Stochastic Inverse Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

The goal of the inverse reinforcement learning (IRL) problem is to recover the reward functions from expert demonstrations. However, the IRL problem like any ill-posed inverse problem suffers the congenital defect that the policy may be optimal for many reward functions, and expert demonstrations may be optimal for many policies. In this work, we generalize the IRL problem to a well-posed expectation optimization problem *stochastic inverse reinforcement learning* (SIRL) to recover the probability distribution over reward functions. We adopt the Monte Carlo expectation-maximization (MCEM) method to estimate the parameter of the probability distribution as the first solution to the SIRL problem. The solution is *succinct, robust, and transferable* for a learning task and can generate alternative solutions to the IRL problem. Through our formulation, it is possible to observe the intrinsic property for the IRL problem from a *global* viewpoint, and our approach achieves a considerable performance on the *objectworld*.

1 Introduction

The IRL problem addresses an inverse problem that a set of expert demonstrations determines a reward function over a Markov decision process (MDP) if the model dynamics are known [Rus98, NR+00]. The recovered reward function provides a *succinct, robust, and transferable* definition of the learning task and completely determines the optimal policy. However, the IRL problem is ill-posed that the policy may be optimal for many reward functions and expert demonstrations may be optimal for many policies. For example, all policies are optimal for a constant reward function. In a real-world scenario, experts always act sub-optimally or inconsistently, which is another challenge.

To overcome these limitations, two classes of probabilistic approaches for the IRL problem are proposed, i.e., Bayesian inverse reinforcement learning (BIRL) [RA07] based on Bayesians' maximum a posteriori (MAP) estimation and maximum entropy IRL (MaxEnt) [ZMBD08, Zie10] based on frequentists' maximum likelihood (MLE) estimation. BIRL solves for the distribution of reward functions without an assumption that experts behave optimally, and encode the external *a priori* information in a choice of a prior distribution. However, BIRL also suffers from the practical limitation that a large number of algorithmic iterations is required for the procedure of Markov chain Monte Carlo (MCMC) in a sampling of posterior over reward functions. Advanced techniques, for example Kernel technique [MH12] and gradient method [CK11], are proposed to improve the efficiency and tractability of this situation.

MaxEnt employs the principle of maximum entropy to resolve the ambiguity in choosing demonstrations over a policy. This class of methods, inheriting the merits from previous non-probabilistic IRL approaches including [NR+00, AN04, RBZ06, ADNT08, SS08, HGE16], imposes regular structures of reward functions in a combination of hand-selected features. Formally, the reward function is a linear or nonlinear combination of the feature basis functions which consists of a set of real-valued functions $\{\phi_i(s,a)\}_{i=1}$ hand-selected by experts. The goal of this approach is to find the best-fitting weights of feature basis functions through the MLE approach. [WOP15] and [LPK11] use deep

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

neural networks and Gaussian processes to fit the parameters based on demonstrations respectively but still suffer from the problem that the true reward shaped by the changing environment dynamics.

Influenced by the work of [FCAL16, FLA16], [FLL17] propose a framework called adversarial IRL (AIRL) to recover robust reward functions in a changing dynamics based on adversarial learning and achieves superior results. Compared with AIRL, another adversarial method called generative adversarial imitation learning (GAIL) [HE16] seeks to directly recover the expert's policy rather than reward functions. Many follow-up methods enhance and extend GAIL for multipurpose in various application scenarios [LSE17, HCS+17, WMR+17]. However, GAIL is in a lack of an explanation of expert's behavior and a portable representation for the knowledge transfer which are the merits of the class of the MaxEnt approach, because the MaxEnt approach is equipped with the "transferable" regular structures over reward functions.

In this paper, under the framework of the MaxEnt approach, we propose a generalized perspective of studying the IRL problem called *stochastic inverse reinforcement learning* (SIRL). It is formulated as an expectation optimization problem aiming to recover a probability distribution over the reward function from expert demonstrations. The solution of SIRL is *succinct* and *robust* for the learning task in the meaning that it can generate more than one weight over feature basis functions which compose alternative solutions to the IRL problem. Benefits of the class of the MaxEnt method, the solution to our generalized problem SIRL is also *transferable*. Since of the intractable integration in our formulation, we employ the Monte Carlo expectation-maximization (MCEM) approach [WT90] to give the first solution to the SIRL problem in a model-based environment. In general, the solutions to the IRL problem are not always best-fitting in the previous approaches because a highly nonlinear inverse problem with the limited information is very likely to get trapped in a secondary maximum in the recovery. Taking advantage of the Monte Carlo mechanism of a *global* exhaustive search, our MCEM approach avoids the secondary maximum and theoretically convergent demonstrated by pieces of literature [CJJ05, CL95]. Our approach is also quickly convergent because of the preset simple geometric configuration over weight space in which we approximate it with a Gaussian Mixture Model (GMM). Hence, our approach works well in a real-world scenario with a small and variability set of expert demonstrations.

