BASIS FOR INTENTIONS: EFFICIENT INVERSE REINFORCEMENT LEARNING USING PAST EXPERIENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper studies how prior reinforcement learning (RL) experience can accelerate inverse reinforcement learning (IRL). IRL is the problem of inferring the reward function of an agent by observing its behavior. IRL can provide a generalizable and compact representation for apprenticeship learning, or enable accurately inferring the preferences of a person in order to assist them. However, effective IRL is challenging, because many reward functions can be compatible with an observed behavior. We propose the algorithm BASIS (Behavior Acquisition through Successor-feature Intention inference from Samples), which leverages multi-task RL pre-training and successor features to allow an agent to build a strong basis for intentions that spans the space of possible goals in a given domain. When exposed to just a few expert demonstrations optimizing a novel goal, the agent uses its basis to quickly and effectively infer the reward function. Our experiments reveal that our method is highly effective at inferring and optimizing demonstrated reward functions, accurately inferring reward functions from less than 100 trajectories (3x more efficient that competitive baselines).

1 INTRODUCTION

IRL seeks to identify a reward function under which observed behavior of an expert is optimal. Once an agent has effectively inferred the reward function, it can then use standard (forward) RL to optimize it, and thus acquire not only useful skills by observing demonstrations, but also a reward function as an explanation for the demonstrator's behavior. By inferring the underlying goal being pursued by the demonstrator, the agent is more likely to be able to generalize to a new scenario in which it must optimize that goal, versus an agent that merely imitates the demonstrated actions. IRL has already proven useful in applications including autonomous driving (Huang et al., 2021; Kim & Pineau, 2016), and is a key component in enabling assistive technologies where a helper agent must infer the goals of the human it is assisting (Hadfield-Menell et al., 2016).

However, IRL becomes difficult when the model does not know which aspects of the environment are potentially relevant for obtaining reward. Hence, effective IRL often depends heavily on good features (Abbeel & Ng, 2004; Ziebart et al., 2008). Inferring relevant features from raw, high-dimensional observations is challenging, because there are many possible reward functions consistent with a set of demonstrations. For this reason, previous work has often focused on IRL with hand-crafted features that manually build in prior knowledge (Ziebart et al., 2008; Abbeel & Ng, 2004; Ratliff et al., 2006). When learning rewards from scratch, modern deep IRL algorithms often require a large number of demonstrations and trials (Garg et al., 2021).

In contrast, humans quickly and easily infer the intentions of other people. As shown by Qian et al. (2021), humans can infer rewards more effectively than our best IRL algorithms, as they bring to bear strong prior beliefs as to what might constitute a reasonable goal – e.g., that a person moving towards a wall is more likely to have the intention of turning off the light as opposed to moving to a random point. This skill comes from humans having access to a lot of previous experience successfully accomplishing prior goals or watching others pursue their preferences (Baker et al., 2007). We hypothesize that prior knowledge of the space of probable goals is important to effectively and efficiently infer intentions with IRL. Rather than build in this knowledge as in prior work (Ng & Russell, 2000; Abbeel & Ng, 2004; Ratliff et al., 2006), we propose that the path towards building scalable IRL methods entails being able to *learn* those features from past experience. Thus,

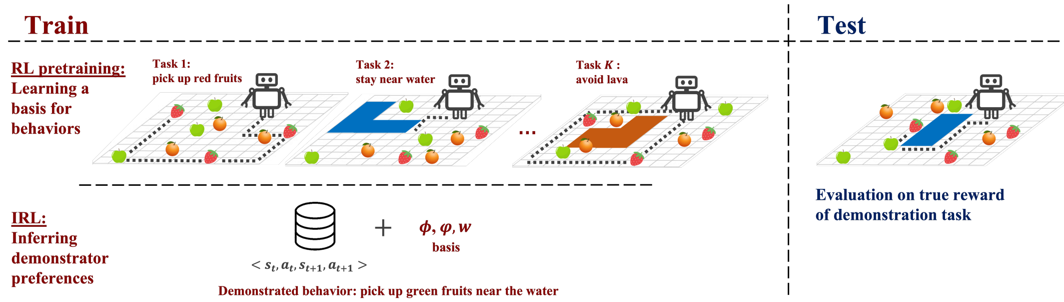


Figure 1: BASIS uses multi-task RL pre-training to learn a *basis for intentions*. It encodes information about both the environment dynamics, and—through modeling the rewards for multiple pre-training tasks—the space of possible goals that can be pursued in the environment. It captures this information in cumulants ϕ, successor representation ψ, and preference vectors $w_{1:K}$. The agent then leverages knowledge from these parameters to rapidly infer the demonstrator's goal shown through demonstrations (s_t, a_t, s_{t+1}), updating the parameters as needed.

we propose an IRL algorithm, BASIS (Behavior Acquisition through Successor-feature Intention inference from Samples), that leverages multi-task RL pre-training and successor features to enable the agent to first learn a *basis for intentions* that spans the space of potential tasks in a domain. Using this basis, the agent can then perform more efficient IRL or inference of goals (see Figure 1).

We use successor features to enable learning a basis for intentions because they provide a representation that naturally decouples the dynamics of the environment from the rewards of the agent, which are represented with a low-dimensional preference vector (Dayan, 1993; Barreto et al., 2018; Filos et al., 2021). Via multi-task pre-training, the agent learns a representation in which the same successor features are shared across multiple tasks, as in Barreto et al. (2018). When the agent is tasked with inferring the rewards of a novel demonstrator via IRL, it initializes its model of the other agent with the learned successor features and a randomly initialized preference vector. Thus, the agent starts with a strong prior over the environment dynamics and the space of reasonable policies. It can then quickly infer the low-dimensional preference vector, while updating the successor features, in order to accurately capture the demonstrated behaviour.

