Estimation of Discount Factor in a Model-Based Inverse Reinforcement Learning Framework

Babatunde H. Giwa[1], **Chi-Guhn Lee**[1]
[1]Department of Mechanical and Industrial Engineering,
Uinversity of Toronto, Canada.
giwa@mie.utoronto.ca, cglee@mie.utoronto.ca

Abstract

We consider the crucial task of estimating an expert's discount factor in Inverse Reinforcement Learning (IRL) to facilitate a better synthesis towards the resulting optimal policy. Existing IRL algorithms have significantly overlooked the vital need to estimate the discount factor, experimental studies and theoretical intuitions show variability of the learnt reward function as the discount factor changes. In this work, we adapt the model-based maximum entropy IRL framework and optimize a utility-based softmax likelihood function via a feature-based gradient update to jointly learn the discount factor and reward. To test our approach, we utilize behavioral data from three Markov decision process (MDP) environments, namely, Grid-World, Mountain-Car Driving and Object-World. Experimental and numerical studies show that our approach is viable for the simultaneous estimation of the discount factor and reward function in IRL.

Introduction

Inverse reinforcement learning (IRL) primarily explains observed behavior in terms of a reward function (Ng and Russell 2000; Ziebart et al. 2008; Neu and Szepesvári 2012; Ramachandran and Amir 2007; Wulfmeier, Ondruska, and Posner 2015). An implicit assumption in IRL is that the discount factor is known; however, experimental studies and theoretical intuitions show variability of learnt rewards as the discount factor changes. Thus, it is crucial to estimate both the discount factor and reward function in IRL. Although a preponderance of IRL literature (Ng and Russell 2000; Ziebart et al. 2008; Boularias, Kober, and Peters 2011; Wulfmeier, Ondruska, and Posner 2015; Neu and Szepesvári 2012; Ramachandran and Amir 2007; Zheng, Liu, and Ni 2014; Vroman 2014; Suay et al. 2016; Choi and Kim 2011) have often assumed the discount factor to be known or arbitrarily chosen, we posit that it is critical to infer the discount factor as part of the learning process in IRL. Besides, IRL is an ill-posed problem, thus, the pioneering mathematical formalism seek to maximize the sum of differences between quality of optimal and second-best actions (Ng and Russell 2000). This would penalize single-step deviations from the optimal policy by making such deviations as costly as possible. However, maximum entropy IRL (Ziebart et al. 2008) employed a probabilistic approach to address ambiguity in IRL.

Typical IRL solution methods can be categorized into model-based and model-free approaches. By model-based, we mean that the IRL algorithm has a known model of the environment, i.e., states transition dynamics (P) while model-free assumes no model of the environment is known a priori. Some model-free approaches (Boularias, Kober, and Peters 2011; Finn, Levine, and Abbeel 2016) heavily rely on the Kullback-Leibler (KL) divergence measure to estimate the partition function which is accurately computed in the model-based approaches. In model-based RL, the discount factor "shapes" the planning horizon which can be short (γ closer to 0) or long (γ closer to 1) depending on the preferences of the decision-maker; a policy applied to a longer horizon might not be applicable in a shorter horizon given the same model of the environment in both horizons. From experimental studies, the use of an arbitrary discount factor in IRL could lead to over- or under-estimation of the value function as well as a different optimal policy when the learnt reward function is optimized in a Markov decision process environment. Thus, it is crucial and non-trivial to estimate the discount factor as part of the IRL problem.

In order to impose some structure on the reward solution space towards managing complexity, a linear approximation in the state features have often been used (Ng and Russell 2000; Ziebart et al. 2008; Neu and Szepesvári 2012; Vroman 2014) and we follow such linear approximation in this work. In this work, an agent (or expert) is one whose reward function (R_A) and discount factor (γ_A) the learner (L) seeks to learn. In other words, we seek to estimate the reward function (R_L) for the learner as well as the discount factor (γ_L) given the behaviour (ξ) of an agent (A).

Motivating Example

Consider a hypothetical scenario wherein an agent in state S1 is making decisions and could get a reward of either 1 in state $S4$ or 10 in state $S5$. This scenario is modelled as a 5-state finite-horizon and deterministic MDP of Figure 1 and we show that the forward and corresponding inverse so-

lutions differ as we change the discount factor as seen in Table 1 using the maximum entropy framework (Ziebart et al. 2008). The actual reward of the agent is shown in the second column of Table 1, based on a discount factor of $\gamma_A = 0.35$, the optimal policy (behavioural data) of the agent is shown in the third column of Table 1. Given this data, the maximum entropy algorithm (Ziebart et al. 2008) employs a discount factor of 0.9 to learn the reward function and the corresponding optimal policy in the fourth and fifth columns, respectively of Table 1. Although based on RL formalism, state $S4$ has a lower reward when compared to state $S5$ and a far-sighted agent would induce a policy that gravitates towards state $S5$ as the terminal state. However, if a myopic human who prefers the reward in state $S4$ (based on desiderata, e.g., shorter path) would induce a policy which is not the same as the far-sighted agent. More so, given the behavioural data, one can better understand the agent's motivation via the discount factor. Seemingly, the results show how critical the discount factor is in IRL and RL in general[1].

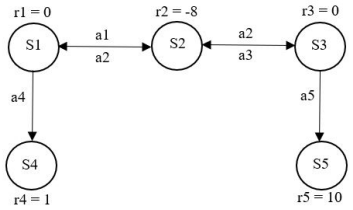


Figure 1: MDP with 5 states.

Table 1: The optimal policies (originally π^*, learnt as $\tilde{\pi}^*$) differed when the discount factor of agent is 0.35 while 0.9 was assumed in the learning; different optimal actions in state $S1$.

