

Figure 6: We evaluate on the Fruit-Picking, Highway, and Roundabout domains (shown left to right).

8 APPENDIX

8.1 ENVIRONMENT DETAILS

Fruit-Picking is built from a library of open-source grid-world domains in sparse reward settings with image observations (Chevalier-Boisvert et al., 2018). We designed this custom environment with different colored fruits spread across the grid that the agent must gather. The number of fruits and types of fruits are customizable, along with the reward received for gathering a specific type of fruit. To learn a basis for intentions using RL pre-training, the agent learns to pick three different colored fruit (red, orange, green) depending on the task ID that is provided in the observation as a one-hot encoded vector. The agent picks as many fruits as it can until the horizon of the episode. Each fruit is replaced in a random location after being picked by the agent such that there are always 3 fruits of each color present in the grid. In Phase II, the demonstrated task is different to the training tasks; namely, the agent shows a varying degree of preference to each of the fruits in the environment i.e. 80% preference for red fruits, 20% preference for orange fruits, and 0% preference for green fruits. This behavior was not seen during pre-training. The final reward of the expert in this task is 40.

Highway-Env & Roundabout-Env (Leurent (2018)) features a collection of autonomous driving and tactical decision-making environments. We chose to model driving behavior as it allows us to determine the ability of IRL algorithms to learn the hidden intentions of a driver. In the highway env, the ego-vehicle is driving on a three-lane highway populated with other vehicles positioned at random. All vehicles can switch lanes and change their speed. We have modified the agent’s reward objective to maintain a target speed while avoiding collisions with neighbouring vehicles and keeping to a preferred lane. Maintaining a target distance away from the front vehicle is also rewarded. As there are many continuous-spaced parameters that determine the reward function for the agent, it is not possible to sample all combinations of behaviors within the training tasks. Thus, to create a novel test task for the demonstrator, it is straightforward to choose a different combination of these behaviors. Specifically, we test on a task of the driving agent maintaining a desired distance 10 from the vehicle in front of it while maintaining a speed of 28 m/s. We perform similar modifications to the Roundabout where the agent must merge onto a roundabout while maintaining a specific speed and target distance away from other vehicles. We build off the implementation of the Highway domain here: <https://github.com/eleurent/highway-env>.

8.2 NETWORK ARCHITECTURE DETAILS

For the MiniGrid Domain, networks for ϕ , ψ and w policy share three convolution layers, with a ReLU after each layer and a max-pooling operation after the first ReLU activation. ψ and ϕ are represented by two separate linear layers with a Tanh activation function between them. Finally, w is represented as a single parameter. The network architecture is as follows:

- Shared feature extraction layer:
 - Conv2d (3, 16) 2x2 filters, stride 1, padding 0
 - ReLU
 - MaxPool2d (2, 2)
 - Conv2d (16, 32) 2x2 filters, stride 1, padding 0
 - ReLU

- Conv2d (32, 64) 2x2 filters, stride 1, padding 0
- ReLU
- ϕ network layer:
 - FC (256 + num_tasks, 64)
 - Tanh
 - FC (64, num_cumulants + num_actions)
- ψ network layer:
 - FC (256 + num_tasks, 64)
 - Tanh
 - FC (64, num_cumulants + num_actions)
- w network layer:
 - Parameter (num_tasks, num_cumulants)

In this domain, num_cumulants=64, num_actions=4, and num_tasks=3. Note that the network architecture stays the same across RL pre-training and IRL, however during IRL, the num_tasks hyper-parameter is not provided, and a dummy value (a vector of 0's equal to number of tasks) is used.

We now present the architecture for the Highway Domain. Networks for ϕ , ψ and w policy share a linear layer with a ReLU activation after. ψ and ϕ are represented by two separate linear layers with a ReLU activation function between them. Finally, w is represented as a single parameter. The network architecture is as follows:

- Shared feature extraction layer:
 - FC (5, 256)
 - ReLU
- ϕ network layer:
 - FC (1280 + num_tasks, 256)
 - ReLU
 - FC (256, num_cumulants + num_actions)
- ψ network layer:
 - FC (1280 + num_tasks, 256)
 - ReLU
 - FC (256, num_cumulants + num_actions)
- w network layer:
 - Parameter (num_tasks, num_cumulants)

In this domain, num_cumulants=64, num_actions=5, and num_tasks=10. Similar to Fruit-picking, the network architecture stays the same between RL pre-training and IRL.

8.3 IMPLEMENTATION DETAILS

On global feature ϕ . We would like to clarify assumptions made in this paper that will address the reviewer's question on why ϕ transfers to the task during IRL (inferring intentions). As ϕ is indeed critical for the method's ability to recover effective rewards on downstream tasks, it is learned from the pre-training tasks which are within the same distribution (noted in Section 4). Our training procedure ensures that the ϕ features are sufficient to represent the pre-training tasks (optimized directly in the objective for ψ and ϕ). If these features then generalize to other in-distribution tasks in pre-training, they should also be sufficient for downstream tasks during IRL from the same distribution. Hence, we reuse ϕ during IRL and do not optimize it further. We have run an additional empirical analysis as suggested in Figure 7, where we initialize ϕ from RL pre-training (learning a

basis) and allow it to be optimized via ITD loss during IRL as well. There is marginal change in the value difference when optimizing and not optimizing ϕ (ours) during IRL in the Highway domain, confirming our intuition.

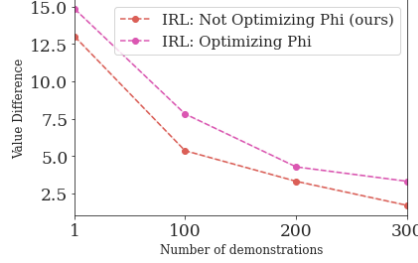


Figure 7: Optimizing ϕ (IRL) causes no significant change.

On global feature ϕ . We would like to clarify assumptions made in this paper that will address the reviewer’s question on why ϕ transfers to the task during IRL (inferring intentions). As ϕ is indeed critical for the method’s ability to recover effective rewards on downstream tasks, it is learned from the pre-training tasks which are within the same distribution (noted in Section 4). Our training procedure ensures that the ϕ features are sufficient to represent the pre-training tasks (optimized directly in the objective for ψ and ϕ). If these features then generalize to other in-distribution tasks in pre-training, they should also be sufficient for downstream tasks during IRL from the same distribution. Hence, we reuse ϕ and do not optimize it further. We have run an additional empirical analysis as suggested in Figure 7, where we initialize ϕ from RL pre-training (learning a basis) and allow it to be optimized via ITD loss during IRL as well. There is marginal change in the value difference when optimizing and not optimizing ϕ (ours) during IRL in the Highway domain, confirming our intuition.

8.4 MAXENTROPY DERIVATION

Expanding on our explanation in Section 4.2, if $r = \phi(s, a)^T w$ is the representation for the reward, and $Q(a, s) = \psi(a, s)w$, then the MaxEnt IRL problem can be written as:

$$\begin{aligned}
 \max_{w, \phi} \mathbb{E}[\log \pi(a|s)] &= \max_{w, \phi} \mathbb{E}[\text{softmax}(Q(s, a))] \\
 &= \max_{w, \phi} \mathbb{E}[\text{softmax}(\psi(s, a)^T w)] \\
 s.t. \quad \psi(s, a) &= \phi(s, a) + \gamma * \mathbb{E}_{a' \sim \text{softmax}(\psi(s', a')^T w)} [\psi(s', a')]
 \end{aligned} \tag{9}$$

This leads to our method, which relaxes the constraint into a soft constraint.