Truly Batch Model-Free Inverse Reinforcement Learning about Multiple Intentions

Giorgia Ramponi* **Amarildo Likmeta*†**
Alberto Maria Metelli* **Andrea Tirinzoni*** **Marcello Restelli***
*Politecnico di Milano, Milan, Italy †Università di Bologna, Bologna, Italy

Abstract

We consider Inverse Reinforcement Learning (IRL) about multiple intentions, i.e., the problem of estimating the unknown reward functions optimized by a group of experts that demonstrate optimal behaviors. Most of the existing algorithms either require access to a model of the environment or need to repeatedly compute the optimal policies for the hypothesized rewards. However, these requirements are rarely met in real-world applications, in which interacting with the environment can be expensive or even dangerous. In this paper, we address the IRL about multiple intentions in a fully model-free and batch setting. We first cast the single IRL problem as a constrained likelihood maximization and then we use this formulation to cluster agents based on the likelihood of the assignment. In this way, we can efficiently solve, without interactions with the environment, both the IRL and the clustering problem. Finally, we evaluate the proposed methodology on simulated domains and on a real-world social-network application.

1 Introduction

Inverse Reinforcement Learning (IRL) has emerged as a valuable tool for inferring an expert's reward function from demonstrations (Osa et al., 2018). In the most studied setting, a single expert demonstrates a behavior by providing a set of trajectories of interaction with the environment. The expert's behavior, encoded by its policy, is optimizing an *unknown* reward function. The goal of IRL consists of finding a reward function that makes the expert's behavior optimal (Ng and Russell, 2000). Compared to other imitation learning approaches (Argall et al., 2009; Hussein et al., 2017), which output an imitating policy, such as Behavioral Cloning (BC, Argall et al., 2009), IRL explicitly provides a succinct representation of the expert's *intent*, i.e., a reward function (Sutton and Barto, 1998). For this reason, it provides more general and transferable information than an imitating policy.

In several realistic scenarios, however, there may be multiple agents that possibly pursue distinct interests. In this case, we are facing a more general problem that we call Inverse Reinforcement Learning about Multiple Intentions (MI-IRL, Babes et al., 2011; Choi and Kim, 2012; Almingol and Montesano, 2015). MI-IRL poses challenges and opportunities. When multiple agents are involved, MI-IRL requires to understand whether the agents share the same intent. In other words, we should address an *intent-clustering* problem in which we aim to identify groups of experts that share the same goal. This problem is formulated at a different level w.r.t. to *behavioral-clustering*, as in a Markov Decision Process (MDP, Puterman, 1994) there might exist multiple policies that optimize the same reward. MI-IRL arises in several common daily situations. In a social network, we can easily recognize groups of users that display similar attitudes (e.g., seeking media attention) that can be, possibly, significantly far away from those of other groups of users (e.g., looking at the news, sharing content with friends, etc.). Identifying the intents of users might, for instance, provide valuable information for designing an effective marketing or advertising strategy.

As an immediate benefit, grouping experts that show different behaviors, but share the same intent allows enlarging the set of demonstrations available for the reward recovery process. This has a significant impact on several realistic scenarios, where the only information available is the demonstration dataset and no further interactions with the environment are allowed. We will refer to this scenario as *batch model-*

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

free IRL. This setting is particularly challenging but quite common. Consider, for instance, the problem of inferring the intentions of a group of car drivers, given a set of demonstrations. We cannot perform forward learning (at least in a non-simulated environment) for safety reasons, and, typically, the amount of data available is not enough to perform off-line learning. In the single-intention IRL case, only a few algorithms are able to avoid interaction with the environment (e.g., Klein et al., 2012, 2013; Pirotta and Restelli, 2016; Metelli et al., 2017). These approaches, however, resort to a single-point estimate of the quantities of interest (e.g., feature expectation) and do not take into account the uncertainty due to estimates with finite samples. This limitation has a large impact on the multiple-intention setting, in which the agents might show a significantly different uncertainty, due, for instance, to the different demonstrated policies.

The contribution of this paper is twofold. We first propose a novel *batch model-free IRL algorithm*, named Σ-*Gradient Inverse Reinforcement Learning* (Σ-GIRL), and then we extend it to the multiple-intention setting. Σ-GIRL is derived from Gradient Inverse Reinforcement Learning (GIRL, Pirotta and Restelli, 2016), an IRL algorithm that searches for a reward function that makes the estimated *policy gradient* (Sutton et al., 2000) vanish. Such reward function is a stationary point of the expected return and, under suitable conditions, it makes the policy demonstrated by the expert an optimal policy. Our method, instead, explicitly considers the uncertainty in the gradient estimation process, by casting the IRL problem as a constrained maximum likelihood problem, in which we look for the reward function that maximizes the likelihood of the estimated policy gradients, under the constraint that such reward is a stationary point of the expected return. The resulting objective function accounts for the variance of the gradient and reduces to the GIRL case for a specific choice of the covariance model (Section 4). Then, we embed Σ-GIRL into the multiple-intention framework by proposing a *clustering algorithm* that, by exploiting the likelihood model of Σ-GIRL, groups the experts according to their intentions. The optimization of the multiple-intention objective is performed in an *expectation-maximization* (EM) fashion, in which the (soft) agent-cluster assignments and the reward functions are obtained through an alternating optimization process (Section 5). After having revised the related works (Section 6), we present an experimental evaluation aimed at highlighting the performance of Σ-GIRL, compared with state-of-the-art methods on simulated domains and on a real-world experiment in which we recover and cluster the intents of a group of Twitter users (Section 7). The proof of all the results are reported in Appendix A.

