{3}$$

where $f(\cdot)$ is restricted to a reward approximator $g(\cdot)$ and a shaping term $h(\cdot)$. The policy π is trained to maximize the entropy-regularized discriminative reward:

$$\begin{aligned}\hat{r}(s, a) &= \log(D(s, a)) - \log(1 - D(s, a))\\ &= f(s, a) - \log\pi(a|s).\end{aligned} \tag{4}$$

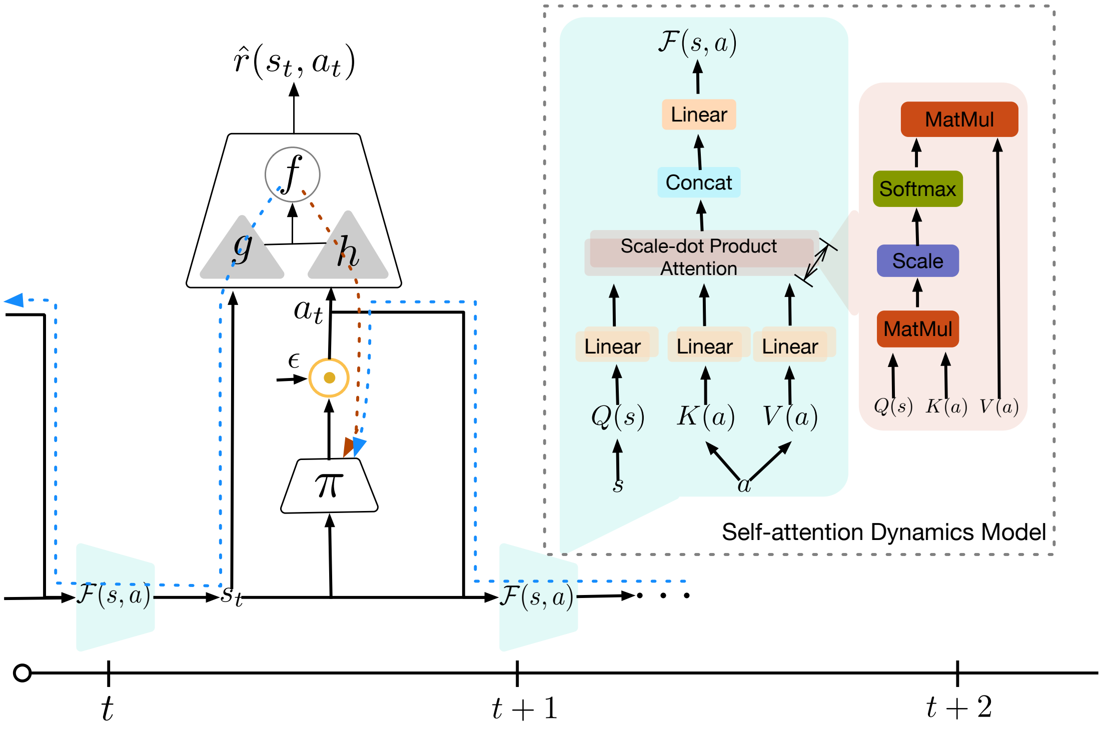


Fig. 1: **Computation graph of Model-based Adversarial Inverse Reinforcement Learning (MAIRL).** At time t of the forward pass, action a_t is sampled from the output action distribution of π with re-parametrization trick: $a_t = \pi(s_t) + \epsilon \odot \sigma$, where $\epsilon \sim \mathcal{N}(0, 1)$. The next state $s_{t+1} = \mathcal{F}(s_t, a_t)$ is computed using the forward model $\mathcal{F}$, and the entire process repeats for time $t + 1$. For the backward pass, the gradient of π comprises the direct backpropagation through the action stream from the differentiable entropy-regularized discriminative reward $\hat{r}$ and the backpropagation through state stream of future time-steps via the differentiable forward model $\mathcal{F}$. $f(\cdot)$ is restricted to a reward approximator $g(\cdot)$ and a shaping term $h(\cdot)$

Thus, when $\hat{r}(s, a)$ is summed over entire trajectories, the entropy-regularized policy objective is obtained,

$$\mathbb{E}_{\pi}\left[\sum_{t=0}^{T} \hat{r}(s_t, a_t)\right] = \mathbb{E}_{\pi}\left[\sum_{t=0}^{T} f(s_t, a_t) - \log \pi(a_t|s_t)\right]. \quad (5)$$