In particular, the contributions of this paper are threefold:

1. We generalize the IRL problem to a well-posed expectation optimization problem SIRL.
2. We provide the first theoretically existing solution to SIRL by the MCEM approach.
3. We show the effectiveness of our approach by comparing the performance of the proposed method to those of the previous algorithms on the *objectworld*.

2 Problem Statement

Given an MDP$\backslash R := (\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma)$ with a known transition function $\mathcal{T} := \mathbb{P}(s_{t+1} = s' \big| s_t = s, a_t = a)$ for $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ and a hand-crafted reward feature basis function $\{\phi_i(s)\}_{i=1}^M$. A stochastic regular structure on the reward function assumes weights $\mathcal{W}$ of the reward feature functions $\phi_i(s)$, which are random variables with a reward conditional probability distribution $\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E)$ conditional on expert demonstrations ζ^E. Parametrizing $\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E)$ with parameter Θ, our aim is to estimate the best-fitting parameter Θ^* from the expert demonstrations ζ^E, such that $\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta^*)$ more likely generates weights to compose reward functions as the ones derived from expert demonstrations, which is called *stochastic inverse reinforcement learning* problem.

Suppose a representative trajectory class $\mathcal{C}_\epsilon^E$ satisfies that each trajectory element set $\mathcal{O} \in \mathcal{C}_\epsilon^E$ is a subset of expert demonstrations ζ^E with the cardinality at least $\epsilon \cdot |\zeta^E|$, where ϵ is a preset threshold and $|\zeta^E|$ is the number of expert demonstrations, written as,

$$\mathcal{C}_\epsilon^E := \big\{\mathcal{O} \big| \mathcal{O} \subset \zeta^E \text{ with } |\mathcal{O}| \geq \epsilon \cdot |\zeta^E|\big\}.$$

Integrate out unobserved weights $\mathcal{W}$, and the SIRL problem is formulated to estimate parameter Θ on an expectation optimization problem over the representative trajectory class as follows:

$$\Theta^* := \arg\max_{\Theta} \mathbb{E}_{\mathcal{O} \in \mathcal{C}_\epsilon^E} \big[\int_{\mathcal{W}} f_{\mathcal{M}}(\mathcal{O}, \mathcal{W}|\Theta) d\mathcal{W} \big], \tag{1}$$

where trajectory element set $\mathcal{O}$ assumes to be uniformly distributed for the sake of simplicity in this study, and $f_{\mathcal{M}}$ is the conditional joint probability density function of trajectory element $\mathcal{O}$ and weights $\mathcal{W}$ for reward feature functions conditional on parameter Θ.

2.0.1 Note:

- Problem 1 is well-posed and typically not intractable analytically.
- Trajectory element set $\mathcal{O}$ is usually known from the rough estimation of the statistics in expert demonstrations in practice.
- We introduce *representative trajectory class* to overcome the limitation of the sub-optimality of expert demonstrations ζ^E [RA07], which is also called a lack of sampling representativeness in statistics [KM79], e.g., driver's demonstrations encode his own preferences in driving style but may not reflect the true rewards of an environment. With the technique of representative trajectory class, each subset of driver's demonstrations (i.e. trajectory element set $\mathcal{O}$) constitutes a subproblem in Problem 1.

3 Experiments

We evaluate our approach on a classic environment *objectworld* introduced by [LPK11] which is a particularly challenging environment with a large number of irrelevant features and the highly nonlinearity of the reward functions. Note that since almost only *objectworld* provides a tool that allows analysis and display the evolution procedure of the SIRL problem in a 2D heat map, we skip the typical invisible physics-based control tasks for the evaluation of our approach, i.e. cartpole [BSA83], mountain car [Moo90], MuJoCo [TET12], and etc.

We employ the expected value difference (EVD) proposed by [LPK11] to be the metric of optimality as follows:

$$\mathrm{EVD}(\mathcal{W}) := \mathbb{E}\Big[\sum_{t=0}^{\infty}\gamma^t \cdot R(s_t, a_t)\Big|\pi^*\Big] - \mathbb{E}\Big[\sum_{t=0}^{\infty}\gamma^t \cdot R(s_t, a_t)\Big|\pi(\mathcal{W})\Big],$$

which is a measure of the difference between the expected reward earned under the optimal policy π^*, given by the true reward, and the policy derived from the rewards sampling from our reward conditional probability distribution $\mathcal{D}(\mathcal{W}|\zeta^E, \Theta^*)$, where Θ^* is the best estimation parameter in our approach.