In summary, we contribute a novel IRL algorithm which combines multi-task RL pre-training and successor features. We evaluate BASIS in multiple environment domains, including two autonomous driving scenarios. On these tasks, BASIS is up to 10x more accurate at recovering the demonstrator's reward than state-of-the-art IRL methods (including pre-training with *IRL*), and achieves up to 15x more ground-truth reward than state-of-the-art imitation learning methods.

2 Related Work

Inverse RL: IRL methods learn the reward function of an agent through observing expert demonstrations. Depending on whether the goal is imitation, explanation, or transfer, downstream applications might use the recovered policy, reward function, or both. In environments with high-dimensional state spaces, there are many possible reward functions that are consistent with a set of demonstrations. Thus, early work on IRL relied on hand-engineered features that were known to be relevant to the reward function (Ng & Russell, 2000; Abbeel & Ng, 2004; Ratliff et al., 2006; Syed et al., 2008; Levine et al., 2010). Ziebart et al. (2008) propose maximum entropy IRL, which assumes that the probability of an action being seen in the demonstrations increases exponentially with its reward, an approach we also follow in this paper. A series of deep IRL methods have emerged that learn reward structures (Jin et al., 2017; Burchfiel et al., 2016; Shiarlis et al., 2016) or features (Wulfmeier et al., 2016; Fahad et al., 2018; Jara-Ettinger, 2019; Fu et al., 2018) from input. Inverse Q-learning (InvQ) (Kalweit et al., 2020) is a recent IRL method which uses inverse-action value iteration to recover the underlying reward. It was benchmarked against (Wulfmeier et al., 2016) and found to provide superior performance, so we compare to InvQ in this paper. However, we are not aware of a prior method that leverages past multi-task RL experience as a way to overcome the underspecification issue to improve IRL.

Meta-inverse reinforcement learning methods, including those proposed by Yu et al. (2019); Xu et al. (2019); Gleave & Habryka (2018), have applied meta-learning techniques to IRL, by pre-training on past *IRL* problems, then using meta-learning to adapt to a new IRL problem at test time. In contrast with meta-IRL methods, we pre-train using *RL*, giving the agent a chance to explore the environment and learn to obtain high rewards on multiple tasks. Relying on RL rather than IRL pre-training provides an advantage, since collecting the demonstrations required for IRL can be expensive, but RL only requires access to a simulator. As we will demonstrate, RL pre-training enables our agent to rapidly infer rewards even in complex and high-dimensional environments.

Imitation learning: IL methods attempt to replicate the policy that produced a set of demonstrations. In behaviour cloning (BC)(Ross et al., 2010; Bain & Sammut, 1996), the agent receives training data of states and actions of the expert demonstrator, and uses supervised learning to imitate this data (Ross & Bagnell, 2010). The agent can thus learn new behaviors without having to interact with the environment. Kostrikov et al. (2020); Jarrett et al. (2020); Chan & van der Schaar (2021) have proposed several other IL methods. Some IL methods allow the agent to interact with the environment in addition to receiving demonstrations, and include adversarial methods such as Ho & Ermon (2016); Kostrikov et al. (2018); Baram et al. (2016). Similar to Borsa et al. (2017); Torabi et al. (2018); Sermanet et al. (2018); Liu et al. (2018); Brown et al. (2019), our approach leverages learning from demonstrations where actions are available but rewards and task annotations are unknown. One among these methods is the non-adversarial method IQ-Learn (Garg et al., 2021), which we compare to in this paper. However, current IL methods that focus on skill transfer by minimizing the supervised learning loss do not recover the reward function, which BASIS does.

Successor features: Barreto et al. (2018) derives generalized successor features from successor representations (Dayan, 1993) to create a representation that disentangles task reward from environment dynamics. They have been used for applications including planning (Zhu et al., 2017), zero-shot transfer (Lehnert et al., 2017; Borsa et al., 2018; Barreto et al., 2020; Filos et al., 2021), exploration (Janz et al., 2019; Machado et al., 2020), skill discovery (Machado et al., 2018; Hansen et al., 2020), apprenticeship learning (Lee et al., 2019), and theory of mind (Rabinowitz et al., 2018). PsiPhi-Learning (Filos et al., 2021) illustrates that generalized value functions, such as successor features, are a very effective way to transfer knowledge about agency in multi-agent settings, and includes an experiment which uses successor features for IRL. Unlike PsiPhi, which learns a new set of successor features for each agent it models, we learn a shared set of successor features spanning all tasks that have been seen in our multi-task learning phase, enabling more effective transfer to new tasks that have not been encountered during training. While our multi-task formulation is similar to Barreto et al. (2018), they do not not address IRL or learning from demonstrations.

3 Background and Problem Setting

Markov decision processes: An agent's interaction in the environment can be represented by a Markov decision process (MDP) (Puterman, 1994). Specifically, an MDP is defined as a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, \mathcal{R}, \gamma \rangle$; $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the state transition probability function, $\mathcal{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. An agent executes an action at each timestep t according to its stochastic policy $a_t \sim \pi(a_t|s_t)$, where $s_t \in \mathcal{S}$. An action a_t yields a transition from the current state s_t to the next state $s_{t+1} \in \mathcal{S}$ with probability $P(s_{t+1}|s_t, a_t)$. An agent then obtains a reward according to the reward function $r_t \sim \mathcal{R}(s_t, a_t)$. An agent's goal is to maximize its expected discounted return $\mathbb{E}\left[\sum_t \gamma^t r_t | s_0, a_0\right] = Q(s_0, a_0)$.

