Meta Inverse Reinforcement Learning via Maximum Reward Sharing

Kun Li * **Joel W. Burdick**

Abstract

This work handles the inverse reinforcement learning (IRL) problem where only a small number of demonstrations are available from a demonstrator for each high-dimensional task, insufficient to estimate an accurate reward function. Observing that each demonstrator has an inherent reward for each state and the task-specific behaviors mainly depend on a small number of key states, we propose a meta IRL algorithm that first models the reward function for each task as a distribution conditioned on a baseline reward function shared by all tasks and dependent only on the demonstrator, and then finds the most likely reward function in the distribution that explains the task-specific behaviors. We test the method in a simulated environment on path planning tasks with limited demonstrations, and show that the accuracy of the learned reward function is significantly improved.

1 Introduction

Inverse reinforcement learning (IRL) [1] algorithms estimate a reward function that explains the motions demonstrated by an operator or other agents on a task described by a Markov Decision Process (MDP) [2]. The accuracy of the recovered function depends heavily on the ratio of visited states in the demonstrations to the whole state space, because the demonstrator's motion policy can be estimated more accurately if every state is repeatedly visited. However, the ratio is low for many useful applications, since they usually have huge or high-dimensional state spaces, while the demonstrations are relatively rare for each task. In practice, usually multiple tasks can be observed from the same demonstrator, and the problem of rare demonstrations can be handled by combining data from all tasks, hence the meta-learning problem.

In many IRL applications, we observe that a demonstrator usually has an inherent reward for each state, materialized as the innate state preferences of a human, the hardware-dependent cost function of a robot, the default structure of an environment, etc. For a given task, the demonstrators are usually reluctant to drastically change the inherent reward function to complete the task; instead, they alter the innate reward function minimally to generate a task-specific reward function and plan the motion. For example, in path planning, the C-space of a mobile robot at home rarely changes, and the robot's motion depends on the goal state; in human motion analysis, the costs of different poses are mostly invariant, while the actual motion depends on the desired directions.

Based on this observation, we propose a meta inverse reinforcement learning algorithm by maximizing the shared rewards among all tasks. We model the reward function for each task as a probabilistic distribution conditioned on an inherent baseline function, and estimate the most likely reward function in the distribution that explains the observed task-specific demonstrations.

*Department of Mechanical and Civil Engineering, California Institute of Technology, kunli@caltech.edu

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

2 Meta Inverse Reinforcement Learning

2.1 Meta Inverse Reinforcement Learning

We assume that an agent needs to handle multiple tasks in an environment, denoted by $\{\mathbb{T}_i | i = 1, N_{\mathbb{T}}\}$, where $\mathbb{T}_i$ denotes the i_{th} task and $N_{\mathbb{T}}$ denotes the number of tasks.

We describe a task $\mathbb{T}_i$ as a Markov Decision Process $S_i, A_i, R_i, P^a_{i,ss'}, \gamma_i$. For a task $\mathbb{T}_i$, the agent performs a set of demonstrations $\zeta_i = \{\zeta_{i,j} | j = 1, \cdots, N_{\zeta_i}\}$, represented by N_{ζ_i} sequences of state-action pairs:

$$\zeta_{i,j} = \{(s^t_{i,j}, a^t_{i,j}) | t = 0, \cdots, N_{\zeta_{i,j}}\},$$

where $N_{\zeta_{i,j}}$ denotes the length of the j_{th} sequence $\zeta_{i,j}$. Given the observed sequences $\zeta = \{\zeta_i | i = 1, \cdots, N_{\mathbb{T}}\}$ for the $N_{\mathbb{T}}$ tasks, inverse reinforcement learning algorithms try to recover a reward function $r_i(s)$ for each task.

Our key observation in multi-task IRL is that the demonstrator has an inherent reward function $r_b(s)$, generating a baseline reward for each state in all tasks. To complete the i_{th} task, the agent generates a reward function $r_i(s)$ from a distribution $P(r_i|r_b)$ conditioned on $r_b(s)$ to plan the motion. Therefore, the motion ζ_i is generated as:

$$P(\zeta_i|r_i)P(r_i|r_b)$$

For the i_{th} task, we want to find the most likely $r_i(s)$ sampled from $P(r_i|r_b)$ that explains the demonstration ζ_i. Assuming all the tasks are independent from each other, the following joint distribution is formulated:

$$\prod_{i=1}^{N_{\mathbb{T}}} P(\zeta_i|r_i)P(r_i|r_b)$$

The reward functions can be found via maximum-likelihood estimation:

$$\min_{r_b(s), r_1(s), \cdots, r_{\mathbb{T}}(s) \in \mathbb{F}} \sum_{i=1}^{N_{\mathbb{T}}} \underbrace{L_i(\zeta_i, r_i(s))}_{\text{IRL loss}} + \underbrace{L(r_i(s), r_b(s))}_{\text{reward sharing loss}} \tag{1}$$

where $\mathbb{F}$ denotes a function space, $L_i(\zeta_i, r_i(s))$ is the negative loglikelihood of $P(\zeta_i|r_i)$, and $L(r_i(s), r_b(s))$ is the negative loglikelihood $P(r_i|r_b)$.