3.1 Evaluation Procedure and Analysis

In this section, we design three tasks to evaluate the effectiveness of our generative model *reward conditional probability distribution* $\mathcal{D}(\mathcal{W}|\zeta^E, \Theta^*)$. For each task, the environment setting is as follows. The instance of a 10×10 *objectworld* has 25 random objects with 2 colors and a 0.9 discount factor. 20 expert demonstrations are generated according to the given optimal policy for the recovery. The length of each expert demonstration is 5-grid size trajectory length. Four algorithms for the evaluation includes MaxEnt, DeepMaxEnt, SIRL, and DSIRL, where SIRL and DSIRL are implemented as Algorithm 2 in Appendix. The weights are drawn from reward conditional probability distribution $\mathcal{D}(\mathcal{W}|\zeta^E, \Theta^*)$ as the coefficients of feature basis functions $\{\phi_i(s, a)\}_{i=1}$.

In our evaluation, SIRL and DSIRL start from 10 samples and double the sample size per iteration until it converges for the convergence issue, refer to Section B.2. In the first stage, the epochs of algorithm iteration are set to 20 and the learning rates are 0.01. The parameter ϵ in representative trajectory set $\mathcal{O}_\epsilon^E$ is preset as 0.95. In the second stage, 3-component GMM for SIRL and DSIRL is set with at most 1000 iterations before convergence. Additionally, the architecture of neural networks in DeepMaxEnt and DSIRL are implemented as 3-layer fully-connected with the sigmoid function.

3.1.1 Recovery Experiment

In the recovery experiment, we compare the true reward function, the optimal policy, and the optimal value with ones derived from expert demonstrations under the four methods. Since our approach is an average of all the outcomes which are prone to imitate the true optimal value from expert

demonstrations, we use the mean of reward conditional probability distribution for SIRL and DSIRL as a comparison.

In Figure 1, the EVD of the optimal values in the last row are 48.9, 31.1, 33.7 and 11.3 for four methods respectively, and the covariances of GMM model for SIRL and DSIRL are limited up to 5.53 and 1.36 on each coordinate respectively. It yields that in a highly nonlinear inverse problem, the recovery abilities of SIRL and DISRL are better than MaxEnt's and DeepMaxEnt's respectively. The reason is mainly because Monte Carlo mechanism in our approach alleviates the problem of getting stuck in local minima by allowing random exit from it.

3.1.2 Robustness Experiment

In the robustness experiment, we evaluate the robustness of our approach that solutions generated by $\mathcal{D}(\mathcal{W}|\zeta^E, \Theta^*)$ are effective to the IRL problem. To capture the robust solutions, we design the following generative algorithm with the pseudocode in Algorithm 1.

In the right generative algorithm, we use Frobenius norm to measure the distance between weights drawn from $\mathcal{D}(\mathcal{W}|\zeta^E, \Theta^*)$ as follows,

$$||\mathcal{W}||_{\mathcal{F}} := \sqrt{\mathrm{Tr}(\mathcal{W} \cdot \mathcal{W}^T)}.$$

We also constrain that each drawn weight $\mathcal{W} \sim \mathcal{D}(\mathcal{W}|\zeta^E, \Theta^*)$ in the solution set $\mathcal{G}$ satisfies as follows,

$$||\mathcal{W} - \mathcal{W}'||_{\mathcal{F}} > \delta \text{ and } \mathrm{EVD}(\mathcal{W}) < \epsilon,$$

where $\mathcal{W}'$ represents any member in the solution set $\mathcal{G}$. δ, ϵ are the preset thresholds in the generative algorithm.

Algorithm 1: Generative Algorithm

Input: $\mathcal{D}(\mathcal{W}|\zeta^E, \Theta^*)$, required solution set size N, and preset thresholds ϵ and δ.
Output: Solution set $\mathcal{G} := \{\mathcal{W}_i\}_{i=1}^N$.

```
while i < N do
    W ~ D(W|ζ^E, Θ*) ;
    for any W' ∈ S do
        if ||W − W'||_F > δ and EVD(W) < ε then
            G ← W;
        end
    end
    i ← i + 1;
end
```

In Figure 2, the right column figures are generated from weights in the solution set $\mathcal{G}$ whose EVD values are around 10.2. Note that the recovered reward function in the first row has a similar but different pattern appearance. The optimal value derived from these recovered reward functions has a very small EVD value with the true reward. It yields the effectiveness of our *robust* generative model which can generate more than one solutions to the IRL problem.

3.1.3 Hyperparameter Experiment

In the hyperparameter experiment, we evaluate the effectiveness of our approach under the influence of different preset quantities and qualities of expert demonstrations. The amount of information carried in expert demonstrations composes a specific learning environment, and hence it has an impact on the effectiveness of our generative model. We verify three hyperparameters including the number of expert demonstrations in Figure 3, the trajectory length of expert demonstrations in Figure 4 and the portion size in representative trajectory class $\mathcal{C}_\epsilon^E$ in Figure 5 on the *objectworld*. The shadow of the line in the figures represents the standard error for each experimental trail. Notice that the EVDs for SIRL and DSIRL are both decreasing as the number and the trajectory length of expert demonstrations, and the portion size in the representative trajectory class are increasing. A notable point in Figure 3 is that very few expert demonstrations (less than 200) for our approach also yields a small EVDs, which manifests the merit of Monte Carlo mechanism in our approach.