2 Preliminaries

Markov Decision Processes without Reward An MDP without Reward (MDP$\backslash R$) (Puterman, 1994) is defined as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \gamma, \mu)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}_+$ is the transition function, which defines the density $P(s'|s,a)$ between two states $s, s' \in \mathcal{S}$ under action $a \in \mathcal{A}$, $\gamma \in [0,1)$ is the discount factor, and $\mu : \mathcal{S} \to \mathbb{R}_+$ is the initial state distribution. The agent's behavior is described by means of a (possibly stochastic) policy $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$, where $\pi(\cdot|s)$ specifies for each state s a distribution over the action space $\mathcal{A}$. We consider stochastic differentiable policies belonging to a parametric space $\Pi_\Theta = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d\}$. When executing a policy, we define a trajectory as a sequence of states and actions $\tau = (s_0, a_0, \ldots, s_{T-1}, a_{T-1}, s_T)$, where T is the trajectory length.

Inverse Reinforcement Learning The goal IRL consists of recovering the *unknown* reward function optimized by an expert given demonstrations of her behavior. More formally, the expert plays a policy π^E which is (nearly) optimal for some unknown reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and we are given a dataset $D = \{\tau_1, \ldots, \tau_n\}$ of trajectories from π^E. We consider linear reward functions:[1]

$$R_{\boldsymbol{\omega}}(s,a) = \boldsymbol{\omega}^T \boldsymbol{\phi}(s,a), \quad \boldsymbol{\omega} \in \mathbb{R}^q_+, \quad \|\boldsymbol{\omega}\|_1 = 1, \tag{1}$$

with $\boldsymbol{\phi} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^q$ is a limited feature vector function. We denote with $\boldsymbol{\omega}^E$ the parameterization of the reward optimized by the expert. We define the *feature expectations* of a policy π as:

$$\boldsymbol{\psi}^\pi = \mathop{\mathbb{E}}_{\substack{S_0 \sim \mu, \\ A_t \sim \pi(\cdot|S_t), \\ S_{t+1} \sim P(\cdot|S_t, A_t)}} \left[\sum_{t=0}^{+\infty} \gamma^t \boldsymbol{\phi}(S_t, A_t) \right],$$

For a parametric policy $\pi_{\boldsymbol{\theta}} \in \Pi_\Theta$, we abbreviate $\boldsymbol{\psi}(\boldsymbol{\theta}) = \boldsymbol{\psi}^{\pi_{\boldsymbol{\theta}}}$. Finally, we define the *discounted expected return* of a policy $\pi_{\boldsymbol{\theta}}$, that decomposes under the linear reward model as:

$$J(\boldsymbol{\theta}, \boldsymbol{\omega}) = \mathop{\mathbb{E}}_{\substack{S_0 \sim \mu, \\ A_t \sim \pi_{\boldsymbol{\theta}}(\cdot|S_t), \\ S_{t+1} \sim P(\cdot|S_t, A_t)}} \left[\sum_{t=0}^{+\infty} \gamma^t R_{\boldsymbol{\omega}}(S_t, A_t) \right] = \boldsymbol{\omega}^T \boldsymbol{\psi}(\boldsymbol{\theta}).$$

Furthermore, we assume that the expert's policy π^E belongs to the *known* parametric policy space Π_Θ, made of differentiable policies, i.e., there exists $\boldsymbol{\theta}^E \in \Theta$ such that $\pi^E = \pi_{\boldsymbol{\theta}^E}$ almost surely. However, we will not assume to know the expert parameterization $\boldsymbol{\theta}^E$.

Multiple-Intent IRL In a multiple intent setting (Babes et al., 2011) there is a set of experts

[1] The constraints force the weight vector $\boldsymbol{\omega}$ to belong to the q-dimensional simplex and allow solving a form of ambiguity of the reward function (Pirotta and Restelli, 2016).

$\mathbf{E} = (E_1, \dots, E_m)$ and a set of (unknown) reward functions $\mathbf{R} = (R_{\boldsymbol{\omega}_1}, \dots, R_{\boldsymbol{\omega}_k})$, with $k \leq m$. Each expert $E_i \in \mathbf{E}$ demonstrates a policy $\pi^{E_i} \in \Pi_\Theta$ (i.e., there exists $\boldsymbol{\theta}^{E_i} \in \Theta$ such that $\pi^{E_i} = \pi_{\boldsymbol{\theta}^{E_i}}$ almost surely) which optimizes a reward $R_{\boldsymbol{\omega}_{r_i}}$, with $r_i \in \{1, \dots, k\}$. Given a dataset $\mathbf{D} = (D_1, \dots, D_m)$, where each $D_i = \{\tau_1, \dots, \tau_{n_i}\}$ is the set of n_i trajectories demonstrated by E_i, the goal is to recover the reward functions optimized by each of the m experts. We assume to know the identity of the expert generating each trajectory and the number of rewards functions k, but not the policy parameters $\boldsymbol{\theta}^{E_i}$.