Successor features decompose the RL value function into features (which support many tasks) and preference vectors (which are specific to a particular reward function) Barreto et al. (2018); Dayan (1993). Thus, SFs enable quick adaptation to optimizing a new reward function in the same environment (with the same transition dynamics). The one-step expected reward is represented as:

$$r(s_t, a_t, s_{t+1}) = \phi(s_t, a_t, s_{t+1})^\top w, \tag{1}$$

where $\phi(s_t, a_t, s_{t+1}) \in \mathcal{R}^d$ are features (cumulants), and the weights $w \in \mathcal{R}^d$ are a representation of a possible goal or task that has a particular reward function, with each w giving rise to a separate task. We use the terms 'task', 'goal', and 'preferences' interchangeably. The state-action value function for a particular policy π can now be decomposed with the following linear form (Barreto et al., 2018):

$$Q^\pi(s, a) = \psi^\pi(s, a)^\top w \quad \text{and} \quad \psi^\pi(s, a) = \mathbb{E}_{[s_t = s, a_t = a]} \sum_{i=t}^{\infty} \gamma^{i-t} \phi(s_{i+1}, a_{i+1}, s_{i+1}) \tag{2}$$

where $\gamma \in [0, 1)$, and $\psi^\pi(s, a)$ are the successor features of π.

Maximum-entropy IRL: Given a set of demonstrations $D = [(s_0, a_0), (s_1, a_1) \dots]$ provided by an expert, the IRL problem (Ng & Russell, 2000) is to uncover the expert's unknown reward function $\mathcal{R}$. Ziebart et al. (2008) propose the maximum entropy IRL framework, under which highly rewarding actions are considered exponentially more probable in the demonstrations, an assumption which we follow in this work. Specifically, MaxEnt IRL states the expert's preference for any given trajectory between specified start and goal states is proportional to the exponential of the reward along the path: $P(s, a|r) \propto \exp\{\sum_{s,a} \mathcal{R}_{s,a}\}$. Thus, the MaxEnt IRL model (Ziebart et al., 2008) is: $P(a|s_i) = \exp(Q(s_i, a))$. The optimal parameters can be found by maximizing the log likelihood with respect to the parameters of the reward, often utilizing either a tabular method similar to value iteration, or approximate gradient estimators based on adversarially trained discriminators.

4 BASIS IRL

We now present our algorithm, BASIS, (Behavior Acquisition through Successor-feature Intention inference from Samples), illustrated in Figure 1. The agent first learns a basis for intentions using successor features and multi-task RL pre-training. Then, the agent uses the pre-trained successor representation as an initialization for inferring the reward function of an expert from demonstrations with IRL. The successor features learned via pre-training act like a prior over intentions, enabling the agent to *learn* the features for linear MaxEnt IRL. Thus, we get the benefits of simplicity and efficiency due to good features, without having to specify those features manually. These successor features can then be refined to recover the demonstrator's policy using IRL.

4.1 RL PRE-TRAINING: LEARNING A BASIS FOR INTENTIONS

We use multi-task RL pre-training and successor features to learn a representation that enables the agent to solve a large variety of tasks. We assume that each of the tasks share the same state space and transition dynamics, but differ in their reward functions. These assumptions are relevant to the setting in which a human demonstrator may have one of several possible goals in the same environment. We learn a global cumulant function ϕ, successor features ψ, and per-task preference vectors $w_{1:K}$. As the agent operates under the same state space and transition dynamics across the different tasks, we can share ϕ across tasks, which enables learning a common basis for reward functions. Thus, ϕ is a task-agnostic set of state features that are relevant to predicting rewards across any training task. The agent's policy is captured by ψ, which estimates the future accumulation of these state features according to eq. (2). We learn a separate preference vector $w_{1:K}$ for each of the K tasks, which enables representing the different reward functions of the tasks. For simplicity, we refer to a preference vector specific for a task as w. We use a neural network to learn both ψ and ϕ, as shown in Figure 2. Initial features are extracted from raw, high-dimensional observations s_t via a shared trunk of convolution layers. Separate heads output ϕ and ψ (parameterized by θ_ϕ and θ_ϕ, respectively). The preference vectors $w_{1:K}$ do not depend on the state, and are learned separately.

As shown in Eq. 2, combining ψ with a particular preference vector w produces $Q^{\pi,w} = \psi^\pi(s, a)^\top w$, which can be used as a policy π. To ensure our representation is suitable for later IRL training, we fit the Q function using a modified version of the Bellman error with a softmax function, following the formulation of MaxEnt IRL (Ziebart et al., 2008):

$$\mathcal{L}_{\mathrm{Q}}(\theta_\psi) \triangleq \mathbb{E}_{(s_t,a_t,s_{t+1},r_t)\sim\mathcal{B}}[||Q^{\pi,w}(a_t, s_t; \theta_\psi) - r_t - \gamma \operatorname*{softmax}_{a_{t+1}} Q^{\pi,w}(s_{t+1}, a_{t+1}; \tilde{\theta_\psi})||], \tag{3}$$

where $\mathcal{B}$ represents the replay buffer. Note that this formulation of Q-learning is equivalent to Soft Q-Learning Haarnoja et al. (2017), which is a maximum entropy RL method that can improve exploration, and is thus a reasonable choice for a forward RL objective.