	Original ($(\gamma = 0.35)$)		Learnt ($\tilde{\gamma} = 0.9$)	
State	Reward	π^*	Reward	$\tilde{\pi}^*$
S_1	0	a_4	0.1	a_2
S_2	-8	a_3	0.5	a_3
S_3	0	a_5	5.0	a_5
S_4	1	-	-0.1	-
S_5	10	-	9.9	-

Significance of Discount Factor Learning

For a trajectory (τ) with horizon length T, the value function of a policy (π), $V^{\pi}(s_t) = E^{\pi}[\sum_{t=0}^{T} \gamma^t R(s_t)]$. For a given state (s), the discount factor (γ) is not just a scale parameter which "shifts" the reward function $R(s)$, in fact, it deeply affects the resulting optimal policy (π^*). To obtain π^*, we seek the action (a) that maximizes $V^{\pi}(s)$ for each state (s) or the action (a) that maximizes $Q^{\pi}(s, a)$ (Sutton and Barto 2018). It is known that $\gamma < 1$, ensures the return (sum of rewards) converges when $T = \infty$, i.e., infinite horizon (Sutton and Barto 2018). Also, in both finite and infinite horizons, the discount factor[2] plays an important role which is telling us how much a future reward is worth at the current time step (t).

There are two ways to look at the intuition behind the need for discount factor estimation in IRL. Suppose we generate an agent's policy - π_A (empirical data) using a specific discount factor (γ_A), existing IRL literature assume that during the learning π_A was generated by some discount factor (γ_L). However, if $\gamma_A \neq \gamma_L$, the empirical data (π_A) would not necessarily be the same for both discount factors. Thus, computational accuracy of learning is affected.

The other perspective is that IRL is originally concerned with explaining human behavior in terms of a reward function (Ng and Russell 2000; Ziebart et al. 2008). It is known that decisions made by humans are implicitly determined by the choice of discount factor (Knox and Stone 2012; Klapproth 2008). For instance, in an MDP context, a human decision maker who is considered myopic would utilize an optimal policy based on a small discount factor say $\gamma = 0.25$, however, RL community usually set $\gamma \geq 0.9$ to extract an optimal policy. Thus, given a behavioral data, it becomes crucial to infer the discount factor associated with such data to enhance a "better" explanation. Besides, in economics literature (Chabris, Laibson, and Schuldt 2010; Benzion, Rapoport, and Yagil 1989), the role of discounting has been investigated. Benzion, Rapoport, and Yagil (1989) inferred the discount rate in a qualitative manner given some behavioral data, however, the 'reward function' was assumed to be known which contrasts the IRL framework. To the best of our knowledge, this is the first work which attempts to estimate both the reward and discount factor in the IRL framework.

The main contributions of this work are: (i) Estimation of discount factor and reward in IRL in a model-based manner, and (ii) Identification of additional source of ambiguity in IRL given the softmax likelihood function.

Markov Decision Process

Markov decision process (MDP) comprises a tuple - $\{S, A, P, R, \gamma\}$; wherein S : finite set of states, A : finite set of actions, P : states transition probability matrix, R: reward function and γ : discount factor. A policy is a function $\pi : S \rightarrow A$ (or a distribution $\pi(a|s)$). An episode (τ) is a sequence of state-action pairs with horizon length (T), i.e., $\tau = \{(s_0, a_0), (s_1, a_1), ..., (s_T, a_T)\}$. The utility (or return) of a trajectory $(U(\tau)) = \sum_{t=0}^{T} \gamma^t R(s_t, a_t) \ \forall \ s_t, a_t \in \tau$. Given a Markov decision process (MDP), reinforcement learning (RL) (Sutton and Barto 2018) is employed to obtain an optimal policy (π^*) which maximizes the expected sum of discounted rewards, i.e., expected utility. Hence, given π^* would imply that $V^{\pi^*}(s) = V^*(s) = sup_{\pi} V^{\pi}(s)$ for all $s \in S$. Similarly, the optimal action-value function is defined as $Q^{\pi^*}(s, a) = Q^*(s, a) = sup_{\pi} Q^{\pi}(s, a)$.

[1] See Appendix for another example.

[2] $\gamma \in (0, 1)$. We do not consider the boundary values of 0 and 1 as discounting. If $\gamma = 0$, this transforms the RL problem to a supervised learning one. On the other hand, if $\gamma = 1$, it is simply a total sum of rewards.

Inverse Reinforcement Learning

Inverse reinforcement learning (IRL) originally seeks to determine the reward function for an underlying Markov decision process (Ng and Russell 2000; Ziebart et al. 2008) and provisions a sub-result for apprenticeship learning (Abbeel and Ng 2004; Neu and Szepesvári 2012). Essentially, an IRL algorithm has as input an MDP without rewards (R), i.e.,$\{\text{MDP}\backslash\text{R}\}$ and demonstrations (or trajectories) as well as (often) the features or basis function – ϕ. The reward function is often linearly parameterized with the basis function – ϕ (Ng and Russell 2000; Ziebart et al. 2008; Vroman 2014). If the reward function is parameterized by θ (θ is also known as the reward weights):

$$R_\theta(s) = \theta^\intercal \phi(s) \tag{1}$$

where ϕ is the reward basis, then the IRL problem becomes finding θ^* given the demonstration data. Inverse RL (IRL) is closely related with the general inverse optimization literature (Aswani, Shen, and Siddiq 2018; Esfahani et al. 2018) in which given a parametric objective function and data, determine what parameter realizations closely match the given data by minimizing an empirical loss function.

Maximum Entropy IRL

IRL, as typified with inverse optimization problems, is an ill-posed problem (it does not satisfy the Hadamard's well-posed problem definition). In the past two decades, there is a known ambiguity problem in IRL wherein (i) multiple reward functions can satisfy the same optimal policy, and (ii) multiple policies can give rise to the same set of demonstration samples. For instance, a fundamental characterization of optimality in the pioneering mathematical IRL framework (Ng and Russell 2000) is $(P^{a*} - P^a)(I - \gamma P^{a*})^{-1} R \geq 0$; this inequality makes apparent the degenerate case $R = 0$ as being optimal for any demonstration set. To address this ambiguity in IRL, several authors have proposed mitigation approaches including use of regularization and probabilistic solution methods. A probabilistic approach is maximum entropy IRL (Ziebart et al. 2008) based on the principle of maximum entropy to match distributions over behavior with data-driven constraints. The feature expectation constraints in the maximum entropy framework (Ziebart et al. 2008) propels the resulting distribution which increases conformity of the maximum likelihood of the trajectories.