However, one shortcoming of the new game defined in AIRL is that it can no longer be solved using the standard gradient descent/ascent because generator G (the policy π) is now *stochastic*. As illustrated in Fig. 2, gradient δ_s in model-free AIRL can not be adopted to update policy. In the back-propagation phase, the gradient δ_s is discarded for policy updating, while the gradient δ_a is blocked at the stochastic sampling unit, where a high-variance gradient estimation δ_{hv} is used. Discarding δ_s can be disastrous. Hence, model-free AIRL updates policy without effective use of δ_s, and suffers from high variance. Owing to stochastic properties of the policy, the exact form of Equation (5) is given by $\mathbb{E}_{s\sim\rho_\pi(s)}\mathbb{E}_{a\sim\pi(\cdot|a)}[\hat{r}(s, a)]$, instead of $\mathbb{E}_{s\sim\rho}[\hat{r}(s, \pi(s)]$ if π was deterministic. Hence, assuming $\pi = \pi_{\theta_g}$, it is no longer possible to apply gradient ascent to differentiate Equation (5) w.r.t. θ_g. An alternative solution is to obtain a gradient estimation by REINFORCE method [18]:

$$\nabla_{\theta_g}\mathbb{E}_{\pi}[\hat{r}(s, a)] \cong \hat{\mathbb{E}}_{\tau_i}[Q(s, a)\nabla_{\theta_g} \log \pi_{\theta_g}(a|s)], \quad (6)$$

where $Q(\hat{s}, \hat{a}) = \hat{\mathbb{E}}_{\tau_i}[\hat{r}(s, a)|s_0 = \hat{s}, a_0 = \hat{a}]$ is the score function of the gradient. Nevertheless, REINFORCE tends to suffer from high training variance, which makes it difficult to work with even after applying variance reduction techniques [19].

B. MGAIL Algorithm

To make the computation of GAIL fully differentiable, MGAIL [17] introduces a forward dynamics model $\mathcal{F}$, in which the information propagates fluently from the discriminator D to the generative model G. For the policy learning process in MGAIL, the gradient of policy π is computed from the Jacobian of the discriminator D:

$$\begin{aligned}
&\left.\frac{\partial D(s_t, a_t)}{\partial \theta}\right|_{s=s_t, a=a_t} \\
=&\left.\frac{\partial D}{\partial a}\frac{\partial a}{\partial \theta}\right|_{a=a_t} + \left.\frac{\partial D}{\partial s}\frac{\partial s}{\partial \theta}\right|_{s=s_t} \qquad (7)\\
=&\left.\frac{\partial D}{\partial a}\frac{\partial a}{\partial \theta}\right|_{a=a_t} + \frac{\partial D}{\partial s}\left(\left.\frac{\partial \mathcal{F}}{\partial s}\frac{\partial s}{\partial \theta}\right|_{s=s_{t-1}} + \left.\frac{\partial \mathcal{F}}{\partial a}\frac{\partial a}{\partial \theta}\right|_{a=a_{t-1}}\right).
\end{aligned}$$

For a multi-step computation graph, the policy gradient objective is given by $J(\theta) = \mathbb{E}\left[\sum_{t=0}\gamma^t D(s_t, a_t)|\theta\right]$. $J(\theta)$ can be differentiated over a trajectory of (s, a, s') transitions by recursively applying Equations (8) and (9) starting from $t = T$ all the way down to $t = 0$:

$$\frac{\partial J}{\partial s} = \mathbb{E}_{p(a|s)}\mathbb{E}_{p(s'|s,a)}\left[\frac{\partial D}{\partial s} + \frac{\partial r}{\partial a}\frac{\partial \pi}{\partial s} + \gamma\frac{\partial J'}{\partial s'}\left(\frac{\partial \mathcal{F}}{\partial s} + \frac{\partial \mathcal{F}}{\partial a}\frac{\partial \pi}{\partial s}\right)\right], \quad (8)$$

$$\frac{\partial J}{\partial \theta} = \mathbb{E}_{p(a|s)}\mathbb{E}_{p(s'|s,a)}\left[\frac{\partial D}{\partial a}\frac{\partial \pi}{\partial \theta} + \gamma\left(\frac{\partial J'}{\partial s'}\frac{\partial \mathcal{F}}{\partial a}\frac{\partial \pi}{\partial \theta} + \frac{\partial J'}{\partial \theta}\right)\right]. \quad (9)$$