To ensure that environment features extracted by ϕ are sufficient to represent the space of possible reward functions, and that each w accurately represents its task-specific reward function, we train both using the following reward loss:

$$\mathcal{L}_{\mathrm{R}}(\theta_\phi, w) \triangleq \mathbb{E}_{(s_t,a_t,r_t)\sim\mathcal{D}}||\phi(s_t, a_t; \theta_\phi)^\top w - r_t||. \tag{4}$$

As shown in Eq. eq. (2), the successor features ψ should represent the accumulation of the cumulants ϕ over time. To enforce this consistency, we train θ_ψ with the following inverse temporal difference

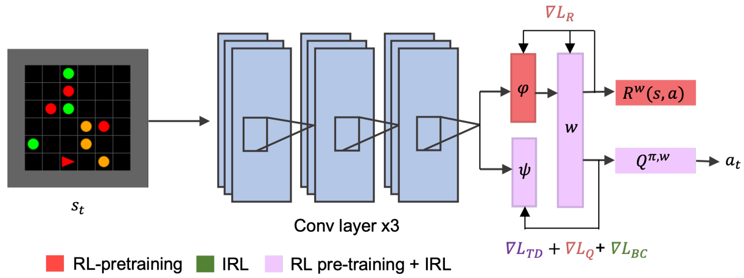


Figure 2: Architecture diagram for learning global cumulants ϕ, successor features ψ, and task-specific preference vectors w. The input s_t is passed through shared convolution layers (blue). This intermediate representation is passed into networks for ψ and ϕ. Taking the dot product of the output from ψ and w, we obtain the action-value function Q. The dot product of ϕ and w gives us the predicted reward at s_t. Networks that are updated in both RL pre-training and during IRL are highlighted in purple (ψ, w) and those learned solely with RL are highlighted in red. The same colour conventions are used to represent the loss functions used for each network parameter, with losses only computed during IRL highlighted in green.

(ITD) loss (Barreto et al., 2018), which is similar to a Bellman consistency loss:

$$\mathcal{L}_{\text{TD}}(\theta_\psi) \triangleq \mathbb{E}_{(s_t,a_t,s_{t+1},a_{t+1})\sim\mathcal{B}}[||\psi(s_t,a_t;\theta_\psi) - \phi(s_t,a_t;\theta_\phi) - \gamma\psi(s_{t+1},a_{t+1};\tilde{\theta}_\psi)||]. \tag{5}$$

We do not train θ_ϕ with this loss (the gradient is not passed through $\phi(s_t, a_t; \theta_\phi)$). This is because we first force ϕ to represent the rewards through Eq. 4, then construct ψ out of the fixed ϕ, which leads to more stable training. Through this process of successor feature learning, our method learns a "basis for intentions" that can be used as an effective prior for IRL in the next phase.

4.2 Inferring intentions with IRL

The goal of IRL is to use observations from a demonstrator to not only recover its policy $\pi_e(a|s)$ but accurately infer its reward function. Our agent is given access to demonstrations without rewards, denoted $D = \tau_1, \tau_2 \dots \tau_N$, where the trajectory $\tau \triangleq (s_0, a_0, \dots)$ is generated by the demonstrator. The demonstrator is optimizing an unknown, ground-truth reward function $r_e(s_t, a_t)$ that was not part of the pre-training tasks. We will now clarify how successor features (Barreto et al., 2018) can be related to the demonstrator's policy and reward function by drawing a parallel between successor features and MaxEnt IRL. This motivates our formulation of IRL with SFs.

MaxEnt IRL assumes that the demonstrator's actions are distributed in proportion to exponentiated Q-values, i.e., $\pi(a|s) \propto \exp(Q(s,a))$ (see Section 3). We then learn the parameters of the expert's Q-function, θ_e, by maximizing the log-likelihood of the expert's actions: $\theta_e^* = \arg\max_{\theta_e} \sum_{a=1}^{a_T} \log P(a|\pi_e)$. Since we can represent the expert's Q-value with successor features ψ_e and preference vector w_e (Eq. 2), we can express the expert's policy as $\pi_e(a|s) \propto \exp(\psi_e^\top w_e)$. This leads to the following loss to fit our task-specific preferences w_e and successor features ψ_e:

$$\mathcal{L}_{\text{BC}}(\theta_{\psi_e}, \omega_e) \triangleq -\mathbb{E}_{\tau\sim D} \log \frac{\exp(\psi_e(s,a)^\top w_e)}{\sum_a (\exp \psi_e(s,a)^\top w_e).} \tag{6}$$

However, this BC-like loss is insufficient to produce an effective IRL method, since we have no way to infer the reward function of the expert. To predict the expert's rewards, we need to use ϕ; i.e.:

$$\phi_e(s,a)^\top w_e = r_e(s,a) \tag{7}$$

To ensure that ψ_e and ϕ_e remain consistent during this optimization, we also require an ITD loss:

$$\mathcal{L}_{\text{TD}}(\theta_{\psi_e}) \triangleq \mathbb{E}_{\mathcal{D}}||\psi_e(s_t,a_t;\theta_{\psi_e}) - \phi_e(s_t,a_t;\theta_{\phi_e}) - \gamma\psi_e(s_{t+1},a_{t+1};\tilde{\theta_{\psi_e}})||. \tag{8}$$

Now we can draw a direct connection to MaxEnt IRL, which proposes inferring the demonstrator's reward function using a linear transformation applied to a set of state features: $\mathcal{R}(f(s);\theta) = \theta^T f(s)$ where the mapping $f : \mathcal{S} \to [0,1]^k$ is a state feature vector, θ are the model parameters, and $\mathcal{R}(f(s), \theta)$ is the reward function. Our approach instead uses successor features to learn a set of continuous state features ϕ_e to replace f, and w_e is analogous to learning θ. Thus, according to the MaxEnt IRL model, if $\psi_e^{\pi}(s,a)$ remains Bellman consistent with ϕ_e (by minimizing the ITD loss) and w_e and $\psi_e^{\pi_e}(s,a)$ are optimized so as to maximize the probability of the observed demonstration actions (as in Eq. 6), we will have recovered the demonstrator's Q-function as $\psi_e(s,a)^T w$, and the demonstrator's reward function as $\phi_e(s,a)^T w_e$.