2.2 Loss for Inverse Reinforcement Learning

While many solutions exist for the inverse reinforcement learning problem, we adopt the solution based on function approximation developed in [3] to handle the practical high-dimensional state spaces.

The core idea of the method is to approximate the Bellman Optimality Equation [2] with a function approximation framework. But with a parameterized *VR function*, we describe the summation of the reward function and the discounted optimal value function as:

$$f_i(s, \theta_i) = r_i(s) + \gamma * V^*_i(s), \tag{2}$$

Thus the Bellman Optimality Equation is reformulated as:

$$Q^*_i(s, a) = \sum_{s'|s,a} P^a_{i,ss'} f_i(s', \theta_i), \tag{3}$$

$$V^*_i(s) = \max_{a \in A} \sum_{s'|s,a} P^a_{i,ss'} f_i(s', \theta_i), \tag{4}$$

$$r_i(s) = f_i(s, \theta_i) - \gamma * \max_{a \in A} \sum_{s'|s,a} P^a_{i,ss'} f_i(s', \theta_i). \tag{5}$$

This framework avoids solving the Bellman Optimality Equation. Besides, this formulation can be generalized to other extensions of Bellman Optimality Equation by replacing the max operator with other types of Bellman backup operators. For example, $V^*(s) = \log \sum_{a \in A} \exp Q^*(s,a)$ is used in the maximum-entropy method[4]; $V^*(s) = \frac{1}{k} \log \sum_{a \in A} \exp k * Q^*(s,a)$ is used in Bellman Gradient Iteration [5].

To apply this framework to IRL problems, this work chooses a motion model $p(a|s)$ based on the optimal Q function $Q_i^*(s,a)$ [6]:

$$P(a|s) = \frac{\exp b * Q_i^*(s,a)}{\sum_{\tilde{a} \in A} \exp b * Q_i^*(s,\tilde{a})}, \tag{6}$$

where b is a parameter controlling the degree of confidence in the agent's ability to choose actions based on Q values. Other models can also be used, like $p(a|s) = \exp(Q(s,a) - V(s))$ in [4].

Assuming the approximation function is a neural network, the parameter $\theta_i = \{w, b\}$-weights and biases, the negative log-likelihood of $P(\zeta_i|\theta_i)$ is given by:

$$L_i(\theta_i) = -\sum_{(s,a) \in \zeta_i} (b * Q_i^*(s,a) - \log \sum_{\hat{a} \in A} \exp b * Q_i^*(s,\hat{a})), \tag{7}$$

where the optimal Q function is given by Equation (3). After estimating the parameter θ_i, the value function and reward function can be computed with Equation (2), (4), and (5).

2.3 Loss for Reward Sharing

Since the demonstrator makes minimal changes to adapt the inherent reward function $r_b(s)$ into task-specific one $r_i(s)$, we model the distribution as:

$$P(r_i(s)|r_b(s)) \propto \exp\left(\mathbb{D}(r_i(s), r_b(s))\right)$$

where $\mathbb{D}(r_i(s), r_b(s)))$ measures the difference between $r_i(s)$ and $r_b(s)$. Thus the loss function for reward sharing is given as:

$$L(r_i(s), r_b(s)) = \log Z - \mathbb{D}(r_i(s), r_b(s)))$$

where $\log Z$ is the partition function and remains the same for all $r_i(s)$.

We test several functions as $\mathbb{D}(r_i(s), r_b(s)))$, including L2 loss, huber loss, standard deviation, and information entropy. With the loss function for IRL and reward sharing, the reward functions can be learned via gradient method.

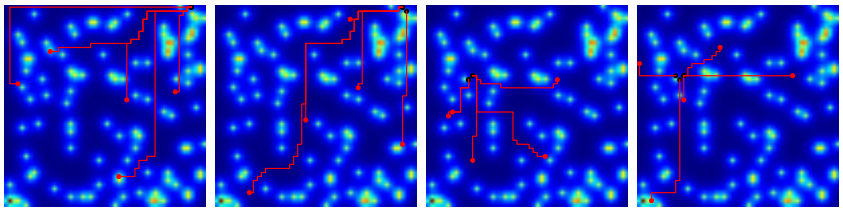
3 Experiments

3.1 Path Planning

We consider a path planning problem on an uneven terrain, where an agent can observe the whole terrain to find the optimal paths from random starting points to arbitrary goal points, but a mobile robot can only observe the agent's demonstrations to learn how to plan paths. Given a starting point and a goal point, an optimal path depends solely on the costs to move across the terrain. To learn the costs, we formulate a Markov Decision Process for each goal point, where a state denotes a small region of the terrain and an action denotes a possible movement. The reward of a state equals to the negative of the cost to move across the corresponding region, while the goal state has an additional reward to attract movements, as shown in Figure 1.