Define *policy likelihood ratio* $\phi(s,a)$ and *state distribution likelihood ratio* $\psi(s)$ as

$$\phi(s,a) = \frac{p(a|s,\pi_E)}{p(a|s,\pi)}, \quad \psi(s) = \frac{p(s|\pi_E)}{p(s|\pi)}, \tag{10}$$

then the partial derivatives of D can be reformulated as the Jacobian of the discriminator:

$$\frac{\partial D}{\partial a} = -\frac{\phi_a(s,a)\psi(s)}{(1+\phi(s,a)\psi(s))^2}, \frac{\partial D}{\partial s} = -\frac{\phi_s(s,a)\psi(s)+\phi(s,a)\psi_s(s)}{(1+\phi(s,a)\psi(s))^2}. \tag{11}$$

The major disadvantage of MGAIL comes to light when trying to recover robust reward functions. Its discriminator outputs 0.5 uniformly across all states and actions at optimality, thus is unsuitable as a reward function. This makes MGAIL perform poorly when learning policies under the environment with significant variations.

IV. Our Method

To address the shortcomings of previous methods, we propose MAIRL, a differentiable adversarial inverse reinforcement learning algorithm with a forward dynamics model. MAIRL does not suffer from the high variance occurred in AIRL because MAIRL updates policy parameters with exact gradient thus REINFORCE method is not required. As a model-based approach, in contrast to MGAIL, MAIRL can recover a generalizable and portable reward function and achieve better performance on tasks that require transfer learning.

A. Self-attention Dynamics Model

To make the framework end-to-end differentiable without high-variance gradient estimation, the crucial component is the dynamics model. However, learning an accurate dynamics model is challenging [20]. Thus, we introduce self-attention dynamics model $s_{t+1} = \mathcal{F}(s_t,a_t)$, which can performs accurate long-term temporal predictions, enabling MAIRL to learn successful behaviors.

Our self-attention dynamics model (cf. Fig. 1) adopts Scaled Dot-Product [9] to compute attention $A(s_t,a_t)$ by

$$A(Q(s_t),K(a_t),V(a_t)) = \text{softmax}\left(\frac{Q(s_t)K(a_t)^T}{\sqrt{d_k}}\right)V(a_t), \tag{12}$$

where queries $Q(s_t)$ are state embedding of dimension d_q, keys $K(a_t)$ are action embedding of dimension d_k, and values $V(a_t)$ are action embedding of dimension d_v. Note that $K(a_t)$ and $V(a_t)$ use independent embedding network and do not share parameters. To jointly attend the information from different representation subspaces of state embeddings and action embeddings, multi-head attention module $M(s_t,a_t)$ is adopted:

$$M(Q(s_t),K(a_t),V(a_t)) = \text{Concat}(\text{head}_1,\cdots,\text{head}_h)W^O, \tag{13}$$

where $\text{head}_i = A(Q(s_t)W_i^{Q(s_t)}, K(a_t)W_i^{K(a_t)}, V(a_t)W_i^{V(a_t)})$. The projections are parameter matrices $W_i^{Q(a)} \in \mathbb{R}^{d_{model}\times d_k}$, $W_i^K \in \mathbb{R}^{d_{model}\times d_k}$, $W_i^V \in \mathbb{R}^{d_{model}\times d_v}$ and $W^O \in \mathbb{R}^{hd_v\times d_{model}}$. Finally, we get the forward dynamics model after adding a decoder Dec$[\cdot]$ to the output of self-attention module:

$$s_{t+1} = \mathcal{F}(s_t,a_t) = \text{Dec}[M(Q(s_t),K(a_t),V(a_t))] \tag{14}$$

Forward dynamics model $\mathcal{F}$ is trained on a trajectory of (s,a,s') transitions sampled from buffer $\mathcal{D}$ via maximizing the maximum likelihood

$$\mathcal{L}(\theta_\mathcal{F}) = \mathbb{E}_{(s,a,s')\sim\mathcal{D}}[\log \mathcal{F}(s'|s,a)] \tag{15}$$