Benefitting from RL pre-training: To initialize ψ_e and ϕ_e, the agent uses the parameters θ_ϕ and θ_ψ that it learned during RL pre-training; i.e. $\theta_{\psi_e} \leftarrow \psi$ and $\theta_{\phi_e} \leftarrow \theta_\phi$. It also initializes a new preference vector w_e as the average of all w vectors across tasks learned in during RL pre-training, to begin with an agnostic representation of the demonstrator's goals. The ϕ learned during RL training represents a shared feature space that can be used to explain all the pre-training tasks. When the agent initializes ϕ_e with ϕ, it is given a strong prior that can help explain the expert's behavior. Hence, even though at the beginning of the IRL stage the agent has never been directly trained on behavior for the new task, it can potentially extrapolate from the learned basis in order to quickly ascertain the demonstrator's preferences. To the extent that the pre-trained successor features ψ provide a good basis for the expert's policy, then this learning process might primarily modify w_e, and only make minor changes to ψ_e to make it consistent with the new policy. In this case, the method can recover the reward and policy for the new task very quickly. However, the astute reader will notice that the basis for intentions learned during RL pre-training and encoded in θ_ϕ and θ_ψ does not necessarily constitute a policy that is optimal for the new task that is being demonstrated by the expert. Hence, the ITD loss (Eq. 5) is necessary to learn the correct ψ_e. Even if the demonstrator policy differs from the ψ learned during RL pre-training, ITD allows us to accurately infer the demonstrated policy.

Algorithm 1 RL pre-training: learning a basis

```
Initialize θ_φ, θ_ψ, and w^k_{k=1}^K
B = []                          ▷ Replay buffer initialization
for each iteration do
    Get initial observation s_t
    Sample task k from K tasks
    for each environment step do
        Get action a ∼ π
        Take action a and observe x' and r
        B ← B ∪ {x, a, r, x', k}
    for each gradient step do
        Update θ_ψ, w w/ Bellman Eq. 3
        Update θ_φ, w w/ reward loss Eq. 4
        Update θ_ψ, w w/ ITD loss Eq. 5
```

Algorithm 2 IRL: Inferring Intentions

```
Input: expert demonstrations D
Initialize θ_{φ_e} ← φ, θ_{ψ_e} ← ψ, w_e ← Σ_1^K w_k/K from multi-task RL pre-training
for each demo <s_t, a_t, s_{t+1}, a_{t+1}> do:
    Update ψ_e, w_e with BC loss in eq. (6)
    Update ψ_e, w_e with ITD loss in eq. (8)
```

Algorithm summary: We now summarize the procedure for both training stages. Algorithm 1 uses multi-task RL pre-training to learn a basis for intentions, updating cumulants, SFs, and preference vectors via the Bellman error, reward loss, and ITD loss. In Algorithm 2 (inferring intentions with IRL), we begin the learning process with parameters initialized from the RL pre-training. We iterate through a batch of a fixed set of demonstrations from D, computing the BC loss and ITD loss to maintain consistency between ψ and ϕ as aforementioned. At test time, we use the inferred cumulants ϕ_e, preference vector w_e, and successor representation ψ_e, to produce a policy that can be executed in the test environment, and measure how well it matches the demonstrator's reward (as in Figure 1).

5 Experiments

Below we describe the research questions that our empirical experiments are designed to address.

Question 1: Will BASIS acquire the behaviors of a demonstrator expert more quickly and effectively than conventional IRL and imitation learning methods?
A central goal of IRL is to be able to reproduce the demonstrated behavior of the expert in a generalizable way. We hypothesize that the RL pre-training and successor representation of BASIS will allow it do this more accurately and with fewer demonstrations than existing techniques. To measure how well a method can acquire demonstrated behaviors, we evaluate performance using the *expected value difference*, which is the difference between the return achieved by the expert policy

and the policy inferred with IRL, both measured under the ground truth reward function (thus, a lower value difference is better). Intuitively, this metric captures how much worse the policy that IRL recovers is vs. the expert demonstrator's policy, and is the metric of choice in for evaluating IRL methods in prior work (Levine et al., 2011; Wulfmeier et al., 2016; Xu et al., 2019). We evaluate the performance of each algorithm with different numbers of demonstrations to study whether BASIS can perform IRL more efficiently (i.e., with fewer demonstrations).

Question 2: Can BASIS more accurately predict the true reward with fewer demonstrations? It is often easier to optimize for the correct behavior of a demonstrator agent than accurately predict its reward function. Even with inaccurate reward values, the agent could still demonstrate the correct behavior as long as it estimated the relative value of different actions correctly. Hence, we perform further analysis to observe how well the agent is able to predict the expert's true reward function. We compute the mean squared error between the agent's prediction of the reward, and the true environment reward that is observed: $MSE = (\phi_e(s_t, a_t)^T w_e - r_t)^2$. This allows us to understand how accurately the agent is able to infer intentions by leveraging its basis for intentions.

Question 3: How closely does BASIS match the demonstrated policy and adhere to the demonstrator's preferences? We visualize the behaviors of agents to understand how well they are able to adhere to the demonstrator's preferences. For example, we show the distribution of behaviors for each method vs. the distribution of the expert's policy.

Question 4: How does multi-task pre-training & successor features benefit IRL? We address this question through ablations that show how much multi-task RL pre-training vs successor features contribute to learning an IRL task. We compare BASIS to two ablations. The first is uses no multi-task pre-training, but does perform IRL via BC and successor features. Denoted as "No pre-training (BC + successor features)", it is used to assess the importance of RL pre-training. The second, "No successor features (pre-train with DQN)", is an algorithm which performs multi-task pre-training via DQN and IRL via BC, and assesses the importance of successor features.

5.1 Domains

We evaluate BASIS on a gridworld environment Fruit-Picking (which allows us to carefully analyze and visualize performance) and high-dimensional autonomous driving environments Highway and Roundabout (which necessitate the use of deep IRL). We modified the domains to be able to create multiple tasks with differing reward functions, enabling us to test generalization to novel demonstrator tasks outside of the set of pre-training tasks. See Appendix 8.1 for further details.