Problem definition

Given a finite-horizon MDP without discount factor (γ) and reward (R), i.e., $\{\text{MDP}\backslash\{R,\gamma\}\}$ and a set of trajectories, $\xi = \{(\tau_1, \tau_2, ..., \tau_{|\xi|})\}$, the question arises as to how we would jointly estimate discount factor and reward. To facilitate a solution, we adapt the maximum entropy IRL framework (Ziebart et al. 2008) by maximizing entropy subjected to discounted feature expectation constraints which transformed to a maximum likelihood estimation of a softmax distribution model.

Softmax Likelihood with Utility Function

For clarity, our approach is different from the use of exponential utility which has been proposed in some RL and IRL literature to capture risk sensitivity of the decision maker (Mihatsch and Neuneier 2002; Singh et al. 2018). Nonetheless, to the best of our knowledge, no existing literature has learnt to maximize the likelihood function as a normalized exponential of utility to learn both the reward and discount factor.

Theoretically, we show that the popular maximum entropy framework (Ziebart et al. 2008) subjected to discounted feature constraints results in the probability of visiting a trajectory (τ) as being proportional to the exponential of the utility of the trajectory ($U(\tau)$). To derive the exponential of utility function, we maximize entropy as in maximum entropy IRL (Ziebart et al. 2008) but subject to discounted feature constraints (we note here that this type of constraint has been utilized for an infinite-horizon IRL problem (Bloem and Bambos 2014)). In view of the feature expectation matching constraint: **Objective**

$$\max \quad -\sum_{\tau\in\xi} p(\tau)\log p(\tau)$$

Constraints (subject to:)

$$\sum_{\tau\in\xi}\sum_{t=0}^{|\tau|}\gamma^t p(\tau)\phi(\tau;t) = \frac{1}{|\xi|}\sum_{\tau\in\xi}\sum_{t=0}^{|\tau|}\gamma^t\phi(\tau;t)$$

$$\sum_{\tau\in\xi} p(\tau) = 1$$

$$p(\tau) \geq 0$$

where $\phi(\tau;t)$ is the feature observed at the t-th transaction along trajectory τ and $p(\tau)$ is the probability of trajectory τ. Following a Lagrangian relaxation and solution of a first-order optimality equation: $p(\tau) = e^{\sum_{t=0}^{|\tau|}\gamma^t\theta^\intercal\phi(\tau;t)+w}$. By axioms of probability: $\sum_{\tau\in\xi} p(\tau) = \sum_{\tau\in\xi} e^w e^{\sum_{t=0}^{|\tau|}\gamma^t\theta^\intercal\phi(\tau;t)} = 1$. Hence, $p(\tau) = \frac{1}{Z}e^{\sum_{t=0}^{|\tau|}\gamma^t\theta^\intercal\phi(\tau;t)}$. Taking out the normalizing constant - Z, we invoke the proportionality relation as follows:

$$p(\tau) \propto e^{\sum_{t=0}^{|\tau|}\gamma^t\theta^\intercal\phi(\tau;t)} = e^{U(\tau)}$$

The probability of visiting a trajectory is directly proportional to exponential of the utility of that episode. Thus, the objective is to find θ^* and γ^* that maximizes the likelihood of demonstration set (ξ).

$$\theta^*, \gamma^* = \arg\max_{\theta,\gamma} \frac{1}{|\xi|}\prod_{\tau\in\xi}\frac{e^{\sum_{\tau\in\xi}\sum_{t=0}^{|\tau|}\gamma^t\theta^\intercal\phi(\tau;t)}}{Z} \tag{2}$$

Take the log of the likelihood function in Equation (2), Equation (3) follows as:

$$l(\theta,\gamma;\xi) = \frac{1}{|\xi|}\sum_{\tau\in\xi}\sum_{t=0}^{|\tau|}\gamma^t\theta^\intercal\phi(\tau;t) - \log Z \tag{3}$$

Explicitly, we define the constraints on the discount factor as $\gamma > 0$ and $\gamma < 1$. With a penalization (λ is a penalty to enforce boundary conditions on discount factor) of the constraints, we obtain the following derivatives with respect to θ and γ, respectively:

$$\nabla_\theta = \frac{1}{|\xi|}\sum_{\tau\in\xi}\sum_{t=0}^{|\tau|}\gamma^t\phi(\tau;t) - \sum_{s\in S}\sum_{t=0}^{|\tau|}\gamma^t P(s_t|\theta,\gamma)\phi(s_t) \tag{4}$$

$$\nabla_\gamma = \frac{1}{|\xi|}\sum_{\tau\in\xi}\sum_{t=0}^{|\tau|}t\gamma^{t-1}\theta^\intercal\phi(\tau;t) - \sum_{s\in\xi}\sum_{t=0}^{|\tau|}t\gamma^{t-1}P(s_t|\theta,\gamma)\theta^\intercal\phi(s_t) + \lambda \tag{5}$$

where $\phi(s)$ is a feature mapped from state s. With a learning rate (α), we define our update rules as follows:

$$\theta \leftarrow \theta + \alpha\nabla_\theta$$
$$\gamma \leftarrow \gamma + \alpha\nabla_\gamma$$

Experiments and Numerical Analysis

We report experimental results for three benchmarks comprising Grid-World, Mountain-Car driving and Object-World to validate our solution approach for the estimation of discount factor in IRL. Behavioral (policy) data were generated using a specific discount factor (γ_A - agent's discount factor). Based on Equations (4) and (5) coupled with the learning update rules; and after a fixed number of iterations, we learn both the discount factor and reward. Data Generation - Given an MDP comprising the true (agent's) reward function (R_A) and discount factor(γ_A), we solve the MDP and obtain the optimal policy (π_A) of the agent. Trajectories ($\tau_i s$) are sampled according to the π_A. We feed the trajectories (data) into our IRL algorithm and subsequently learn both the reward (R_L) and discount factor(γ_L). Given R_L and γ_L, we solve the forward (RL) problem and obtain the learnt policy (π_L). The implementation was done via Python and MATLAB toolkit for IRL (Levine, Popovic, and Koltun 2011).