Algorithm 1 Model-based Adversarial Inverse Reinforcement Learning (MAIRL)

```
Require: Expert trajectories $\tau_E$, Experience buffer $\mathcal{D}$, initial forward dynamics model parameters $\theta_\mathcal{F}$, initial policy and discriminator parameters $\theta_g$, $\theta_d$, terminal time $T$, entropy-regularized discriminative reward $\hat{r}$.
Ensure: Weights $\theta_\mathcal{F}$, $\theta_g$, $\theta_d$
  $\theta_\mathcal{F}$, $\theta_g$, $\theta_d \leftarrow$ Initialize parameters.
  repeat
    for $t = 0$ to $T$ do
      Collect data with $\pi$ in real environment: $\mathcal{D} = \mathcal{D} \cup \{(s_t, a_t, s_t^{'})\}$.
    end for
    Train forward dynamics model $\mathcal{F}$ on buffer $\mathcal{D}$ via maximum likelihood: $\theta_\mathcal{F} \leftarrow \arg\max_{\theta_\mathcal{F}} \mathbb{E}_\mathcal{D}[\log \mathcal{F}(s'|s,a)]$ ((cf. Section IV-A)
    Train discriminator model $D_{\theta_d}$ using $\mathcal{D}$ by minimizing Equation (2) ((cf. Section IV-B)
    Set $j_{\theta_g}^{'} = 0, j_s^{'} = 0$
    for $t = T$ down to 0 do
      $j_{\theta_g} = [\hat{r}_a\pi_{\theta_g} + \gamma(j_{s'}^{'}\mathcal{F}_a\pi_{\theta_g} + j_{\theta_g}^{'})]\Big|_\epsilon$, $j_s = [\hat{r}_s + \hat{r}_a\pi_s + \gamma j_{s'}^{'}(\mathcal{F}_s + \mathcal{F}_a\pi_{\theta_g})]\Big|_\epsilon$
    end for
    Update $\theta_g$ by applying gradient $j_{\theta_g}^0$ ((cf. Section IV-C)
  until convergence of parameters ($\theta_\mathcal{F}$, $\theta_g$, $\theta_d$)
  return $\theta_\mathcal{F}$, $\theta_g$, $\theta_d$
```

B. Differentiable Entropy-regularized Discriminative Reward

We start by analyzing the characteristics of the entropy-regularized discriminative reward function, which is different from MGAIL. Given the definition of $\hat{r}(s,a)$ in Equation 4, we can get the formula of $\frac{\partial\hat{r}}{\partial s}$ and $\frac{\partial\hat{r}}{\partial a}$, which will be used in Section IV-C:

$$\begin{aligned}\frac{\partial\hat{r}}{\partial s} &= \frac{1}{D}\frac{\partial D}{\partial s} + \frac{1}{1-D}\frac{\partial D}{\partial s},\\ \frac{\partial\hat{r}}{\partial a} &= \frac{1}{D}\frac{\partial D}{\partial a} + \frac{1}{1-D}\frac{\partial D}{\partial a},\end{aligned} \tag{16}$$

From the Jacobian of D in Equation 11, the partial derivatives of $\hat{r}$ are

$$\frac{\partial\hat{r}}{\partial s} = -\frac{\phi_s(s,a)\psi(s)+\phi(s,a)\psi_s(s)}{\phi(s,a)\psi(s)}, \frac{\partial\hat{r}}{\partial a} = -\frac{\phi_a(s,a)\psi(s)}{\phi(s,a)\psi(s)} \tag{17}$$

C. Policy Learning in MAIRL

Different from MGAIL, our MAIRL updates policy from reward function $\hat{r}(s,a)$ rather than the direct output of discriminator D. For the policy learning process in MAIRL, the gradient of policy π comprises two parts: 1) the direct backpropagation through action stream from the differentiable reward $\hat{r}$, and 2) the backpropagation through the state stream of future time-steps via the differentiable forward model.

a) Backpropagating through State Stream via a Forward Model. : In the model-based approach, s_t can be written as a function of the previous state and action: $s_t = \mathcal{F}(s_{t-1}, a_{t-1})$, where $\mathcal{F}$ is the forward model (cf. Fig. 1). Using the law of total derivatives, we get the gradient backpropagated from $\hat{r}(s,a)$ is:

$$\begin{aligned} &\nabla_\theta \hat{r}(s_t, a_t)\Big|_{s=s_t, a=a_t} \\ =&\frac{\partial \hat{r}}{\partial a}\frac{\partial a}{\partial \theta}\Big|_{a=a_t} + \frac{\partial \hat{r}}{\partial s}\left(\frac{\partial \mathcal{F}}{\partial s}\frac{\partial s}{\partial \theta}\Big|_{s=s_{t-1}} + \frac{\partial \mathcal{F}}{\partial a}\frac{\partial a}{\partial \theta}\Big|_{a=a_{t-1}}\right). \end{aligned} \tag{18}$$