3 Gradient-based approaches to IRL

In this section, we revise the main elements that are necessary to understand the approaches to IRL based on the policy gradient (e.g., Pirotta and Restelli, 2016; Metelli et al., 2017; Tateo et al., 2017). We start by stating the policy gradient (Sutton et al., 2000; Peters and Schaal, 2008) and its decomposition under a linear reward model $R_{\boldsymbol{\omega}}(s,a) = \boldsymbol{\omega}^T \boldsymbol{\phi}(s,a)$:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}, \boldsymbol{\omega}) = \mathop{\mathbb{E}}_{\substack{S_0 \sim \mu, \\ A_t \sim \pi_{\boldsymbol{\theta}}(\cdot|S_t), \\ S_{t+1} \sim P(\cdot|S_t, A_t)}} \left[\sum_{t=0}^{+\infty} \gamma^t R_{\boldsymbol{\omega}}(S_t, A_t) \times \sum_{l=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_l|S_l) \right] = \nabla_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}) \boldsymbol{\omega},$$

where $\nabla_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}) = (\nabla_{\boldsymbol{\theta}} \psi_1(\boldsymbol{\theta}) | \dots | \nabla_{\boldsymbol{\theta}} \psi_q(\boldsymbol{\theta})) \in \mathbb{R}^{d \times q}$ is the Jacobian matrix. When the expert's policy $\pi_{\boldsymbol{\theta}^E} \in \Pi_\Theta$ optimizes the reward function $R_{\boldsymbol{\omega}^E}$, $\boldsymbol{\theta}^E$ is a *stationary point* of the expected return $J(\boldsymbol{\theta}, \boldsymbol{\omega}^E)$ and thus the gradient of $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}^E, \boldsymbol{\omega}^E) = \nabla_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}^E) \boldsymbol{\omega}^E$ must vanish (first-order necessary conditions for optimality (Nocedal and Wright, 2006)). In other words, the weight $\boldsymbol{\omega}^E$, associated to the reward function optimized by the expert, belongs to the *null space* of the Jacobian $\nabla_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}^E)$.

Gradient Inverse Reinforcement Learning GIRL (Pirotta and Restelli, 2016) leverages on this observation to recover the expert's weight vector $\boldsymbol{\omega}^E$. In practice, however, we do not have access to the true Jacobian matrix $\nabla_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}^E)$, but to a finite-sample estimate $\widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}^E)$, obtained starting from the trajectories in $D = \{\tau_1, ..., \tau_n\}$.[2] This estimation might result of full rank due to estimation errors, preventing the search of the corresponding null space. For this reason, GIRL, instead of looking for the null space of $\widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}^E)$, seeks for the direction of minimum growth by minimizing the L^p-norm of the gradient estimate, leading to the optimization problem:

$$\min_{\substack{\boldsymbol{\omega} \in \mathbb{R}^q_+ \\ \|\boldsymbol{\omega}\|_1 = 1}} \left\| \widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}^E) \boldsymbol{\omega} \right\|_p^p. \tag{2}$$

This objective has the desirable property of being convex for any choice of $p \geq 1$.

Estimating the expert's policy The computation of the Jacobian $\nabla_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}^E)$ requires the knowledge of the functional form of the expert's policy, in order to compute the score $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)$. Since we assume that the expert's policy belongs to a parametric policy space Π_Θ made of differentiable policies, as explained in Pirotta and Restelli (2016), we can recover an approximation of the expert's parameters $\boldsymbol{\theta}^E$ through behavioral cloning, exploiting the trajectories in D. If we resort to a maximum-likelihood estimation, we solve the following optimization problem, obtaining an estimate $\widehat{\boldsymbol{\theta}}^E$ of $\boldsymbol{\theta}^E$:

$$\max_{\boldsymbol{\theta} \in \Theta} \frac{1}{n} \sum_{l=1}^{n} \sum_{t=0}^{T-1} \log \pi_{\boldsymbol{\theta}}(a_{l,t}|s_{l,t}). \tag{3}$$

It is known that the maximum likelihood estimation is consistent under mild regularity conditions on the policy space Π_Θ and assuming the identifiability property (Casella and Berger, 2002). Some finite-sample guarantees on the concentration of the distance $\|\widehat{\boldsymbol{\theta}}^E - \boldsymbol{\theta}^E\|_p$ were also derived under stronger assumptions (e.g., Spokoiny et al., 2012).

4 Σ-Gradient Inverse Reinforcement Learning

In this section, we introduce a novel batch model-free IRL algorithm, named Σ-*Gradient Inverse Reinforcement Learning* (Σ-GIRL), which extends GIRL (Pirotta and Restelli, 2016) to account for the uncertainties injected by the Jacobian estimation process. Compared to GIRL, we take a different perspective to address the problem of the inaccurate Jacobian estimation. Instead of looking for the direction of minimum growth, we allow the components of $\widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta})$ to move in order to generate a new Jacobian $\mathbf{M} \in \mathbb{R}^{d \times q}$ that has a non-empty null space. Intuitively, the more a component $\widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}_{ij}(\boldsymbol{\theta})$ is uncertain, the more we are allowed to move it. This notion can be formalized as a constrained maximum-likelihood problem. We consider a (matrix) Gaussian distribution (Gupta and Nagar, 2018) to model the Jacobian estimate: $\widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta}) \sim \mathcal{N}\left(\mathbf{M}, \frac{1}{n} \boldsymbol{\Sigma}\right)$, where $\boldsymbol{\Sigma} \in \mathbb{R}^{dq \times dq}$ is the covariance matrix. This choice is justified by the Central Limit Theorem as the estimated Jacobian is a mean of n samples, thus, its distribution approaches a normal as n grows to infinity (Casella and Berger,

[2] An unbiased sample-based estimate of $\nabla_{\boldsymbol{\theta}} \boldsymbol{\psi}(\boldsymbol{\theta})$ can be obtained with the standard policy gradient estimators, such as REINFORCE (Williams, 1992) or G(PO)MDP(Baxter and Bartlett, 2001).