Grid-World Environment

In a 5×5 Grid-World, five actions correspond to moves in north ($\uparrow$), west($\leftarrow$), south ($\downarrow$), east ($\rightarrow$) and no action (.); with the MDP environment designed such that with probability of 0.7 success, it moves in the intended direction. As stated in Equation 1, the reward is linear in the features. The initial state in each trajectory is randomly drawn from the possible states and the episode is completed at horizon $|\tau|$. To generate behavioral data (ξ), each trajectory or episode (τ) is sampled from the optimal policy (π^*) with discount factor of the agent (γ_A); where $\xi = \{\tau^1, \tau^2, ..., \tau^N\}$ and $\tau^i = \{(s_1, a_1)^i, (s_2, a_2)^i, ..., (s_T, a_T)^i\}$. The reward weights ($\theta$) and discount factor ($\gamma$) are randomly initialized between 0 and 1.

Results For brevity, we have reported very few experimental results for both sparse and dense reward structures. Nonetheless, experimental results for other trials (both discount factor and reward learning) exhibit similar observations for the respective reward structures. Furthermore, the value of λ only enforces boundary conditions, it has no impact if the discount factor is not close to the open boundaries of 0 and 1 based on our extensive experiments for λ in the range -5 to 15. For accurate inference of discount factor: the gradient information of Equation (5) used to update the discount factor is the difference of discounted functions as well as a further dependence on state visitation frequency whose parameters are θ and γ.

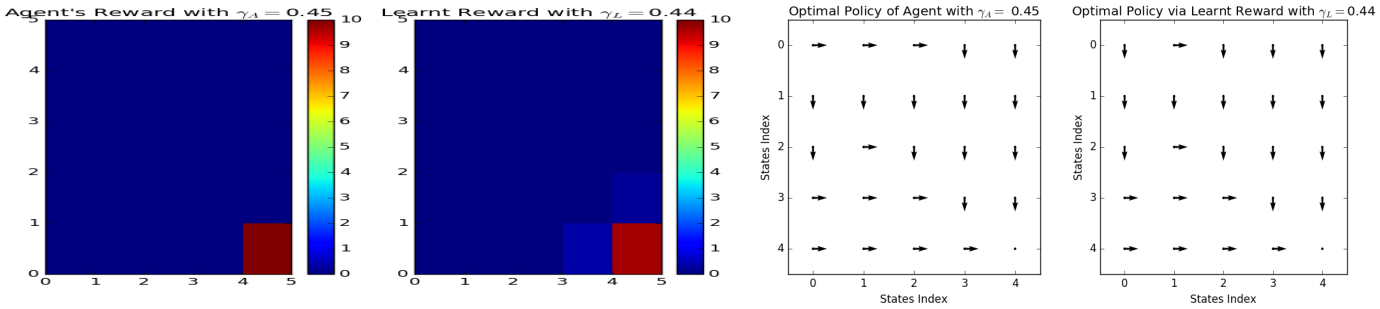
In Figures 2 and 3, the Grid-World states are visualized in a heat map to compare the agent's and learnt rewards. The learnt discount factor (γ_L) is closely approximated with a difference of -0.01. While it may appear our algorithm underestimates the discount factor, this was not the case in all experimental results wherein some values were precisely estimated or overestimated. For the sparse reward case, our algorithm estimates the discount factor within a difference of magnitude 0.02 in general.

Sparse reward structure: From our experimental studies, a very high similarity between learnt policy and agent's policy is also seen with dissimilar optimal actions in states not more than two for the sparse reward structure. In addition, as seen in Figure 2b, the absolute difference of the agent's and learnt optimal value functions were very minimal (between 0 and 1) for all states.

Non-sparse reward structure: For the non-sparse case as seen in Figure 3a, our algorithm precisely estimated the discount factor. However, in general, estimation of the discount factor was within a difference of magnitude 0.03. We note that learning for the non-sparse reward structure is more challenging than the sparse case as we explain in the section on analysis of softmax function. However, as we increase the number of samples, there is an improvement in the joint learning of reward and discount factor when the reward structure is dense as seen in Figure 4 with $\gamma_A = 0.8$. In Figures 4b and 4c, as the sample size increases from 150 to 240, there is an improved accuracy in the estimation of γ_L (0.79 to 0.8) and reward. The comparison of the difference of value function of optimal policies of agent and learner gives insight into the accuracy of learning. For brevity, we report this measure for results obtained in Figure 2a. Although, the optimal actions for the agent and learner differed in two states as seen in Figure 2b, the difference in the value functions of the resulting optimal policies was close to zero as shown in Figure 5a.

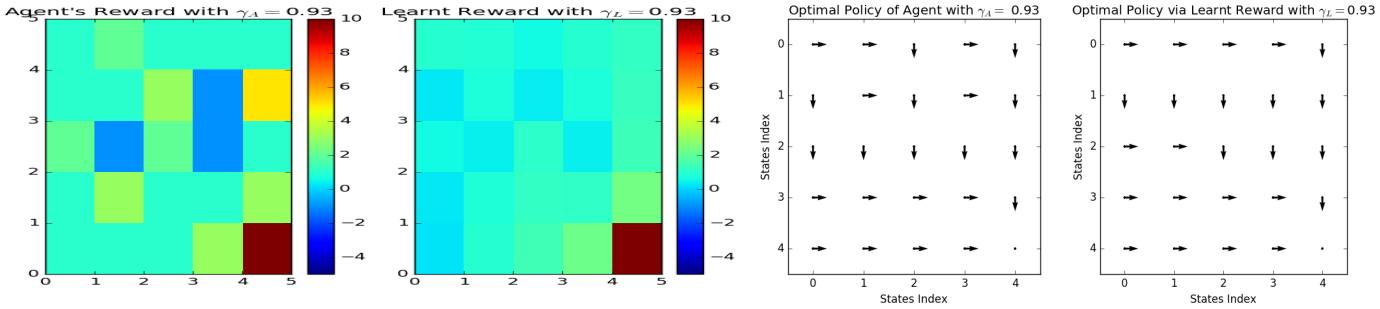
Mountain-Car Driving Environment

The mountain car is a testing domain in RL wherein an under-powered car is tasked with driving up on a steep hill. Since gravity is stronger than the car's engine. The car is situated in a valley and in order to drive up the hill, it has to first move away from the goal (position > 0.5) and climb the slope on its left, and then it moves right, gaining enough momentum to climb the hill on the right on top of which lies the goal. The agent receives a negative reward at every