Note that $\hat{r}(s_t, a_t)$ is not the direct output of discriminator D, which is different from MGAIL. The policy gradient objective for a multi-step computation graph is given by $J(\theta) = \mathbb{E}\left[\sum_{t=0}\gamma^t \hat{r}(s_t,a_t)|\theta\right]$. Similar to SVG [21], we can differentiate $J(\theta)$ over a trajectory of (s,a,s') transitions by recursively applying Equations (19) and (20) starting from $t = T$ all the way down to $t = 0$ and thus the final policy gradient $\nabla_\theta J$ can be calculated as

$$\frac{\partial J}{\partial s} = \mathbb{E}_{p(a|s)}\mathbb{E}_{p(s'|s,a)}\left[\frac{\partial \hat{r}}{\partial s} + \frac{\partial \hat{r}}{\partial a}\frac{\partial \pi}{\partial s} + \gamma\frac{\partial J'}{\partial s'}\left(\frac{\partial \mathcal{F}}{\partial s} + \frac{\partial \mathcal{F}}{\partial a}\frac{\partial \pi}{\partial s}\right)\right], \tag{19}$$

$$\frac{\partial J}{\partial \theta} = \mathbb{E}_{p(a|s)}\mathbb{E}_{p(s'|s,a)}\left[\frac{\partial \hat{r}}{\partial a}\frac{\partial \pi}{\partial \theta} + \gamma\left(\frac{\partial J'}{\partial s'}\frac{\partial \mathcal{F}}{\partial a}\frac{\partial \pi}{\partial \theta} + \frac{\partial J'}{\partial \theta}\right)\right]. \tag{20}$$

b) Backpropagation through Action Stream. : For continuous action distribution, assume a stochastic Gaussian policy $\pi_\theta(a|s) = \mu_\theta(s) + \sigma_\theta^2(s) \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0,1)$, mean and variance are given by deterministic functions $\mu_\theta, \sigma_\theta$. $\odot$ denotes the element-wise product. The derivative of the expected value of $\hat{r}(s,a)$ with respect to θ is

$$\begin{aligned} \nabla_\theta \mathbb{E}_{\pi(a|s)}\hat{r}(s,a) &= \mathbb{E}_{\rho(\epsilon)}\nabla_a \hat{r}(s,a)\nabla_\theta \pi_\theta(a|s) \\ &\cong \frac{1}{M}\sum_{i=1}^{M}\nabla_a \hat{r}(s,a)\nabla_\theta \pi_\theta(a|s)\Big|_{\epsilon=\epsilon_i}. \end{aligned} \tag{21}$$

For discrete action distribution, *concrete distribution* [22], [23] can be used here to relax the discrete action distribution to be continuous and differentiable with reparameterization trick:

$$a_i = \frac{\exp[(\log \pi(a_i|s) + g_i)/\tau]}{\sum_{j=1}^{k}\exp[(\log \pi(a_j|s) + g_j)/\tau]}, \tag{22}$$

where τ is the temperature hyper-parameter, $g_i = -\log(-\log(U^i))$ is the ith Gumbel random variable, U^i is a uniform random variable. In [22], it is proved that $p(\lim_{\tau\to 0} a_i = 1) = \log \pi(a_i|s)/(\sum_{j=0}^{n}\log \pi(a_j|s))$, making this relaxation unbiased once converged. This renders $\hat{r}(s,a)$ differentiable for discrete action distribution. **Algorithm 1** shows the complete training procedure.

V. Experiments

We first verify that our method can recover the ground-truth reward function in a diagnostic environment more accurately than the previous inverse RL methods. We then evaluate MAIRL on the continuous control tasks and show that it achieves better performances in both transfer learning and imitation learning settings. Finally, the proposed method is demonstrated on a robot platform UR5 with a higher success rate on the Reaching and Grasping task. We compare the average episode reward of different methods in simulation and report the average success rate of different methods with UR5 robotic arm. These metrics are computed for each method averaged over 100 test runs.