2002). The corresponding likelihood function, given the trajectory set $D = \{\tau_1, \dots \tau_n\}$ is:

$$\mathcal{L}_{\boldsymbol{\Sigma}}(\mathbf{M}|D) = \frac{\sqrt{n}}{\sqrt{(2\pi)^{dq}|\boldsymbol{\Sigma}|}} e^{-\frac{n}{2}\left\|\mathrm{vec}\left(\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})-\mathbf{M}\right)\right\|^2_{\boldsymbol{\Sigma}^{-1}}}, \quad (4)$$

where, for a matrix $\mathbf{A}$, $\mathrm{vec}(\mathbf{A})$ denotes the vectorization of $\mathbf{A}$, i.e., the vector obtained by stacking the columns of $\mathbf{A}$. We now formulate the IRL problem as the problem of finding the weight vector $\boldsymbol{\omega}$ and the new Jacobian $\mathbf{M}$ that, jointly, maximize the likelihood, while $\boldsymbol{\omega}$ belongs to the null space of $\mathbf{M}$:

$$\min_{\substack{\boldsymbol{\omega}\in\mathbb{R}^q_+ \\ \|\boldsymbol{\omega}\|_1=1}} \min_{\substack{\mathbf{M}\in\mathbb{R}^{d\times q} \\ \mathbf{M}\boldsymbol{\omega}=\mathbf{0}}} \left\|\mathrm{vec}\left(\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta}) - \mathbf{M}\right)\right\|^2_{\boldsymbol{\Sigma}^{-1}}. \quad (5)$$

This optimization problem can be simplified since the inner maximization can be solved in a closed form, by leveraging on a weighted low-rank approximation of matrix $\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$ (Manton et al., 2003).

Theorem 4.1. *If $\boldsymbol{\Sigma}$ is positive definite, the optimization problem* (5) *can be restated as:*

$$\min_{\substack{\boldsymbol{\omega}\in\mathbb{R}^q_+ \\ \|\boldsymbol{\omega}\|_1=1}} \left\|\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})\boldsymbol{\omega}\right\|^2_{\left[(\boldsymbol{\omega}\otimes\mathbf{I}_d)^T\boldsymbol{\Sigma}(\boldsymbol{\omega}\otimes\mathbf{I}_d)\right]^{-1}}, \quad (6)$$

where $\otimes$ denotes the Kronecker product and $\mathbf{I}_d$ is the identity matrix of order d. Furthermore, the approximating Jacobian $\mathbf{M}(\boldsymbol{\omega})$ is given by:

$$\mathrm{vec}\left(\mathbf{M}(\boldsymbol{\omega})\right) = \Big\{\mathbf{I}_{dq} - \boldsymbol{\Sigma}(\boldsymbol{\omega}\otimes\mathbf{I}_d)\left[(\boldsymbol{\omega}\otimes\mathbf{I}_d)^T\boldsymbol{\Sigma}(\boldsymbol{\omega}\otimes\mathbf{I}_d)\right]^{-1} \times (\boldsymbol{\omega}\otimes\mathbf{I}_d)^T\Big\}\mathrm{vec}\left(\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})\right).$$

Unfortunately, the objective function (5) is non-convex for a generic choice of $\boldsymbol{\Sigma}$. However, for specific choices of $\boldsymbol{\Sigma}$ we are able to prove the convexity and recover the objective function optimized by GIRL.

Corollary 4.1. *Let $\mathbf{Q} \in \mathbb{R}^{d\times d}$ be a positive definite matrix and let $\mathbb{1}_q$ denote the q-dimensional vector of all ones. If $\boldsymbol{\Sigma} = \mathbb{1}_q\mathbb{1}_q^T \otimes \mathbf{Q}$, then objective function* (6) *is convex. Furthermore, if $\mathbf{Q} = \mathbf{I}_d$, then the objective function* (6) *is equivalent to* (2) *with $p = 2$.*

We can approximate a generic matrix $\boldsymbol{\Sigma}$ as a matrix of the form $\mathbb{1}_q\mathbb{1}_q^T\otimes\mathbf{Q}$, getting a closed form for $\mathbf{Q}$. In such case, the gap in the objective function can be upper bounded by $\frac{2dq}{s_{\min}(\boldsymbol{\Sigma})^2}\left\|\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})\right\|^2_F\left\|\boldsymbol{\Sigma} - \mathbb{1}_q\mathbb{1}_q^T \otimes \mathbf{Q}\right\|_F$. More details are given in Appendix B. When we do not have access to the true covariance matrix $\boldsymbol{\Sigma}$, we can approximate it with the empirical covariance $\widehat{\boldsymbol{\Sigma}}$.[3]

In the following, we are going to denote with $p(D|\boldsymbol{\omega}) = \mathcal{L}_{\boldsymbol{\Sigma}}(\mathbf{M}(\boldsymbol{\omega})|D)$ the value of the objective as a function of the weight vector attained by the optimal mean matrix $\mathbf{M}(\boldsymbol{\omega})$. As intuition suggests, this quantity can be interpreted as the likelihood of the dataset D, given a weight vector $\boldsymbol{\omega}$. We will employ it in the clustering procedure (Section 5).

[3]The empirical covariance matrix $\widehat{\boldsymbol{\Sigma}}$ might be singular when $dq \gg n$. In such cases, we resort to standard corrections to enforce well-conditioning (Ledoit and Wolf, 2004).

4.1 Theoretical Analysis

In this section, we provide a finite-sample analysis of Σ-GIRL, under the assumption that $\boldsymbol{\Sigma}$ is the true covariance matrix of the distribution having generated the estimated Jacobian $\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$. We consider the case in which the weight vector $\boldsymbol{\omega}^E$ is unique under the simplex constraint (Equation (1)), to avoid multiple solutions. Similarly to what was done in Pirotta and Restelli (2016), we can evaluate the norm of the difference between the expert's weights $\boldsymbol{\omega}^E$ and the recovered ones $\widehat{\boldsymbol{\omega}}$. The following result provides a finite-sample bound for this quantity.