(a) R_A and R_L (b) π^* for A and L

Figure 2: Actual and learnt rewards with discount factors. Forward solutions show corresponding optimal policies for agent (A) and learner (L).



(a) R_A and R_L (b) π^* for A and L

Figure 3: Actual and learnt rewards & discount factors. Forward solutions show corresponding optimal policies for agent (A) and learner (L).

time step when the goal is not reached. This deterministic (finite) MDP environment comprises continuous state space (Position $\in$ [-1.2,0.6]; Velocity $\in$ [-0.07,0.07] discretized into 400 states (Position into 20; Velocity into 20) with 3 discrete actions being go left, go right or do nothing. The reward weights (θ) are randomly initialized between -1 and 1 while discount factor (γ) is randomly initialized between 0 and 1 .

Results

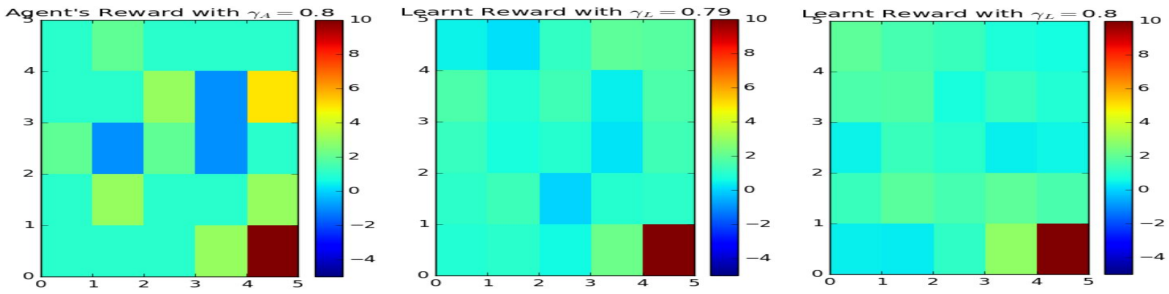
Experiments for Mountain-Car environment are non-trivial to run due to the fact that one has to allow for "adequate" exploration coupled with the granularity of the environment. In this approach, the two continuous state variables are pushed into discrete states by bucketing each continuous variable into multiple discrete states and for stable learning, the learning rate was set to 0.001. Also, a decaying exploration rate facilitated a better convergence for the Q-learning approach used to solve the forward (RL) problem.

Object-World Environment

Object-World (OW) is an MDP environment comprising $|S|$ states in which possible actions include motions in all four directions ($\leftarrow$, $\uparrow$, $\downarrow$, $\rightarrow$) and also remaining in the same position (no action). Two different set of state features are implemented based on colors placed randomly. The reward is positive for cells which are both within distance 3 of color 1 and distance 2 of color 2, negative if only within distance 3 of color 1 and zero otherwise. See Levine, Popovic, and Koltun (2011) and Wulfmeier, Ondruska, and Posner (2015) for a more detailed exposition of the Object-World environment. The reward weights (θ) and discount factor (γ) are randomly initialized between 0 and 1 and subsequently updated with the feature-based gradients. In this OW environment, the reward is represented in three distinct color phases, namely white, grey and black. On the continuum of white-grey-black, white color has the highest reward down to the black color with the lowest reward.

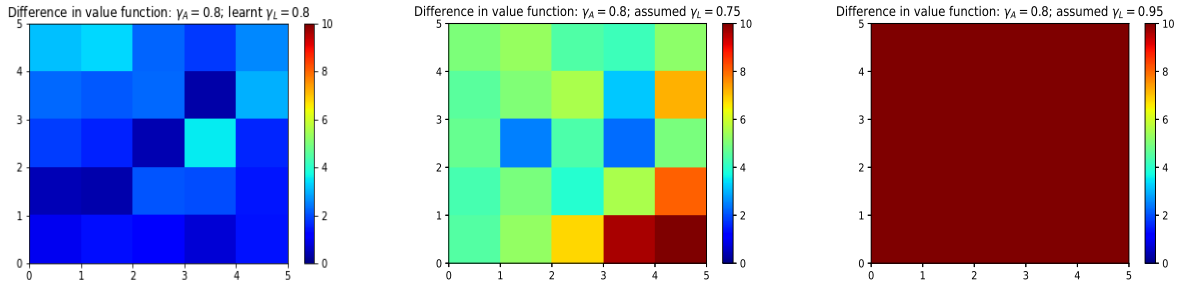
Results

Although, this example exemplifies non-linearity of the rewards in IRL, a linear approximation for our 121-state example shows plausible results as seen in Figures 8. In Figure 8-(a), the agent's discount factor is 0.45 which was accurately inferred whilst the reward was learnt to a good extent. At a glance, it appears the optimal policies are the same, however, the optimal policies of the agent and the learner differed in one state. On the other hand, in Figure 8-(b), the learnt discount factor was overestimated by 0.01 with the reward structure of the agent being quite similar with the learnt



(a) Agent's Reward (b) $|\xi| = 150$ (c) $|\xi| = 240$

Figure 4: Actual and learnt rewards & discount factors as sample size ($|\xi|$) increases.