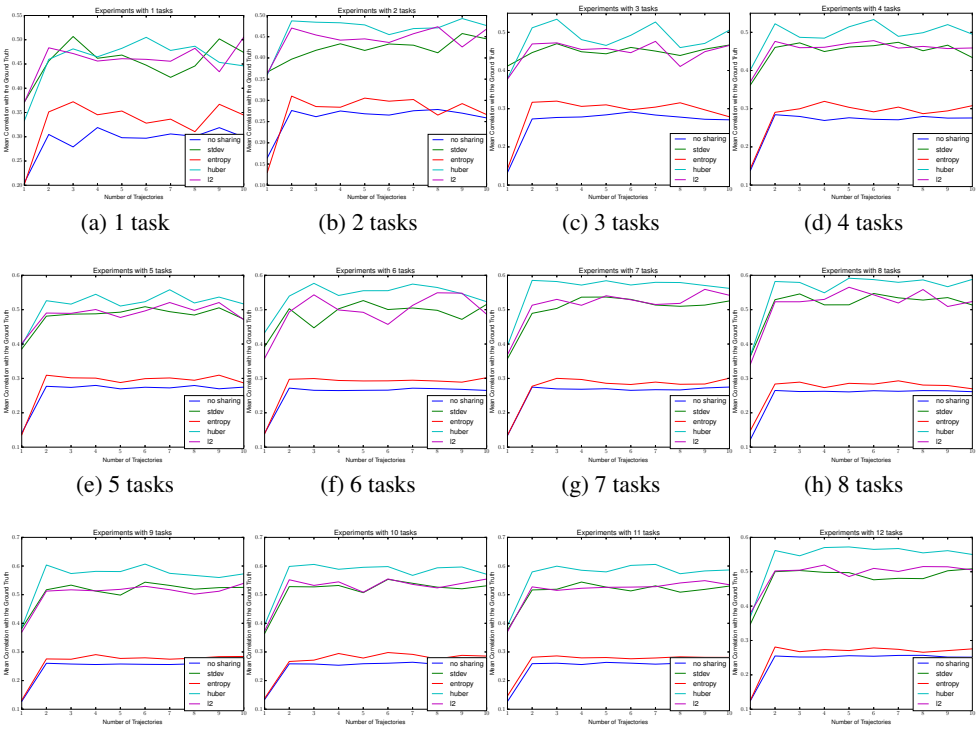
In this work, we create a discretized terrain with several hills, where each hill is defined as a peak of cost distribution and the costs around each hill decay exponentially, and the true cost of a region is the summation of the costs from all hills. Ten worlds are randomly generated, and in each world, twelve tasks are generated, each with a different goal state. For each task, the agent demonstrates ten trajectories, where the length of a trajectory depends on how many steps to reach the goal state.

We evaluate the proposed method with different reward sharing loss functions under different number of tasks and different number of trajectories. The evaluated loss functions include no reward sharing, reward sharing with standard deviation, information entropy, L2 loss, and huber loss. The



(a) (b) (c) (d)

Figure 1: Different behaviors under different goal states and goal rewards: Figure 1a and Figure 1b share the same goal state, but the goal reward of Figure 1b is larger than Figure 1a.Figure 1c and Figure 1d share the same goal state, but the goal reward of Figure 1d is larger than Figure 1c. Five trajectories are plotted in each figure, where red dots denote the starting point and black dots denote the ending point.



(a) 1 task (b) 2 tasks (c) 3 tasks (d) 4 tasks

(e) 5 tasks (f) 6 tasks (g) 7 tasks (h) 8 tasks

(i) 9 tasks (j) 10 tasks (k) 11 tasks (l) 12 tasks

Figure 2: The result with five reward sharing loss functions on 12 tasks with at most 10 demonstrations for each task in 10 environment.

number of tasks ranges from 1 to 12, and for each task, the number of trajectories ranges from 1 to 10. The learning rate is 0.01, with Adam optimizer. The accuracy of a reward is computed as the correlation coefficient between the learned reward function and the ground truth one. The results are shown in Figure 2.

The result shows that the meta learning step can significantly improve the accuracy of reward learning, among which the huber loss function leads to the best performance in average. L2 loss and standard deviation have similar performance, not surprisingly. However, the information entropy has a really bad performance.

References

[1] Andrew Y Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *in Proc. 17th International Conf. on Machine Learning*, 2000.

[2] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[3] Kun Li and Joel W Burdick. Large-scale inverse reinforcement learning via function approximation for clinical motion analysis. *arXiv preprint arXiv:1707.09394*, 2017.

[4] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, pages 1433–1438, 2008.

[5] K. Li and J. W. Burdick. Bellman Gradient Iteration for Inverse Reinforcement Learning. *ArXiv e-prints*, July 2017.

[6] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2586–2591, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.