4 Conclusion

In this paper, we propose a generalized problem SIRL for the IRL problem to get the distribution of reward functions. The new problem is well-posed and we employ the method of MCEM to give the first *succinct, robust, and transferable* solution. In the experiment, we evaluate our approach on the *objectworld* and the experimental results confirm the effectiveness of our approach.

References

[ADNT08] Pieter Abbeel, Dmitri Dolgov, Andrew Y Ng, and Sebastian Thrun. Apprenticeship learning for motion planning with application to parking lot navigation. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1083–1090. IEEE, 2008.

[AN04] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004.

[BH99] James G Booth and James P Hobert. Maximizing generalized linear mixed model likelihoods with an automated monte carlo em algorithm. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(1):265–285, 1999.

[BSA83] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

[CJJ05] Brian S Caffo, Wolfgang Jank, and Galin L Jones. Ascent-based monte carlo expectation–maximization. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):235–251, 2005.

[CK11] Jaedeug Choi and Kee-Eung Kim. Map inference for bayesian inverse reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1989–1997, 2011.

[CL95] KS Chan and Johannes Ledolter. Monte carlo em estimation for time series models involving counts. *Journal of the American Statistical Association*, 90(429):242–252, 1995.

[FCAL16] Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016.

[FLA16] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58, 2016.

[FLL17] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[FM+03] Gersende Fort, Eric Moulines, et al. Convergence of the monte carlo expectation maximization for curved exponential families. *The Annals of Statistics*, 31(4):1220–1259, 2003.

[HCS+17] Karol Hausman, Yevgen Chebotar, Stefan Schaal, Gaurav Sukhatme, and Joseph J Lim. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 1235–1245, 2017.

[HE16] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in neural information processing systems*, pages 4565–4573, 2016.

[HGE16] Jonathan Ho, Jayesh Gupta, and Stefano Ermon. Model-free imitation learning with policy optimization. In *International Conference on Machine Learning*, pages 2760–2769, 2016.

[KM79] William Kruskal and Frederick Mosteller. Representative sampling, iii: The current statistical literature. *International Statistical Review/Revue Internationale de Statistique*, pages 245–265, 1979.

[LPK11] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. In *Advances in Neural Information Processing Systems*, pages 19–27, 2011.

[LSE17] Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. In *Advances in Neural Information Processing Systems*, pages 3812–3822, 2017.

[MH12] Bernard Michini and Jonathan P How. Improving the efficiency of bayesian inverse reinforcement learning. In *2012 IEEE International Conference on Robotics and Automation*, pages 3651–3656. IEEE, 2012.

[MNW+18] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging {AI} applications. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pages 561–577, 2018.

[Moo90] Andrew William Moore. Efficient memory-based learning for robot control. 1990.

[NR+00] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[RA07] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *IJCAI*, volume 7, pages 2586–2591, 2007.

[RBZ06] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum margin planning. In *Proceedings of the 23rd international conference on Machine learning*, pages 729–736. ACM, 2006.

[Rus98] Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 101–103, 1998.

[SS08] Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in neural information processing systems*, pages 1449–1456, 2008.

[TET12] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[WMR+17] Ziyu Wang, Josh S Merel, Scott E Reed, Nando de Freitas, Gregory Wayne, and Nicolas Heess. Robust imitation of diverse behaviors. In *Advances in Neural Information Processing Systems*, pages 5320–5329, 2017.

[WOP15] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

[WT90] Greg CG Wei and Martin A Tanner. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American statistical Association*, 85(411):699–704, 1990.

[Zie10] Brian D Ziebart. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. 2010.

[ZMBD08] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

A Preliminary

An MDP is a tuple $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, and the transition function (a.k.a. model dynamics) $\mathcal{T} := \mathbb{P}(s_{t+1} = s' \big| s_t = s, a_t = a)$ for $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ is the probability of being current state s, taking action a and yielding next state s'. Reward function $\mathcal{R}(s, a)$ is a real-valued function and $\gamma \in [0, 1)$ is the discount factor. A policy $\pi : \mathcal{S} \to \mathcal{A}$ is deterministic or stochastic, where the deterministic one is written as $a = \pi(s)$, and the stochastic one is as a conditional distribution $\pi(a|s)$. Sequential decisions are recorded in a series of episodes which consist of states s, actions a, and rewards r. The goal of reinforcement learning aims to get optimal policy π^* for maximizing the expected total reward, i.e.

$$\pi^* := \arg\max_{\pi} \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \cdot \mathcal{R}(s_t, a_t) \Big| \pi \Big].$$

Given an MDP without a reward function $\mathcal{R}$, i.e., MDP\R=$\langle \mathcal{S}, \mathcal{A}, \mathcal{T} \rangle$, and m expert demonstrations $\zeta^E := \{\zeta^1, \cdots, \zeta^m\}$. Each expert demonstration ζ^i is a sequential of state-action pairs. The goal of the IRL problem is to estimate the unknown reward function $\mathcal{R}(s, a)$ from expert demonstrations ζ^E and the reward functions compose a complete MDP. The estimated complete MDP yields an optimal policy that acts as closely as the expert demonstrations.