(a) VFD (b) VFD (baseline) (c) VFD (baseline)

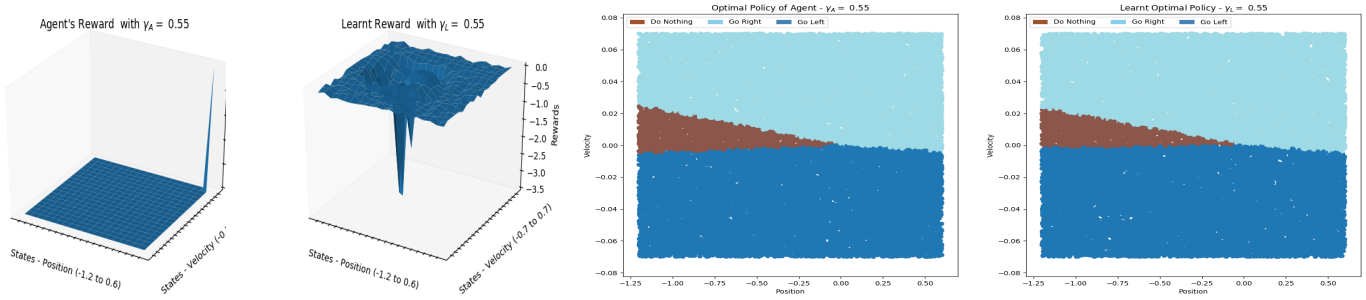
Figure 5: Absolute difference of value functions (VFD) with baseline comparisons.

reward. Interestingly, the optimal policies of the agent and learner are the same. The Object-World complexity poses a challenge for a linear approximation reward learning with discount factor estimation. Nonetheless, from our experimental studies comprising 200 samples with a linear approximation, our algorithm learns the discount factor within an estimation error of ± 0.02. However, the learnt reward was most accurate for areas of non-negative rewards in the underlying MDP.

Analysis of Softmax Function in MLE

Maximum entropy and maximum likelihood are convex duals. By strong duality, the solutions of the primal and dual are the same. However, in general, optimal values are not the same. As seen from the mathematical formalism of our approach, following the use of Lagrangian multipliers and Karush-Kuhn-Tucker (KKT) conditions, we evolved a maximum likelihood (ML) model from a maximum entropy framework in which the parameters (γ, R) to be optimized in an exponential-family, specifically, the softmax (or Boltzmann) distribution. Due to the monotonicity of the log function, we seek to maximize the log-likelihood as stated in Equation (3) to mitigate against computational underflow from product of probabilities (the log transformation of the likelihood allows for the sum of these probabilities). The maximum likelihood estimate is a point estimate and a key property is the general feature that ML estimators achieve optimal accuracy being asymptotically (vis-a-vis sample size) consistent, and achieve the Cramer–Rao lower bound on estimate variability (Lehmann and Casella 2006; Lennart 1999). Furthermore, two properties that guarantee asymptotic convergence of the ML estimator are identification and concavity (Reverdy and Leonard 2015). Recall that the reward is a product of reward weights and known features as in Equation (1), hence given ϕ and θ, R is obtainable. For $0 \leq R \leq R_{max} < \infty : R, R_{max} \in \mathbb{R}^+$ and $0 < \gamma < 1$, the log-likelihood function $l(\theta, \gamma; \xi) = \frac{1}{|\xi|} \sum_{\tau \in \xi} \sum_{t=0}^{|\tau|} \gamma^t \theta^\intercal \phi(\tau_{t:|\tau|}) - \log Z$ is concave given the negative semi-definiteness of the matrix of second derivatives - Hessian ($H(\theta, \gamma)$). When the reward is lower bounded by a negative number and with MDP being a mix of positive and negative rewards $-\infty < R_{min} \leq R \leq R_{max} < \infty$, i.e., $R_{min} \in \mathbb{R}^-$, one can not conclude if the likelihood function is concave or otherwise. However, for a positive shift of the rewards with a scalar $K \in \mathbb{R}^+$, the function is concave.

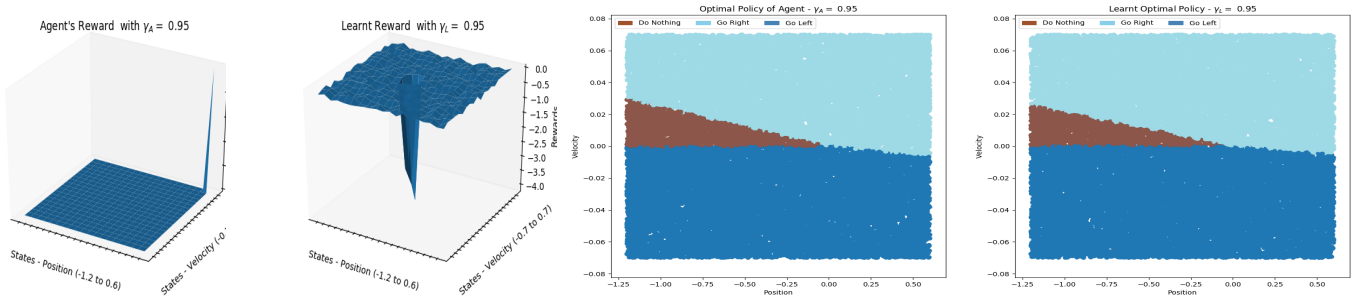
Identification is a major concern in softmax decision-making models (Asadi and Littman 2017; Reverdy and Leonard 2015), such identifiability concern can be data driven (Reverdy and Leonard 2015). A common trick is to fix some of the parameters, in this light, the sparse reward structure is less susceptible to this non-identification property of the softmax operator. Thus, for a sparse reward, the maximum likelihood estimates for both the reward and discount factor would converge to the true parameter values as the sample size increases.



(a) Agent's and Learnt Rewards

(b) π^* for A and L

Figure 6: Actual and learnt rewards with discount factors. Forward solutions show corresponding optimal policies for agent (A) and learner (L) with $\gamma_A = 0.55$ and $\gamma_L = 0.55$.