Theorem 4.2. *Let $\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$ be an unbiased estimate of the Jacobian $\nabla_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$ obtained with the trajectories $D = \{\tau_1, \dots, \tau_n\}$. Let $\frac{1}{n}\boldsymbol{\Sigma} = \mathbb{C}\mathrm{ov}[\mathrm{vec}(\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta}))]$ be the true covariance matrix of the estimated Jacobian. Let $\widehat{\boldsymbol{\omega}}$ be the weight vector recovered by Σ-GIRL run with covariance matrix $\boldsymbol{\Sigma}$ and $\boldsymbol{\omega}^E$ be the expert's weight vector. If $\nabla_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$ and $\mathbf{M}(\widehat{\boldsymbol{\omega}})$ have rank $q-1$ and $s_{q-1}(\nabla_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})) = s > 0$, where $s_{q-1}(\cdot)$ denotes the $(q-1)$-th singular value, then it holds that:*

$$\mathbb{E}\left[\left\|\widehat{\boldsymbol{\omega}} - \boldsymbol{\omega}^E\right\|_2\right] \leq \sqrt{\frac{16dq\left\|\boldsymbol{\Sigma}\right\|_2}{s^2 n}},$$

where the expectation is taken w.r.t. the randomness of the trajectories in D used to compute $\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$.

Our result extends Theorem 13.2 of Pirotta (2016) in a few aspects. The result of Pirotta (2016) is clearly applicable to Σ-GIRL, as it assumes that the estimated Jacobian $\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$ has already rank $q-1$. In such case, GIRL and Σ-GIRL behave in the same way as $\mathbf{M}(\widehat{\boldsymbol{\omega}}) = \widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$. However, Theorem 4.2 is more general and applies for Σ-GIRL even for a full-rank estimated Jacobian $\widehat{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\psi}(\boldsymbol{\theta})$. Furthermore, although Σ-GIRL is presented considering a Gaussian likelihood model, Theorem 4.2 makes no assumption on the distribution of the Jacobian, but just requires that $\boldsymbol{\Sigma}$ is the true covariance matrix.

5 Multiple-Intention Σ-GIRL

In this section, we consider the problem of Multiple-Intention IRL (MI-IRL), using Σ-GIRL as a building block for solving the single-intention IRL problem. A naïve solution would be to solve m independent IRL problems, one for each expert agent E_i. However, since we typically have $k \ll m$ (i.e., fewer intentions than experts), this solution would be highly sub-optimal, especially in situations where few demonstrations are available per agent. In such cases, it is much wiser

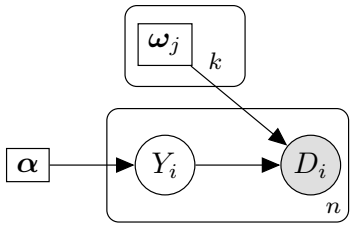

Figure 1: Plate notation of the probabilistic model employed for the clustering procedure.

to cluster the trajectories during the IRL step. To this end, we adopt an *expectation-maximization* (EM) approach (Dempster et al., 1977) to find the parameters $\boldsymbol{\omega}_j$, together with the agent-cluster assignments that maximize the overall likelihood. We introduce the hidden random variable $Y_i \in \{1, \dots, k\}$ that, for each expert E_i with $i \in \{1, \dots, m\}$, indicates to which cluster E_i is assigned. In other words, $Y_i = j$ means that agent E_i is optimizing the reward $R_{\boldsymbol{\omega}_j}$. For these random variables we assume a prior distribution $\alpha_j = p(Y_i = j)$, independent from i, where $\alpha_j \geq 0$ and $\sum_{j=1}^{k} \alpha_j = 1$. We will denote with $\mathbf{Y} = (Y_1, \dots Y_m)$ the concatenation of all the Y_is. The collection of parameters we are going to optimize on is given by the concatenation of the weight vectors $\boldsymbol{\omega}_j$ and the prior probabilities α_j, i.e., $\Omega = (\boldsymbol{\omega}_1, \dots, \boldsymbol{\omega}_k, \alpha_1, \dots, \alpha_k)$. Figure 1 reports the probabilistic model of the clustering procedure.

The crucial observation, for applying EM, is that we can compute the likelihood of a dataset D_i, once we know the cluster assignment of agent E_i, i.e., Y_i:

$$p(D_i | Y_i = j; \Omega) = p(D_i | \boldsymbol{\omega}_j), \tag{7}$$

where the latter is defined in Section 4 (Equation (4)). Exploiting the independence of the datasets $\mathbf{D} = (D_1, \dots, D_m)$ and recalling that $p(D_i, Y_i | \Omega) = p(D_i | Y_i; \Omega) p(Y_i | \Omega)$, we can define the likelihood $\mathcal{L}(\Omega | \mathbf{D}, \mathbf{Y}) = p(\mathbf{D}, \mathbf{Y} | \Omega)$ of all the data as:

$$\mathcal{L}(\Omega | \mathbf{D}, \mathbf{Y}) = \prod_{i=1}^{m} p(D_i, Y_i | \Omega) = \prod_{i=1}^{m} \alpha_{Y_i} p(D_i | \boldsymbol{\omega}_{Y_i}).$$

According to Bilmes et al. (1998), in the expectation step (E-step), we compute the probability of the assignment Y_i, conditioned by the data D_i, in terms of the old parameters Ω^{old}, i.e., $z_{ij} = p(Y_i = j | D_i; \Omega^{\text{old}})$. We derive z_{ij} using Bayes theorem:

$$z_{ij} = \frac{p(D_i | Y_i = j; \Omega^{\text{old}}) p(Y_i = j | \Omega^{\text{old}})}{p(D_i | \Omega^{\text{old}})} = \frac{\alpha_j^{\text{old}} p(D_i | \boldsymbol{\omega}_j^{\text{old}})}{\sum_{h=1}^{k} \alpha_h^{\text{old}} p(D_i | \boldsymbol{\omega}_h^{\text{old}})}.$$