A.1 Regular Structure of Reward Functions

In this section, we provide a formal definition of the regular (linear/nonlinear) structure of reward functions. The linear structure [NR+00, ZMBD08, SS08, HGE16] is an M linear combination of feature basis functions, written as,

$$\mathcal{R}(s, a) := \sum_{i}^{M} \alpha_i \cdot \phi_i(s, a),$$

where $\phi_i : \mathcal{S} \times \mathcal{A} \longmapsto \mathbb{R}^d$ are a d-dimensional feature functions hand-selected by experts.

The nonlinear structure [WOP15] is of the form as follows,

$$\mathcal{R}(s, a) := \mathcal{N}\big(\phi_1(s, a), \cdots, \phi_M(s, a)\big),$$

where $\mathcal{N}$ is neural networks of hand-crafted reward feature basis functions $\{\phi_i(s, a)\}_{i=1}^{M}$.

B Methodology

In this section, we propose a novel approach to estimate the best-fitting parameter Θ^* in Problem 1, which is called the two-stage hierarchical method, *a variant of MCEM* method.

B.1 Two-stage Hierarchical Method

The two-stage hierarchical method requires us to write parameter Θ in a profile form $\Theta := (\Theta_1, \Theta_2)$. The conditional joint density $f_{\mathcal{M}}(\mathcal{O}, \mathcal{W}|\Theta)$ in Equation 1 can be written as the product of two conditional densities $g_{\mathcal{M}}$ and $h_{\mathcal{M}}$ as follows:

$$f_{\mathcal{M}}(\mathcal{O}, \mathcal{W}|\Theta_1, \Theta_2) = g_{\mathcal{M}}(\mathcal{O}|\mathcal{W}, \Theta_1) \cdot h_{\mathcal{M}}(\mathcal{W}|\Theta_2). \tag{2}$$

Take the log of both sides in Equation 2, and we have

$$\log f_{\mathcal{M}}(\mathcal{O}, \mathcal{W}|\Theta_1, \Theta_2) = \log g_{\mathcal{M}}(\mathcal{O}|\mathcal{W}, \Theta_1) + \log h_{\mathcal{M}}(\mathcal{W}|\Theta_2). \tag{3}$$

We maximize the right side of Equation 3 over the profile parameter Θ in the expectation-maximization (EM) update steps at the t-th iteration independently as follows,

$$\Theta_1^{t+1} := \arg\max_{\Theta_1} \mathbb{E}\Big(\log g_{\mathcal{M}}(\mathcal{O}|\mathcal{W}, \Theta_1) \Big| \mathcal{C}_{\epsilon}^{E}, \Theta^t \Big); \tag{4}$$

$$\Theta_2^{t+1} := \arg\max_{\Theta_2} \mathbb{E}\Big(\log h_{\mathcal{M}}(\mathcal{W}|\Theta_2) \Big| \Theta^t \Big). \tag{5}$$

B.1.1 Initialization

We randomly initialize profile parameter $\Theta^0 := (\Theta_1^0, \Theta_2^0)$ and sample a collection of N_0 rewards weights $\{\mathcal{W}_1^{\Theta^0}, \cdots, \mathcal{W}_N^{\Theta^0}\} \sim \mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta_2^0)$. The reward weights $\mathcal{W}_i^{\Theta^0}$ compose reward $R_{\mathcal{W}_i^{\Theta^0}}$ in each learning task $\mathcal{M}_i^0 := (\mathcal{S}, \mathcal{A}, \mathcal{T}, R_{\mathcal{W}_i^{\Theta^0}}, \gamma)$ for $i = 1, \cdots, N_0$.

B.1.2 First Stage

In the first stage, we aim to update parameter Θ_1 in the intractable expectation of Equation 4. Specifically, we take a Monte Carlo method to estimate model parameters Θ_1^{t+1} in an empirical expectation at the t-th iteration,

$$\mathcal{E}\big[\log g_{\mathcal{M}}(\mathcal{O}|\mathcal{W}, \Theta_1^{t+1})\Big|\mathcal{C}_\epsilon^E, \Theta^t\big] := \frac{1}{N_t} \cdot \sum_{i=1}^{N_t} \log g_{\mathcal{M}_i^t}(\mathcal{O}_i^t|\mathcal{W}_i^{\Theta^t}, \Theta_1^{t+1}), \tag{6}$$

where reward weights at the t-th iteration $\mathcal{W}_i^{\Theta^t}$ are randomly drawn from the reward conditional probability distribution $\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta^t)$ and compose a set of learning tasks $\mathcal{M}_i^t := (\mathcal{S}, \mathcal{A}, \mathcal{T}, R_{\mathcal{W}_i^{\Theta^t}}, \gamma)$ with a trajectory element set $\mathcal{O}_i^t$ uniformly drawn from representative trajectory class $\mathcal{C}_\epsilon^E$, for $i = 1, \cdots, N_t$.