A. Training Details

For a fair comparison, our imitation learning framework shares the same training hyperparameters with baselines such as AIRL and GAIL, if not stated otherwise. We use a two-layer ReLU network with 32 units for the discriminator. For MAIRL and AIRL, the reward approximator $g(\cdot)$ and shaping term $h(\cdot)$ are two-layer FC network with 32 units, respectively. For the policy, we also use a two-layer ReLU neural network with 32 units. Entropy regularizer weight is set as 0.1. We use a batch size of 512 steps per update and $T = 100$. The learning rate for the generator and forward dynamics model is 0.0001. The learning rate for discriminator is 0.001. For the Forward Dynamics Model $\mathcal{F}$, we use two-layer fully-connected network with 32 units for embedding Q, K, V and Decoder $Dec[\cdot]$ respectively, which do not share weights. The size of buffer $\mathcal{D}$ is 1M. In this work, we employ 4 parallel attention heads, $d_{model} = 64$. For each of attention head we use $d_q = d_k = d_v = d_{model}/4 = 16$.

B. Discrete Control Tasks

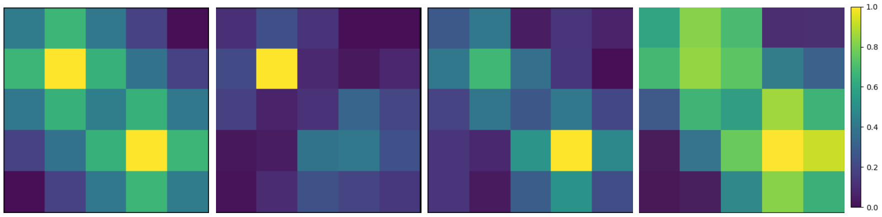
To demonstrate that MAIRL can recover the reward function, we begin with a diagnostic environment GridWorld where the state-space is finite. GridWorld task has 25 states, 5 actions (Stop, Left, Right, Up, Down), and the ground-truth reward function is presented in Fig. 3a. The initial state is randomly generated. The quantitative results of imitation learning with two input modes, state (s) and state-action (s,a), for GridWorld is shown as Table I. For the discrete control task, which is relatively simple, MAIRL performs better than AIRL in both input modes. Qualitatively, the heatmaps of ground truth reward and learned reward for GridWorld with various inverse reinforcement learning algorithms are plotted in Fig. 3. It reveals that MAIRL with a state-only reward function can recover the ground truth reward. Ours and the ground-truth are similar: where the ground-truth reward is high, our method is also high, and where the ground-truth reward is low, our method is also low. LP can only find one high reward area. The transition areas between high rewards and low rewards in AIRL are different from the ground-truth.

C. Policy Learning Performance in Imitation Learning

We further benchmark the MAIRL for imitation learning in various standard continuous control tasks, including

TABLE I: GridWorld Results

Methods	Reward
Expert	15.09 ± 1.56
GAIL	14.01 ± 1.21
GAN-GCL	10.90 ± 2.42
AIRL(s,a)	14.06 ± 1.64
AIRL(s)	13.10 ± 2.31
Ours(s)	$\mathbf{15.02 \pm 1.35}$
Ours(s, a)	15.00 ± 2.09



(a) Ground-truth (b) LP [6] (c) AIRL [12] (d) Ours

Fig. 3: **Reward Maps**. Rewards learned by our method are closer to the ground-truth rewards than those learned by AIRL. LP learns a suboptimal reward function

TABLE II: The evaluation of imitation learning on benchmark control tasks. Mean scores (higher the better) with standard deviation are presented for each method

Methods	Environments	
	HalfCheetah-v2	Ant-v2
Expert	2811.02 ± 448.44	1823.90 ± 655.70
MGAIL	2924.00 ± 23.51	2809.41 ± 12.51
GAIL	2159.00 ± 15.72	2738.72 ± 9.49
GAN-GCL	281.00 ± 32.54	16.96 ± 0.01
AIRL(s,a)	2983.00 ± 251.62	2645.90 ± 41.75
AIRL(s)	1020.00 ± 359.98	809.90 ± 415.80
MAIRL (s,a)	$\mathbf{3186.40 \pm 65.28}$	$\mathbf{3321.5 \pm 67.28}$

TABLE III: Evaluation Success Rate (%) on UR5. Our approach outperforms the other baselines on the tasks of reaching a block with a significant margin, however, it is still much far from perfect completion of the task

Method	Reaching	Grasping
Expert	100	100
Random	0	0
MGAIL(s,a)	45	14
GAIL(s,a)	35	13
AIRL(s,a)	36	10
MAIRL(s,a)	52	21