(a) Agent's and Learnt Rewards

(b) π^* for A and L

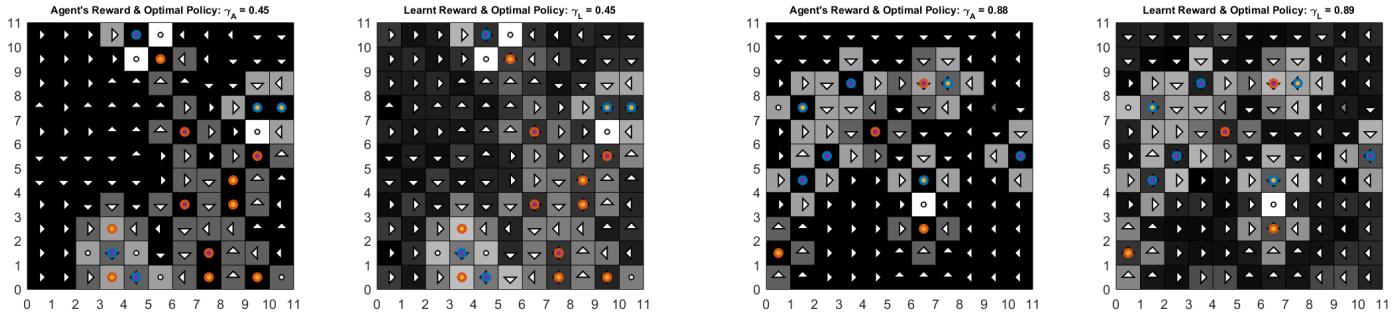
Figure 7: Actual and learnt rewards with discount factors. Forward solutions show corresponding optimal policies for agent (A) and learner (L) with $\gamma_A = 0.95$ and $\gamma_L = 0.95$.

Insight: Another source of multiplicity of reward solutions is introduced in IRL with the softmax distribution model. Although the utility objective in the softmax operator is concave, different combination of parameter values would give rise to the same likelihood, hence, non-identifiable. Thus, there's an additional source of ambiguity to the two aforementioned ones in IRL. To address the ambiguity due to the softmax operator, we can learn a sparse reward and show (Walter and Pronzato 1997) that the softmax operator becomes identifiable. However, the recovered sparse reward would give rise to multiplicity of optimal policies in the forward RL solution.

Related Work

No prior work has simultaneously estimated the discount factor and reward in IRL, more so, discount factor has not been looked at in the general IRL literature. Linear programming approaches like Ng and Russell (2000) in IRL explicitly solve for the reward via a mathematical program and enforce sparsity via L1-regularization. It is worthy to state that the constitutional mathematical framework (Ng and Russell 2000) was very susceptible to changes in discount factor. For instance, as we try to recover sparse solutions via L1-regularization, the learning becomes unstable with changes in discount factor as sparsity and stability "do no not mix" (Xu, Caramanis, and Mannor 2011). To illuminate, given the mathematical program (Ng and Russell 2000) which was transformed into a "proper" linear program (LP) by introduction of auxiliary variables and additional constraints, optimal (learnt) reward was obtained. However, given the LP, as discount factor is varied for the same demonstration data and penalty term for sparsity, the learnt reward varied significantly, hence unstable.

Another class of IRL algorithms (Abbeel and Ng 2004) is based on matching the feature expectations of an expert and learner. However, different optimal policies can yield the same feature expectation. Probabilistic models based on Bayesian and maximum entropy frameworks mitigate against known ambiguities in IRL framework. Choi and Kim (2011) employed a probabilistic approach to resolve the ambiguity in reward solution space via the gradient-based maximum a posteriori (MAP) estimation for recovering the reward function. This was facilitated by the (sub)differentiability of the posterior distribution. A more thoroughly probabilistic and model-based approach (Ziebart et al. 2008) was adopted based on the principle of maximum entropy (Jaynes 1957) in which learning from demonstration (expert trajectories) was done with the objective to find



(a) $\gamma_A = 0.45$ and $\gamma_L = 0.45$

(b) $\gamma_A = 0.88$ and $\gamma_L = 0.89$.

Figure 8: Actual and learnt rewards, discount factors and optimal policies.

a reward that maximizes the likelihood of the demonstrated trajectories. However, an expensive step in estimating the gradient is the computation of state visitation frequencies ($P(s|\theta, \gamma)$) for which an algorithm was devised.

Furthermore, a neural network extension of the maximum entropy framework (Ziebart et al. 2008) is seen in Wulfmeier, Ondruska, and Posner (2015). Interestingly, the maximum likelihood IRL (Vroman 2014) and maximum entropy (Ziebart et al. 2008) algorithms seek a maximum likelihood estimate for the reward function.

Common model-free approaches (Boularias, Kober, and Peters 2011; Finn, Levine, and Abbeel 2016) have surprisingly learnt good optimal policies at the expense of a badly learnt reward function. Another pseudo-IRL approach (Ho and Ermon 2016) under the class of imitation learning bypass the learning of the reward function and directly learns the policy and follows a connection with a model-free framework (Finn, Levine, and Abbeel 2016).

Conclusions and Future Work

Inverse Reinforcement Learning is mainly concerned with explaining (human) behavior. Hence, it is critical to fully comprehend such behavior and the knowledge of the discount factor will propel better synthesis of the learnt reward function. It was shown that the recovered reward function was non-unique with respect to the choice of the discount factor.

After extensive experiments, the normalized exponential of utility based approach propelled the learning of discount factor and reward. The maximum likelihood estimate for γ_L was subject to boundary constraints of the discount factor as we penalized the log-likelihood function of Equation (3) with penalty term λ. Experimental results of our method on selected benchmarks have shown a very promising first step towards the simultaneous estimation of rewards and discount factor.

Despite the encouraging results, it is important to state the following: the sample size affects the learning of the reward and discount factor. Thus, the sample complexity of our approach is subject to further investigation. Also, a suggestion for future work is a characterization of the extent to which parameters are identifiable given the samples, especially for a non-sparse reward structure.

Acknowledgments

Special thanks to Scott Sanner for his reviews on earlier versions of this work.