Fruit-Picking is a custom environment based on Chevalier-Boisvert et al. (2018), with different colored fruits the agent must gather. In each task, the number and type of fruits varies, along with the reward received for gathering a specific type of fruit. During RL pre-training, the agent learns to pick one type of fruit per task. For the IRL phase, the demonstrated task is different from the training tasks in that the demonstrator shows a preference for multiple types of fruit i.e. 80% preference for red, 20% preference for orange. This behavior was not seen during pre-training.

Highway-Env & Roundabout-Env are autonomous driving and tactical decision-making environments Leurent (2018). We have modified the agent's reward objective to maintain a target speed, target distance from the front vehicle, and target lane, while avoiding collisions with neighbouring vehicles. As there are many continuous parameters that determine the reward function for the agent, it is not possible to sample all combinations of behaviors within the training tasks. Thus, it is straightforward to create a novel test task for the demonstrator.

5.2 Baselines and Comparisons

We compare BASIS to three baselines. For all baselines, we use the same hyperparameters as BASIS when applicable, and maintain default values from source code otherwise. **IQ-learn** (Garg et al., 2021) is a state-of-the-art, dynamics-aware imitation learning (IL) method that is able to perform with very sparse training data, scaling to complex image-based environments. As it is able to implicitly learn rewards, this method can also be used in IRL. We use the authors' original implementation from: `https://github.com/Div99/IQ-Learn`. **InvQ (Inverse Q-learning)** (Kalweit et al., 2020) is a state-of-the-art inverse RL method, which uses inverse-action value iteration to recover the underlying reward of an external agent, providing a strong comparison representative of recent inverse

RL methods. **Multi-task IRL pre-training** is included to enable a fair comparison to methods which leverage multi-task pre-training. To our knowledge there is no prior method that uses RL pre-training to acquire a starting point for IRL. Closest in spirit is prior work on meta-IRL (Yu et al., 2019; Xu et al., 2019; Gleave & Habryka, 2018), which pre-train on other *IRL* tasks, rather than pretraining on a set of standard RL problems, as in our method. Since these works generally assume a tabular or small discrete-state MDP, they are not directly applicable to our setting. We thus create a multi-task IRL pre-training baseline by applying the same network architecture and BC and ITD losses as our own method (essentially performing our IRL phase (Algorithm 2) during pre-training as well). We note that in general, IRL pre-training has the significant disadvantage that it requires many more expert demonstrations during the pre-training phase, whereas our method does not.

6 Results

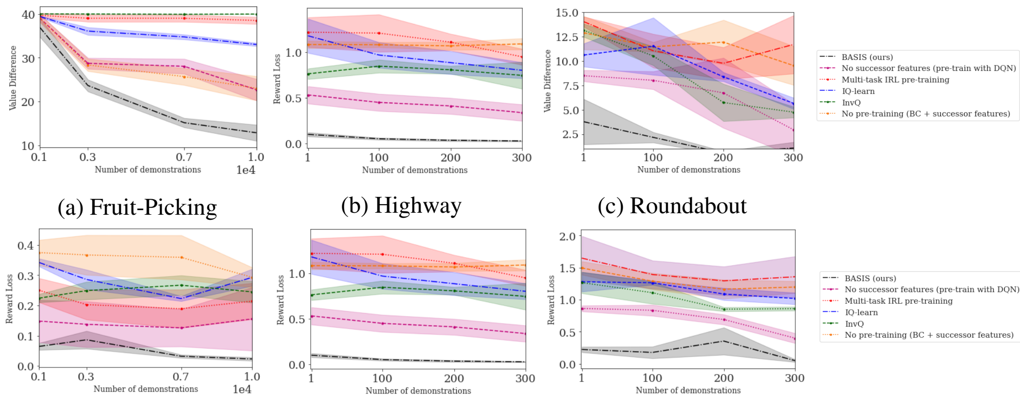
We demonstrate BASIS's ability to leverage prior experience to help infer preferences quickly and efficiently on a diverse suite of domains. The code is at https://github.com/3047iclr2023/basis-irl and videos are at https://sites.google.com/view/basis-irl.

Question 1: Acquiring demonstrated behavior. Figure 3 (top row) shows the value difference (VD) when evaluating an agent after learning from a series of demonstrations for all three environments. In Fruit-Picking, we observe that BASIS is better able to optimize the demonstrated policy than IQ-learn, InvQ, and multi-task IRL pre-training, achieving the lowest VD of 15 at 10000 demonstrations, and surpassing the final performance of the baselines with less than 1/3 of the demonstrations. We see similar trends in Figures 3b and 3c, showing the VD when inferring previously unseen driving preferences in the Highway and Roundabout domains. The VD for BASIS is significantly lower than all baselines across all numbers of demonstrations, and it is once again able to surpass the best baseline with only 1/3 of the data requirements. This experiment allows us to determine whether BASIS fulfills the first requirement of being an IRL algorithm as well as an IL algorithm: being able to reproduce the behavior demonstrated by the expert. Further, because the performance of BASIS after only a few demonstrations surpasses the performance of both baselines after three times the number of demonstrations, this provides evidence that building a strong prior over the space of reasonable goals helps BASIS infer the expert's behavior rapidly and efficiently.

Question 2: Predicting rewards. Figure 3 (bottom row) shows the reward loss (MSE in predicting the ground truth reward) obtained by each of the methods. Across all three environments, BASIS achieves the lowest error vs. any of the baseline techniques, often reaching the best performance in a fraction of the examples. In Fruit-Picking (Fig. 3d), BASIS achieves the best performance after only 100 demonstrations, surpassing the performance of other techniques after 1000 demonstrations. Similarly, as shown in 3e, BASIS converges to a lower reward MSE loss than any of the other techniques after only one demonstration. This rapid inference of the expert's reward suggests that the basis acquired during pre-training was sufficient to explain the expert's behavior, and the algorithm was able to adapt rapidly to the expert's task by simply updating w_e (as explained in Section 4.2). This is consistent with prior work using successor features for multi-agent learning (Filos et al., 2021), which also showed 1-shot adaptation to a novel test task.