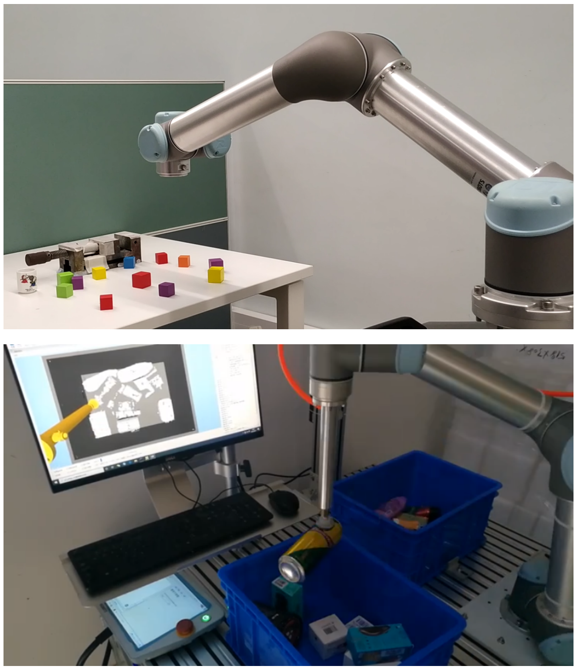


Fig. 4: Reaching and Grasping Task on UR5 Robot Platform

HalfCheetah-v2, Ant-v2. For each task, we provide 50 expert demonstrations generated by a policy trained on a ground-truth reward using TRPO [24]. Table II presents the means and standard deviations of imitation learning performance scores. It can be seen that MAIRL(s, a) surpasses the AIRL(s, a) in most tasks and successfully learns to imitate the expert policy, whereas AIRL(s) and GAN-GCL fail.

D. Training and Evaluation on Real Robot Platform

Current IRL research mostly uses the simulation environment as the evaluation platform. There remains a gap between simulation and real-world applications. To evaluate the methods in real environment, we conduct experiments across two tasks: **Reaching** a block and **Grasping** (cf. Fig. 4). These tasks are easy to solve in simulation but can be difficult for real-world robots [25], [26].

The robot arm system used in the experiment is UR5, a lightweight and flexible industrial robot with six joints manufactured by Universal Robots. We introduce the setup of the imitation learning task on the UR5 so that an off-the-shelf implementation of a standard IRL method can perform effectively and reliably. Firstly, we look at "**Reaching** a block" task on a real robotic platform using UR5 following [27]. Different from their reinforcement learning tasks, our agent learns to achieve tasks by imitating expert demonstrations in our setting. UR5 Reacher is a task with all six joints actuated. The observation space s includes the joint angles, the joint velocities, and the vector difference between the target and the fingertip coordinates. The action space a include angular degrees for each joint angle. Angular speed is constrained between $[-0.3, +0.3]rad$ for each time step. UR5 Reacher consists of episodes of interactions, where each episode is 100 time steps long to allow adequate exploration. The fingertip of UR5 Reacher is confined within a 3-dimensional $0.7m \times 0.5m \times 0.4m$ boundary. The robot is also constrained within a joint-angular boundary to avoid self-collision. At each episode, the target position is fixed, and the arm starts from the middle of the right boundary.

For the **Grasping** task, on the basis of reaching, a vacuum suction cup is equipped at the end of UR5 and controlled by the digital output of UR5. It can provide a 6.86N adsorption force under 0.6MPa powered by a vacuum compressor. And the action space is added one dimension to control the trigger of the vacuum gripper suction cup. The suction cup state is also added to the observation space.

In the task of **Reaching**, the robot is required to start at a given location and then move to a goal location of the block. Since the controller of the robot is imperfect, we consider a reach to be successful if the robot reaches within 5cm of the block. We consider **Grasping** to be successful if the robot reaches and succeeded in grasping the target object. For all these tasks, at each episode, the target position for training and evaluation is chosen randomly within the boundary. The UR5 robotic arm is controlled by human volunteers to reach the target, and thus 50 expert demonstration trajectories are generated. All methods are trained on a real UR5 robotic arm for 5000 episodes. As shown in Table III, the approach of using MAIRL outperforms the other methods.