References

Abbeel, P., and Ng, A. Y. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, 1.

Asadi, K., and Littman, M. L. 2017. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, 243–252.

Aswani, A.; Shen, Z.-J.; and Siddiq, A. 2018. Inverse optimization with noisy data. *Operations Research* 66(3):870–892.

Benzion, U.; Rapoport, A.; and Yagil, J. 1989. Discount rates inferred from decisions: An experimental study. *Management science* 35(3):270–284.

Bloem, M., and Bambos, N. 2014. Infinite time horizon maximum causal entropy inverse reinforcement learning. In *53rd IEEE Conference on Decision and Control*, 4911–4916. IEEE.

Boularias, A.; Kober, J.; and Peters, J. 2011. Relative entropy inverse reinforcement learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 182–189.

Chabris, C. F.; Laibson, D. I.; and Schuldt, J. P. 2010. Intertemporal choice. In *Behavioural and Experimental Economics*. Springer. 168–177.

Choi, J., and Kim, K.-E. 2011. Map inference for bayesian inverse reinforcement learning. In *Advances in Neural Information Processing Systems*, 1989–1997.

Esfahani, P. M.; Shafieezadeh-Abadeh, S.; Hanasusanto, G. A.; and Kuhn, D. 2018. Data-driven inverse optimization with imperfect information. *Mathematical Programming* 167(1):191–234.

Finn, C.; Levine, S.; and Abbeel, P. 2016. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, 49–58.

Ho, J., and Ermon, S. 2016. Generative adversarial imitation learning. In *Advances in neural information processing systems*, 4565–4573.

Jaynes, E. T. 1957. Information theory and statistical mechanics. *Physical review* 106(4):620.

Klapproth, F. 2008. Time and decision making in humans. *Cognitive, Affective, & Behavioral Neuroscience* 8(4):509–524.

Knox, W. B., and Stone, P. 2012. Reinforcement learning from human reward: Discounting in episodic tasks. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, 878–885. IEEE.

Lehmann, E. L., and Casella, G. 2006. *Theory of point estimation*. Springer Science & Business Media.

Lennart, L. 1999. System identification: theory for the user. *PTR Prentice Hall, Upper Saddle River, NJ* 1–14.

Levine, S.; Popovic, Z.; and Koltun, V. 2011. Nonlinear inverse reinforcement learning with gaussian processes. In *Advances in Neural Information Processing Systems*, 19–27.

Mihatsch, O., and Neuneier, R. 2002. Risk-sensitive reinforcement learning. *Machine learning* 49(2-3):267–290.

Neu, G., and Szepesvári, C. 2012. Apprenticeship learning using inverse reinforcement learning and gradient methods. *arXiv preprint arXiv:1206.5264*.

Ng, A. Y., and Russell, S. J. 2000. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, 2.

Ramachandran, D., and Amir, E. 2007. Bayesian inverse reinforcement learning. In *IJCAI*, volume 7, 2586–2591.

Reverdy, P., and Leonard, N. E. 2015. Parameter estimation in softmax decision-making models with linear objective functions. *IEEE Transactions on automation science and engineering* 13(1):54–67.

Singh, S.; Lacotte, J.; Majumdar, A.; and Pavone, M. 2018. Risk-sensitive inverse reinforcement learning via semi-and non-parametric methods. *The International Journal of Robotics Research* 37(13-14):1713–1740.

Suay, H. B.; Brys, T.; Taylor, M. E.; and Chernova, S. 2016. Learning from demonstration for shaping through inverse reinforcement learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 429–437.

Sutton, R. S., and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Vroman, M. C. 2014. *Maximum likelihood inverse reinforcement learning*. Ph.D. Dissertation, Rutgers University-Graduate School-New Brunswick.

Walter, E., and Pronzato, L. 1997. Identification of parametric models. *Communications and control engineering* 8.

Wulfmeier, M.; Ondruska, P.; and Posner, I. 2015. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*.

Xu, H.; Caramanis, C.; and Mannor, S. 2011. Sparse algorithms are not stable: A no-free-lunch theorem. *IEEE transactions on pattern analysis and machine intelligence* 34(1):187–193.

Zheng, J.; Liu, S.; and Ni, L. M. 2014. Robust bayesian inverse reinforcement learning with sparse behavior noise. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Ziebart, B. D.; Maas, A. L.; Bagnell, J. A.; and Dey, A. K. 2008. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, 1433–1438. Chicago, IL, USA.

Appendix

We adopt the following example in Figure 9 from (Sutton and Barto 2018) to show the significance of discount factor and consequently the effect on trajectories sampled from the optimal policy. Suppose an expert acted in the MDP of Figure 9 with discount factor $\gamma_1 = 0.25$ or $\gamma_2 = 0.92$. The rewards (r) are received deterministically after each action. The key decision is to be made at state $S1$, a solution of this MDP would give rise to actions $a2$ and $a3$ as optimal actions in state $S1$ for the expert when $\gamma_1 = 0.2$ and $\gamma_2 = 0.92$, respectively. Based on the optimal policies of the respective discount factors, the following (selected finite) trajectories (τ) would be generated: for $\gamma_1 - \tau = \{(S1, a2), (S2, a1), (S1, a2)\}$; and for $\gamma_2 - \tau = \{(S1, a3), (S3, a1), (S1, a3)\}$. Evidently, expert when $\gamma_1 = 0.2$ would never visit state $S3$, albeit acting optimally with respect to the discount factor. On the other hand, with $\gamma_2 = 0.92$, the expert would never visit state $S2$.

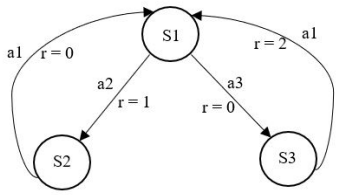


Figure 9: MDP with 3 states represented as nodes. The rewards (r) and actions ($a1, a2, a3$) are shown on the arcs. For instance, action $a2$ in state $S1$ would take an expert to state $S2$ and get a reward, $r = 1$.