The parameter Θ_1^{t+1} in Equation 6 has N_t coordinates written as $\Theta_1^{t+1} := \Big((\Theta_1^{t+1})_1, \cdots, (\Theta_1^{t+1})_{N_t}\Big)$. For each learning task $\mathcal{M}_i^t$, the i-th coordinate $(\Theta_1^{t+1})_i$ is derived from maximization of a posteriori,

$$(\Theta_1^{t+1})_i := \arg\max_\theta \log g_{\mathcal{M}_i^t}\big(\mathcal{O}_i^t|\mathcal{W}_i^{\Theta^t}, \theta\big),$$

which is a convex formulation maximized by a gradient ascent method.

In practice, we move m steps uphill to the optimum in each learning task $\mathcal{M}_i^t$. The update formula of m-step reward weights ${\mathcal{W}^m}_i^{\Theta^t}$ is written as

$${\mathcal{W}^m}_i^{\Theta^t} := \mathcal{W}_i^{\Theta^t} + \sum_{i=1}^{m} \lambda_i^t \cdot \nabla_{(\Theta_1)_i} \log g_{\mathcal{M}_i^t}\big(\mathcal{O}_i^t|\mathcal{W}_i^{\Theta^t}, (\Theta_1)_i\big),$$

where the learning rate λ_i^t at the t-th iteration is preset. Hence, the parameter Θ_1^{t+1} is represented as $\Theta_1^{t+1} := \Big({\mathcal{W}^m}_1^{\Theta^t}, \cdots, {\mathcal{W}^m}_{N_t}^{\Theta^t}\Big)$.

B.1.3 Second Stage

In the second stage, we aim to update parameter Θ_2 in the intractable expectation of Equation 5. Specifically, we consider the empirical expectation at the t-th iteration as follows,

$$\mathcal{E}\big(\log h_{\mathcal{M}}(\mathcal{W}|\Theta_2^{t+1})\Big|\Theta^t\big) := \frac{1}{N_t} \cdot \sum_{i=1}^{N_t} \log h_{\mathcal{M}_i^t}({\mathcal{W}^m}_i^{\Theta^t}|\Theta_2^{t+1}), \tag{7}$$

where $h_{\mathcal{M}}$ is implicit but fitting a set of m-step reward weights $\{{\mathcal{W}^m}_i^{\Theta^t}\}_{i=1}^{N_t}$ in a generative model yields a large empirical expectation value. The reward conditional probability distribution $\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta_2^{t+1})$ is a generative model formulated as a Gaussian Mixture Model (GMM), i.e.

$$\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta_2^{t+1}) := \sum_{k=1}^{K} \alpha_k \cdot \mathcal{N}(\mathcal{W}|\mu_k, \Sigma_k),$$

where $\alpha_k \geq 0$ and $\sum_{k=1}^K \alpha_k = 1$, and parameter set $\Theta_2^{t+1} := \{\alpha_k; \mu_k, \Sigma_k\}_{k=1}^K$.

We estimate parameter Θ_2^{t+1} in GMM by EM approach and initialize GMM with the t-th iteration parameter Θ_2^t with the procedure as follows:

For $i = 1, \cdots, N_t$, we have

- Expectation Step: Compute responsibility γ_{ij} for m-step reward weight ${\mathcal{W}^m}_i^{\Theta^t}$,

$$\gamma_{ij} := \frac{\alpha_j \cdot \mathcal{N}({\mathcal{W}^m}_i^{\Theta^t} | \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \alpha_k \cdot \mathcal{N}({\mathcal{W}^m}_i^{\Theta^t} | \mu_k, \Sigma_k)}.$$

- Maximization Step: Compute weighted mean μ_j and variance Σ_j by,

$$\mu_j := \frac{\sum_{i=1}^{N_t} \gamma_{ij} \cdot {\mathcal{W}^m}_i^{\Theta^t}}{\sum_{i=1}^{N_t} \gamma_{ij}}; \quad \alpha_j := \frac{1}{N_t} \cdot \sum_{i=1}^{N_t} \gamma_{ij}; \quad \Sigma_j := \frac{\sum_{i=1}^{N_t} \gamma_{ij} \cdot ({\mathcal{W}^m}_i^{\Theta^t} - \mu_j) \cdot ({\mathcal{W}^m}_i^{\Theta^t} - \mu_j)^T}{\sum_{i=1}^{N_t} \gamma_{ij}}.$$

After EM converges, $\Theta_2^{t+1} := \{\alpha_k; \mu_k, \Sigma_k\}_{k=1}^{K}$ and profile parameter $\Theta^{t+1} := (\Theta_1^{t+1}, \Theta_2^{t+1})$.