Question 3: Matching behavior statistics. To assess how well the different methods match the demonstrated behavior, we visually compare the distribution of behaviors for each technique to the expert's distribution. For the fruit-picking task in Fig. 4, InvQ gathers all fruits in roughly equal proportion, showing it has not learned to discern which fruits the expert prefers. This could be due to a failure to scale to high-dimensional environments. Although IQ-learn and Multi-task IRL pre-training are better able to capture the correct distribution, BASIS shows a distribution closer to the ground truth than either of the baselines. Figure 5 conducts a a similar analysis in Highway, visualizing how well the learned agent adheres to staying in the preferred left lane. BASIS shows a preference for the left lane 80% of the time, compared to IQ-learn which matches the expert's preference 60% of the time. Both InvQ and Multi-task IRL pre-training show an almost uniform lane preference.

Question 4: How does multi-task pre-training & successor features benefit IRL? We conduct ablation experiments to assess which components of BASIS contribute to its effectiveness. We observe value difference and reward loss in Figure 3 to be larger without successor features *and* without multi-task pre-training, demonstrating the benefit of both components of our approach. We note that for both Fruit-Picking and Highway, the *No pre-training* baseline does best, actually outperforming or matching the performance of all three baseline techniques (IQ-Learn, InvQ, and



(a) Fruit-Picking (b) Highway (c) Roundabout

(d) Fruit-Picking (e) Highway (f) Roundabout

Figure 3: **Top row: value difference**, the difference in the return obtained by each algorithm and the expert policy. Across the Fruit-Picking (a), Highway (b), and Roundabout (c) domains, BASIS shows the lowest value difference indicating its behavior is closest to that of the optimal policy. It is able to surpass the performance of all baselines with less than one third of the data. **Bottom row: reward loss**, the error in predicting the expert's true reward. BASIS is able to converge to a lower reward loss compared to baselines in all three domains, i.e. it is most accurate in predicting the expert's reward. The error bars show the standard deviation of 10 seeds.

multi-task IRL pre-training), suggesting successor features are highly effective for IRL. However, *No pre-training* performs poorly in the more complex Roundabout environment. In Roundabout, *No successor features* actually gives better performance than two of the baseline techniques (InvQ and multi-task IRL pre-training), demonstrating a strong benefit of learning a prior with RL pre-training.

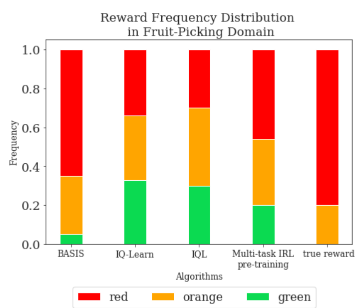


Figure 4: On Fruit-Picking, the agent must infer the demonstrator's preference to consume 80% red and 20% orange fruits. BASIS is able to achieve this distribution more accurately than baselines. Plots show the mean of 10 seeds.

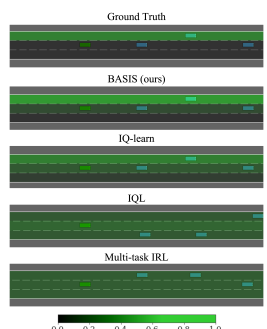


Figure 5: State occupancy map for the degree of left lane preference in the Highway Domain. BASIS is best able to adhere to the demonstrator agent's preferences (ground truth).

7 Conclusion

A major challenge in inverse RL is that the problem is underconstrained. With many different reward functions consistent with observed expert behaviour, it is difficult to infer a reward function for a new task. BASIS presents an approach to this problem by building a strong basis of intentions by combining multi-task RL pre-trainin with successor features. BASIS leverages past experience to infer the intentions of a demonstrator agent on an unseen task. We evaluate our method on domains with high dimensional state spaces, and compare to state-of-the-art inverse RL and imitation learning baselines, as well as pre-training with multi-task IRL. Our results show that BASIS is able to achieve better performance than prior work in less than one third of the demonstrations. The limitation of this approach is that it requires the design of a set of tasks for RL pre-training that are relevant to the expert demonstrations. Nevertheless, this work shows the potential of building a generalizable basis of intentions for efficient IRL.

REFERENCES

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, pp. 1, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015430. URL https://doi.org/10.1145/1015330.1015430.

Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pp. 103–129. Oxford University Press, 1996.

Chris Baker, Joshua Tenenbaum, and Rebecca Saxe. Goal inference as inverse planning. *Proceedings of the 29th Annual Conference of the Cognitive Science Society*, 01 2007.

Nir Baram, Oron Anschel, and Shie Mannor. Model-based adversarial imitation learning, 2016.

André Barreto, Shaobo Hou, Diana Borsa, David Silver, and Doina Precup. Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences*, 117: 30079 – 30087, 2020.

André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, Hado van Hasselt, and David Silver. Successor features for transfer in reinforcement learning, 2018.

Diana Borsa, Bilal Piot, Rémi Munos, and Olivier Pietquin. Observational learning by reinforcement learning, 2017.

Diana Borsa, André Barreto, John Quan, Daniel J. Mankowitz, Rémi Munos, Hado van Hasselt, David Silver, and Tom Schaul. Universal successor features approximators. *CoRR*, abs/1812.07626, 2018. URL http://arxiv.org/abs/1812.07626.

Daniel S. Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations, 2019.

Benjamin Burchfiel, Carlo Tomasi, and Ronald Parr. Distance minimization for reward learning from scored trajectories. In *AAAI*, AAAI'16, pp. 3330–3336. AAAI Press, 2016.