In the maximization step (M-step), instead, we look for the new parameters Ω that maximize the expectation of the log-likelihood $\log \mathcal{L}(\Omega | \mathbf{D}, \mathbf{Y})$ under the previously found distribution over the assignments Y_i:

$$Q(\Omega, \Omega^{\text{old}}) = \mathop{\mathbb{E}}_{\mathbf{Y} \sim p(\cdot | \mathbf{D}; \Omega^{\text{old}})} [\log \mathcal{L}(\Omega | \mathbf{D}, \mathbf{Y})] = \sum_{j=1}^{k} \sum_{i=1}^{m} z_{ij} \log \alpha_j + \sum_{j=1}^{k} \sum_{i=1}^{m} z_{ij} \log p(D_i | \boldsymbol{\omega}_j).$$

The derivation of $Q(\Omega, \Omega^{\text{old}})$ is reported in Appendix A.3. Thus, the EM algorithm keeps alternating the E-step by computing the probabilities z_{ij} and the M-step by computing the new parametrization Ω and α optimizing $Q(\Omega, \Omega^{\text{old}})$. Algorithm 1 reports an overview of Multiple-Intention Σ-GIRL.[4] It is worth noting that for the computation of the new Ω it is required the solution of k IRL problems. Indeed, for a fixed $j \in \{1, \dots, k\}$, denoting with $\widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}_i(\boldsymbol{\theta})$ the estimated Jacobian of agent E_i, the objective function to be optimized is given by:

$$\min_{\substack{\boldsymbol{\omega}_j \in \mathbb{R}_+^q \\ \|\boldsymbol{\omega}_j\|_1 = 1}} \sum_{i=1}^{m} z_{ij} n_i \left\| \widehat{\nabla}_{\boldsymbol{\theta}} \boldsymbol{\psi}_i(\boldsymbol{\theta}) \boldsymbol{\omega}_j \right\|^2_{[(\boldsymbol{\omega}_j \otimes \mathbf{I}_d) \boldsymbol{\Sigma}_i (\boldsymbol{\omega}_j \otimes \mathbf{I}_d)^T]^{-1}}.$$

Algorithm 1 Multiple-Intention Σ-GIRL

input: datasets $\mathbf{D} = (D_1, \dots, D_m)$, number of clusters k, number of iterations N_{ite}
output: optimal parameters $\Omega = (\boldsymbol{\omega}_1, \dots, \boldsymbol{\omega}_k, \alpha_1, \dots, \alpha_k)$
  Initialize Ω^0 randomly
  for $t = 1, \dots, N_{\text{ite}}$ **do**
    E-step: Compute $z_{ij} = \frac{\alpha_j^{t-1} p(D_i | \boldsymbol{\omega}_j^{t-1})}{\sum_{h=1}^{k} \alpha_h^{t-1} p(D_i | \boldsymbol{\omega}_h^{t-1})}$
    M-step: Optimize $\Omega^t \in \arg\max_{\boldsymbol{\omega}_j, \alpha_j} Q(\Omega, \Omega^{t-1})$
      $= \sum_{j=1}^{k} \sum_{i=1}^{m} z_{ij} (\log \alpha_j + \log p(D_i | \boldsymbol{\omega}_j))$
  end for
  return $\Omega^{N_{\text{ite}}}$

6 Related Works

There has been a growing interest in making IRL algorithms scale over real-world continuous domains, where only few or no environment interactions are allowed. Boularias et al. (2011) proposed a model-free variant of MaxEnt IRL (Ziebart et al., 2008), named Relative Entropy IRL (REIRL). Although REIRL avoids solving the MDP, it requires a dataset collected under an explorative policy. A model-free variant of the Maximum-Likelihood IRL (MLIRL, Babes et al., 2011) was recently proposed by Jain et al. (2019). Although the approach only requires expert demonstrations, it repeatedly needs to solve batch RL problems to compute (approximately) optimal Q-functions. However, none of the methods presented above fully address the IRL problem in a truly batch model-free fashion. Instead, Klein et al. (2012) and Klein et al. (2013) reduced IRL to a structured classification problem which, similarly to GIRL (Pirotta and Restelli,

[4] The computational cost is given in Appendix C.

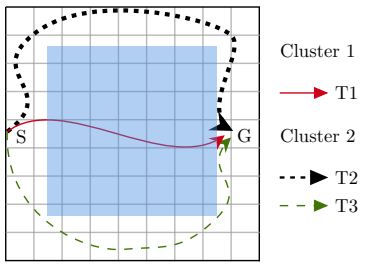

Figure 2: Gridworld with puddles showing the start state (S), the goal state (G), puddles (light blue) and three sample trajectories: T1 (red) from the first cluster, T2 (black) and T3 (green) from the second cluster.

2016) and its extensions (e.g., Metelli et al., 2017; Tateo et al., 2017), can be solved efficiently using only the observed trajectories and with no further interaction with the environment. These algorithms require a good estimate of the expert's feature expectations for each action, even those that have not been demonstrated. Σ-GIRL mitigates this issue, by considering an explicit model of the uncertainty $\boldsymbol{\Sigma}$ of the gradient estimate, while importing all the advantages of GIRL.