Finally, when the two-stage hierarchical method converges, parameter Θ_2 of profile parameter Θ is our desired best-fitting parameter Θ^* for $\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta^*)$.

B.1.4 Pseudocode

Algorithm 2: Stochastic Inverse Reinforcement Learning

Input: Model-based environment $(\mathcal{S}, \mathcal{A}, \mathcal{T})$ and expert demonstrations ζ^E, Monte Carlo sample size N_0, and preset thresholds δ_{MCEM} and ϵ_{MCEM}.
Output: Reward conditional probability distribution $\mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta^*)$.

Initialization: Randomly initialization of profile parameter $\Theta^0 := (\Theta_1^0, \Theta_2^0)$;
while *stopping criteria not satisfied (refer to Section B.2)* **do**
  Draw N_t reward weights $\mathcal{W}_i^{\Theta^t} \sim \mathcal{D}^{\mathcal{R}}(\mathcal{W}|\zeta^E, \Theta_2^t)$ to compose learning task $\mathcal{M}_i^t$ with uniformly drawn trajectory element set $\mathcal{O}_i^t$;
  # *First Stage: Monte Carlo estimation of weights for reward function*;
  for $\mathcal{M}_i^t$ **do**
    Evaluate $\nabla_{(\Theta_1)_i} \log g_{\mathcal{M}_i^t}\big(\mathcal{O}_i^t | \mathcal{W}_i^{\Theta^t}, (\Theta_1)_i\big)$;
    Compute updated weight parameter
    ${\mathcal{W}^m}_i^{\Theta^t} \leftarrow \mathcal{W}_i^{\Theta^t} + \sum_{i=1}^{m} \lambda_i^t \cdot \nabla_{(\Theta_1)_i} \log g_{\mathcal{M}_i^t}\big(\mathcal{O}_i^t | \mathcal{W}_i^{\Theta^t}, (\Theta_1)_i\big)$;
  end
  Update $\Theta_1^{t+1} \leftarrow \{{\mathcal{W}^m}_i^{\Theta^t}\}_{i=1}^{N_t}$;
  # *Second Stage: Fit GMM with m-step reward weight* $\{{\mathcal{W}^m}_i^{\Theta^t}\}_{i=1}^{N_t}$ with EM parameter initialization Θ_2^t ;
  while *EM not converge* **do**
    Expectation Step: $\gamma_{ij} \leftarrow \frac{\alpha_j \cdot \mathcal{N}({\mathcal{W}^m}_i^{\Theta^t} | \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \alpha_k \cdot \mathcal{N}({\mathcal{W}^m}_i^{\Theta^t} | \mu_k, \Sigma_k)}$;
    Maximization Step:

$$\mu_j \leftarrow \frac{\sum_{i=1}^{N_t} \gamma_{ij} \cdot {\mathcal{W}^m}_i^{\Theta^t}}{\sum_{i=1}^{N_t} \gamma_{ij}}; \quad \alpha_j \leftarrow \frac{1}{N_t} \cdot \sum_{i=1}^{N_t} \gamma_{ij};$$
$$\Sigma_j \leftarrow \frac{\sum_{i=1}^{N_t} \gamma_{ij} \cdot ({\mathcal{W}^m}_i^{\Theta^t} - \mu_j) \cdot ({\mathcal{W}^m}_i^{\Theta^t} - \mu_j)^T}{\sum_{i=1}^{N_t} \gamma_{ij}};$$

  end
  Update Θ_2^{t+1} and profile parameter $\Theta^{t+1} \leftarrow (\Theta_1^{t+1}, \Theta_2^{t+1})$;
end

B.2 Termination Criteria

In this section, we discuss the termination criteria in our algorithm. EM terminates usually when the parameters do not substantively change after enough iterations. For example, one classic termination

criterion in EM terminates at the t-th iteration satisfying as follows,

$$\max \frac{|\theta^t - \theta^{t-1}|}{|\theta^t| + \delta_{EM}} < \epsilon_{EM},$$

for user-specified δ_{EM} and ϵ_{EM}, where θ is the model parameter in EM.

However, the same termination criterion for MCEM has a risk of early terminating because of the Monte Carlo error in the update step. Hence, we adopt a practical method in which the following stopping criterion holds in three consecutive times,

$$\max \frac{|\Theta^t - \Theta^{t-1}|}{|\Theta^t| + \delta_{MCEM}} < \epsilon_{MCEM},$$

for user-specified δ_{MCEM} and ϵ_{MCEM} [BH99]. Other stopping criteria for MCEM refers to [CJJ05, CL95].