Even though adversarial imitation learning strategy has succeeded in simulation environments, real robotic grasping lags far behind human performance and remains unsolved in the field of robot learning. Sometimes even though the grasping action is executed, the grasping is not successful because of the irregular shape of the object. For example, objects with regular shapes such as balls are easier to grasp, but irregular objects such as instant noodle buckets are very difficult to grasp.

VI. Conclusion

We present an end-to-end differentiable Model-based Adversarial Inverse Reinforcement Learning framework. Our approach addresses the high-variance gradient estimator problem in previous AIRL and learns a policy using the exact gradient. Experiments show the strengths of using a dynamics model with self-attention over the model-free approaches and other baselines in terms of improved performance, and the potential application to the real-world robots.

References

[1] R. S. Sutton, A. G. Barto *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[2] C. Finn, S. Levine, and P. Abbeel, “Guided cost learning: Deep inverse optimal control via policy optimization,” in *International Conference on Machine Learning*, 2016, pp. 49–58.

[3] M. Kuderer, S. Gulati, and W. Burgard, “Learning driving styles for autonomous vehicles from demonstration,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2641–2646.

[4] J. Huang, S. Xie, J. Sun, Q. Ma, C. Liu, D. Lin, and B. Zhou, “Learning a decision module by imitating driver’s control behaviors,” in *Proceedings of the Conference on Robot Learning (CoRL) 2020*.

[5] J. Sun, H. Sun, T. Han, and B. Zhou, “Neuro-symbolic program search for autonomous driving decision module design,” in *Proceedings of the Conference on Robot Learning (CoRL) 2020*.

[6] A. Y. Ng and S. J. Russell, “Algorithms for inverse reinforcement learning,” in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML ’00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, p. 663–670.

[7] A. Coates, P. Abbeel, and A. Y. Ng, “Learning for control from multiple demonstrations,” in *Proceedings of the 25th International Conference on Machine Learning*, 2008, pp. 144–151.

[8] J. Ho and S. Ermon, “Generative adversarial imitation learning,” in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4565–4573.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008.

[10] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[11] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, “Maximum entropy inverse reinforcement learning,” in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI’08. AAAI Press, 2008, p. 1433–1438.

[12] J. Fu, K. Luo, and S. Levine, “Learning robust rewards with adverserial inverse reinforcement learning,” in *International Conference on Learning Representations*, 2018.

[13] M. Deisenroth and C. E. Rasmussen, “Pilco: A model-based and data-efficient approach to policy search,” in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

[14] K. Chua, R. Calandra, R. McAllister, and S. Levine, “Deep reinforcement learning in a handful of trials using probabilistic dynamics models,” in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018, pp. 4754–4765.

[15] D. Ha and J. Schmidhuber, “Recurrent world models facilitate policy evolution,” in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 2450–2462.

[16] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, “Dream to control: Learning behaviors by latent imagination,” in *International Conference on Learning Representations*, 2020.

[17] N. Baram, O. Anschel, I. Caspi, and S. Mannor, “End-to-end differentiable adversarial imitation learning,” in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70, International Convention Centre, Sydney, Australia, 06–11 Aug 2017, pp. 390–399.

[18] R. J. Williams, “Simple statistical gradient-following algorithms for connectionist reinforcement learning,” *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[19] A. Mnih and K. Gregor, “Neural variational inference and learning in belief networks,” in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1791–1799.

[20] M. Janner, J. Fu, M. Zhang, and S. Levine, “When to trust your model: Model-based policy optimization,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 12 519–12 530.

[21] N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa, “Learning continuous control policies by stochastic value gradients,” in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2944–2952.

[22] Y. W. T. Chris J. Maddison, Andriy Mnih, “The concrete distribution: A continuous relaxation of discrete random variables,” in *International Conference on Learning Representations*, 2017.

[23] E. Jang, S. Gu, and B. Poole, “Categorical reparameterization with gumbel-softmax,” 2017.

[24] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1889–1897.

[25] S. Gu, E. Holly, T. Lillicrap, and S. Levine, “Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3389–3396.

[26] X. Chen, Z. Ye, J. Sun, Y. Fan, F. Hu, C. Wang, and C. Lu, “Transferable active grasping and real embodied dataset,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 3611–3618.

[27] A. R. Mahmood, D. Korenkevych, G. Vasan, W. Ma, and J. Bergstra, “Benchmarking reinforcement learning algorithms on real-world robots,” in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., vol. 87. PMLR, 29–31 Oct 2018, pp. 561–591.