In its first formulation, the MI-IRL problem was solved via an EM algorithm considering a Boltzmann policy for the expert (Babes et al., 2011). This method, however, requires accessing the environment (that must have finite a state-action space) and needs a careful tuning of the temperature parameter. Our multiple-intention version of Σ-GIRL can be employed in continuous environments and just requires the selection of the number of clusters k, as unique hyperparameter. Bayesian non-parametric approaches can be employed when k is unknown at the cost of a significantly larger overhead, typically requiring the solution of the forward RL problem (Choi and Kim, 2012). The MI-IRL problem in continuous state-action spaces was addressed in a limited number of works (e.g., Almingol and Montesano, 2015; Rajasekaran et al., 2017).

7 Experiments

This section is devoted to the experimental evaluation of Σ-GIRL in both single-intention (Section 7.1) and multiple-intention (Section 7.2 and 7.3) settings.[5] For the single intention case, Σ-GIRL is compared with some batch model-free IRL algorithms in two continuous domains: the Linear Quadratic Gaussian regulator (LQG, Dorato et al., 2000) and a Gridworld with puddles domain (Figure 2). For the multiple-intention case, we test the quality of the clusters identified by Multiple-Intention Σ-GIRL compared to Maximum-

[5]The code is available at github.com/sigma-girl-MIIRL.

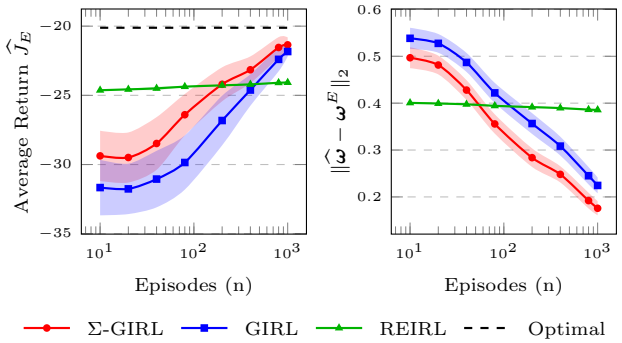

Figure 3: Average return in the original environment $\widehat{J}_E$ of the optimal policy given the recovered weights $\widehat{\boldsymbol{\omega}}$ (left) and distance between the recovered and expert's weights $\|\widehat{\boldsymbol{\omega}} - \boldsymbol{\omega}^E\|_2$ (right) in the LQG experiment. 100 runs, 95 % c.i.

Likelihood IRL (MLIRL, Babes et al., 2011) in the Gridworld with puddles. Finally, we evaluate Σ-GIRL in a real-world case study in which we infer and cluster the intentions of a group of Twitter users. Additional experimental results are reported in Appendix D.

7.1 Single-IRL experiments

We start evaluating the performance of Σ-GIRL compared with state-of-the-art model-free IRL algorithms in the single-intention IRL problem.

Linear Quadratic Gaussian regulator We consider the two-dimensional LQG environment in which the agent has to reach the origin, limiting the magnitude of the actions. The reward features are the state and actions squared $\boldsymbol{\phi}(\mathbf{s}, \mathbf{a}) = (-s_1^2, -s_2^2, -a_1^2, -a_2^2)^T$ and the weights are $\boldsymbol{\omega}^E = (1, 1, 1, 1)^T$. The expert plays a Gaussian linear policy, in which the control matrix $\mathbf{K}$ is computed in closed form and with fixed diagonal covariance $\mathrm{diag}(0.1, 2)$.[6] We show, for each of the algorithms considered, the performance of the optimal policies with the recovered reward function in the original environment $\widehat{J}_E$ and the distance between the weights found and the expert weights $\|\widehat{\boldsymbol{\omega}} - \boldsymbol{\omega}^E\|_2$, as a function of the number of trajectories. In Figure 3, we notice that Σ-GIRL outperforms both GIRL and REIRL in both indexes, achieving better performance and weights closer to the original ones. REIRL requires, besides the expert demonstrations, a dataset collected using a second policy uniform in the action space. This requirement partially violates the assumption of no interaction with the environment. The full sample covariance matrix was used in this experiment since it resulted well-conditioned.

[6]This asymmetric choice induces a higher variance in the second dimension of the state and action spaces; so that we can easily see the benefits of Σ-GIRL in modeling the gradient uncertainty.

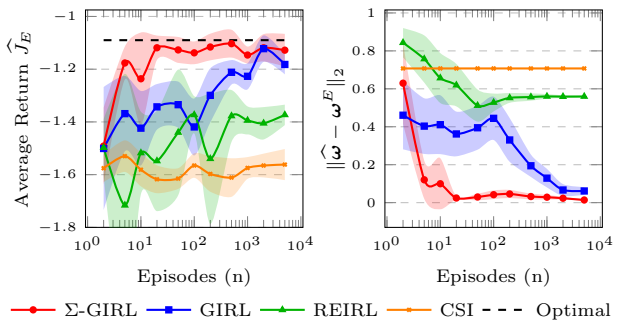

Figure 4: Average return in the original environment $\widehat{J}_E$ of the optimal policy trained with G(PO)MDP with the recovered weights $\widehat{\boldsymbol{\omega}}$ (left) and distance between the recovered and expert's weights $\|\widehat{\boldsymbol{\omega}} - \boldsymbol{\omega}^E\|_2$ (right) in the Gridworld experiment. 20 runs, 95 % c.i.