B.3 Convergence Issue

The convergence issue of MCEM is more complicated than EM. In light of model-based interactive MDP\R, we can always increase the sample size during each iteration of MCEM. In practice, we require the Monte Carlo sample size satisfy the following inequality,

$$\sum_t \frac{1}{N_t} < \infty.$$

Additional requirement for the convergence property is a compact assumption over that parameter space. A comprehensive proof refers to [CL95, FM+03].

C Experiments

C.1 Objectworld

The *objectworld* is a learning environment for the IRL problem. It is an $N \times N$ grid board with colored objects placed in randomly selected cells. Each colored object is assigned one inner color and one outer color from C preselected colors. Each cell on the grid board is a state, and stepping to four neighbor cells (up, down, left, right) or staying in place (stay) are five actions with a 30% chance of moving in a random direction.

The ground truth of reward function is defined in the following way. Suppose two primary colors of C preselected colors are red and blue. The reward of a state is 1 if the state is within 3 steps of an outer red object and 2 steps of an outer blue object, -1 if the state is within 3 steps of an outer red object, and 0 otherwise. The other pairs of inner and outer colors are distractors. Continuous and discrete versions of feature basis functions are provided. For the continuous version, $\phi(s)$ is a $2C$-dimensional real-valued feature vector. Each dimension records the Euclidean distance from the state to objects. For example, the first and second coordinates are the distances to the nearest inner and outer red object respectively, and so on through all C colors. For the discrete version, $\phi(s)$ is a $(2C \cdot N)$-dimensional binary feature vector. Each N-dimensional vector records a binary representation of distance to the nearest inner or outer color object with the d-th coordinate 1 if the corresponding continuous distance is less than d.

C.1.1 Evaluation Platform

All the methods are implemented in Python 3.5 and Theano 1.0.0 with a machine learning distributed framework *Ray* [MNW+18]. The experiments are conducted on a machine with Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz and Nvidia GeForce GTX 1070 GPU.

C.1.2 Experimental Results

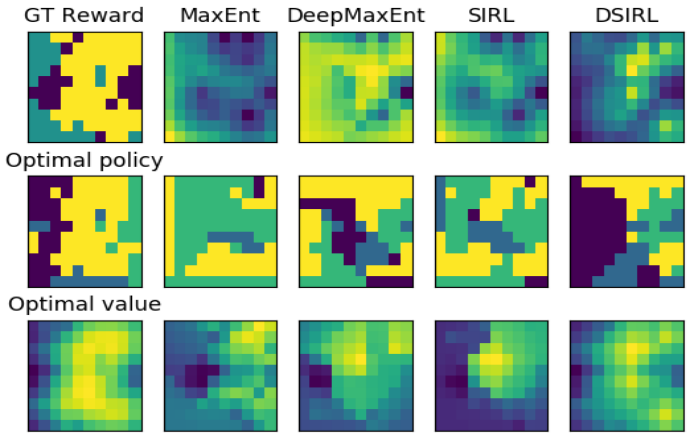

Figure 1: Results for recovery experiment.

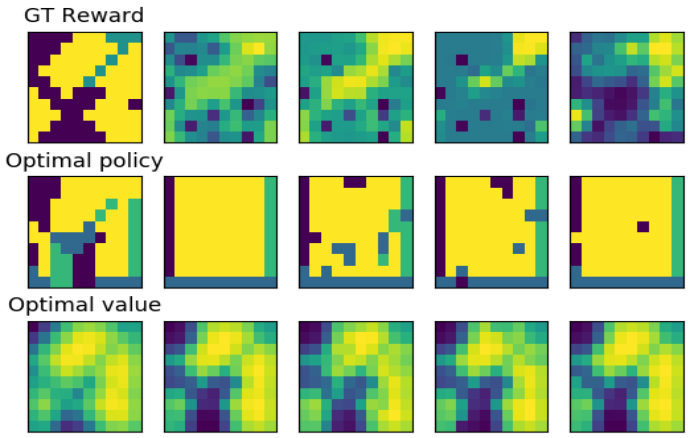

Figure 2: Results for recovery experiment.

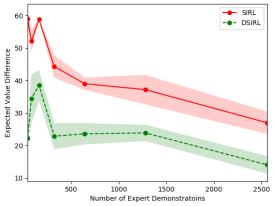

Figure 3: Results under 40, 80, 160, 320, 640, 1280 and 2560 expert demonstrations.

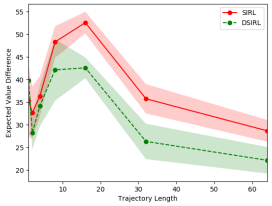

Figure 4: Results under 1, 2, 4, 8, 16, 32, and 64 grid size trajectory length of expert demonstrations.

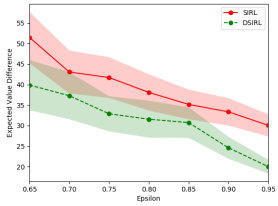

Figure 5: Results under 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, and 0.95 portion size in $\mathcal{C}_{\epsilon}^{E}$.