Alex J. Chan and Mihaela van der Schaar. Scalable bayesian inverse reinforcement learning, 2021.

Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid, 2018.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993. doi: 10.1162/neco.1993.5.4.613.

Muhammad Fahad, Zhuo Chen, and Yi Guo. Learning how pedestrians navigate: A deep inverse reinforcement learning approach. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 819–826, 2018. doi: 10.1109/IROS.2018.8593438.

Angelos Filos, Clare Lyle, Yarin Gal, Sergey Levine, Natasha Jaques, and Gregory Farquhar. Psiphi-learning: Reinforcement learning with demonstrations using successor features and inverse temporal difference learning, 2021.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rkHywl-A-.

Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. IQ-learn: Inverse soft-q learning for imitation. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=Aeo-xqtb5p.

Adam Gleave and Oliver Habryka. Multi-task maximum entropy inverse reinforcement learning, 2018.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pp. 1352–1361. PMLR, 2017.

Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell. Cooperative inverse reinforcement learning, 2016.

Steven Hansen, Will Dabney, Andre Barreto, Tom Van de Wiele, David Warde-Farley, and Volodymyr Mnih. Fast task inference with variational intrinsic successor features, 2020.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/cc7e2b878868cbae992d1fb743995d8f-Paper.pdf.

Zhiyu Huang, Jingda Wu, and Chen Lv. Driving behavior modeling using naturalistic human driving data with inverse reinforcement learning, 2021.

David Janz, Jiri Hron, Przemysł aw Mazur, Katja Hofmann, José Miguel Hernández-Lobato, and Sebastian Tschiatschek. Successor uncertainties: Exploration and uncertainty in temporal difference learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/1b113258af3968aaf3969ca67e744ff8-Paper.pdf.

Julian Jara-Ettinger. Theory of mind as inverse reinforcement learning. *Current Opinion in Behavioral Sciences*, 29:105–110, 2019. ISSN 2352-1546. doi: https://doi.org/10.1016/j.cobeha.2019.04.010. URL https://www.sciencedirect.com/science/article/pii/S2352154618302055. Artificial Intelligence.

Daniel Jarrett, Ioana Bica, and Mihaela van der Schaar. Strictly batch imitation learning by energy-based distribution matching. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7354–7365. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/524f141e189d2a00968c3d48cadd4159-Paper.pdf.

Ming Jin, Andreas Damianou, Pieter Abbeel, and Costas Spanos. Inverse reinforcement learning via deep gaussian process, 2017.

Gabriel Kalweit, Maria Huegle, Moritz Werling, and Joschka Boedecker. Deep inverse q-learning with constraints, 2020.

Beomjoon Kim and Joelle Pineau. Socially adaptive path planning in human environments using inverse reinforcement learning. *International Journal of Social Robotics*, 8:51–66, 01 2016. doi: 10.1007/s12369-015-0310-2.

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, 2018.

Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Imitation learning via off-policy distribution matching. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Hyg-JC4FDr.

Donghun Lee, Srivatsan Srinivasan, and Finale Doshi-Velez. Truly batch apprenticeship learning with deep successor features, 2019. URL https://arxiv.org/abs/1903.10077.

Lucas Lehnert, Stefanie Tellex, and Michael L. Littman. Advantages and limitations of using successor features for transfer in reinforcement learning, 2017.

Edouard Leurent. An environment for autonomous driving decision-making. https://github.com/eleurent/highway-env, 2018.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Feature construction for inverse reinforcement learning. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper/2010/file/a8f15eda80c50adb0e71943adc8015cf-Paper.pdf.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper/2011/file/c51ce410c124a10e0db5e4b97fc2af39-Paper.pdf.

YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation, 2018.

Marlos C. Machado, Clemens Rosenbaum, Xiaoxiao Guo, Miao Liu, Gerald Tesauro, and Murray Campbell. Eigenoption discovery through the deep successor representation, 2018.

Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. Count-based exploration with the successor representation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 5125–5133, Apr. 2020. doi: 10.1609/aaai.v34i04.5955. URL https://ojs.aaai.org/index.php/AAAI/article/view/5955.

Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *in Proc. 17th International Conf. on Machine Learning*, pp. 663–670. Morgan Kaufmann, 2000.

ML Puterman. Markov decision processes. 1994. *Jhon Wiley & Sons, New Jersey*, 1994.

Yingdong Qian, Marta Kryven, Tao Gao, Hanbyul Joo, and Josh Tenenbaum. Modeling human intention inference in continuous 3d domains by inverse planning and body kinematics, 2021.

Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, S. M. Ali Eslami, and Matthew Botvinick. Machine theory of mind. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4218–4227. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/rabinowitz18a.html.

Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pp. 729–736, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143936. URL https://doi.org/10.1145/1143844.1143936.

Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/ross10a.html.

Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. No-regret reductions for imitation learning and structured prediction. *CoRR*, abs/1011.0686, 2010. URL http://arxiv.org/abs/1011.0686.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video, 2018.

Kyriacos Shiarlis, Joao Messias, and Shimon Whiteson. Inverse reinforcement learning from failure. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, AAMAS '16, pp. 1060–1068, Richland, SC, 2016. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450342391.

Umar Syed, Michael Bowling, and Robert E Schapire. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*, pp. 1032–1039, 2008.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation, 2018.

Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning, 2016.

Kelvin Xu, Ellis Ratner, Anca Dragan, Sergey Levine, and Chelsea Finn. Learning a prior over intent via meta-inverse reinforcement learning, 2019.

Lantao Yu, Tianhe Yu, Chelsea Finn, and Stefano Ermon. Meta-inverse reinforcement learning with probabilistic context variables, 2019.

Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. Visual semantic planning using deep successor representations, 2017.

Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pp. 1433–1438. AAAI Press, 2008. ISBN 9781577353683.