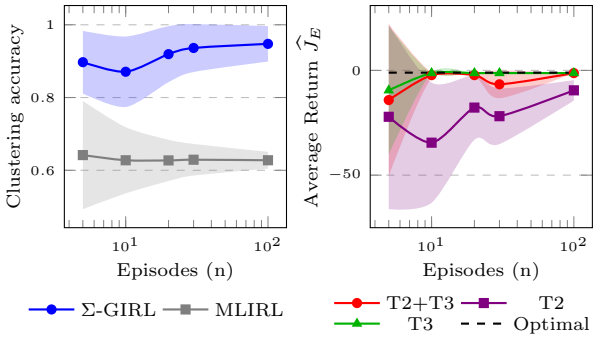

Figure 5: Clustering accuracy of Σ-GIRL and MLIRL (left) and average return in the original environment $\widehat{J}_E$ of the policies trained given the weights recovered separately for T2 and T3 and their cluster (right) in the Gridworld experiment. 20 runs, 98% c.i.

Gridworld The second experiment aims at evaluating the performance of Σ-GIRL in a continuous Gridworld environment. The agent is initialized in a random position and has to reach the goal in the minimum number of steps, by playing a bivariate Gaussian policy, linear in a set of 9×9 radial basis functions, that generates the x and y displacement. There is a region in the border of the environment that should be avoided. To make the environment more challenging the agent is also penalized for performing high magnitude actions. The reward feature space is given by three features: two binary features indicating whether the agent is on the border or in the central region and $-\|\boldsymbol{a}\|_2^2$ to penalize the magnitude of actions. The expert's weights are $\boldsymbol{\omega}^E = (1, 100, 0)^T$. In Figure 4, we compare the performance of Σ-GIRL with GIRL, REIRL, and CSI (Cascade Supervised IRL, Klein et al., 2013). We notice that Σ-GIRL is able to recover weights that are almost identical to the expert's ones even with a small (30) number of trajectories. This, as expected, reflects on the performance, in the original environment, of the optimal policy learned using the recovered weights. While GIRL is still able to obtain a good weighting, the performance of REIRL and CSI are significantly suboptimal, especially the latter that is unable to provide a suitable weighting.

7.2 Multiple-intentions experiments

In this experiment, we compare the Multiple-Intention Σ-GIRL with MLIRL (Babes et al., 2011), to test the capability of clustering agents which demonstrate multiple intentions. We consider a Gridworld-puddles consisting of a start state, a goal state, and some states are puddles, as in Figure 2. The world is characterized by a three-features reward: one for the goal state, one for the puddles and one for the other states. In this setting, we consider two clusters of agents which demonstrate different behaviors: the first cluster (T1) goes to the goal ignoring puddles, the second (T2+T3) goes to the goal avoiding puddles. For the second cluster, we have three agents which have three different but equivalently optimal policies (Figure 2). The first agent (T2) always chooses as the first action to go up, the second (T3) to go down and the third one randomly performs up or down as the first action, and then they all follow the border. The first cluster weights are $\boldsymbol{\omega}^{\mathrm{T1}} = (1, 1, 1)^T$ and the second cluster weights are $\boldsymbol{\omega}^{\mathrm{T2,T3}} = (1, 10, 1)^T$. The agents are initialized in the start state S and they play bivariate Gaussian policies linear in the state space. We set the number of clusters to 2 and with an increasing number of trajectories per agent (from 5 to 100). For MLIRL we set the same hyperparameters as in Babes et al. (2011), and we opportunely discretize the actions.

	Demonstrations per agent			
Running Time	5	10	30	100
Σ-GIRL	1.25s	1.28s	1.21s	1.51s
MLIRL	60.96s	62.20s	69.22s	93.23s

Table 1: Running time of Σ-GIRL and MLIRL with increasing number of trajectories per agent.

In Figure 5 left, we show the clustering accuracy of the two algorithms. Multiple-Intention Σ-GIRL succeeds in clustering the agents, even if they show different behaviors (but same intents), outperforming MLIRL. Figure 5 right shows the benefit of clustering in the IRL setting: the two experts T2 and T3, described above, optimized the same reward $\boldsymbol{\omega}^{\mathrm{T2+T3}}$ and indeed are clustered together. Considering only the trajectories of T2 and using her estimated gradients, Σ-GIRL cannot recover the right reward weights; instead, by clustering T2 together with T3, we correctly recover the expert weights. Finally, in Table 1, we report the running times with an increasing number of trajecto-

	Reward Features			
	Popularity	N. retweets	δ_{time}	N. agents
Cluster 1	0.56	0.00	0.44	4
Cluster 2	0.16	0.19	0.65	6
Cluster 3	0.78	0.03	0.19	4

Table 2: The reward weights learned by Σ-GIRL: popularity score of a retweet, number of retweets in a window T, and retweet proximity (δ_{time}).

ries. We notice that Σ-GIRL is faster than MLIRL and less sensitive to the sample size.

7.3 Twitter experiment

In the last experiment, we employ Σ-GIRL algorithm to cluster and infer Twitter users' intentions.[7] In particular, we turn to the questions: "Why does a user decide to retweet a post? What is her intention in deciding to post the tweet?"

Data The dataset is made of 14 Twitter accounts (agents) and their followings. The total number of followings is 5745. We collected their tweets from November 2018 to the end of January 2019 using a crawling process. The total number of followings' tweets is 468304. We suppose that every user can see only the tweets of whom she follows. We assume that a user sees a tweet with probability 0.01 to simulate the real behavior of a social network's user. After receiving a tweet the agent has to select between to actions: she can re-post it on her page or to not re-post it.

Features We represent the state space with three features: the *popularity* of a tweet, the *number of retweets* out of the last $T = 10$ (retweet window) tweets seen by the agent, and the *retweet proximity*. The popularity score is the weighted sum of the number of tweet's likes and the number of retweets:

$$\text{Popularity-score} = \alpha N_{\text{like}} + (1-\alpha) N_{\text{retweet}}$$

where $\alpha = 0.5$ and then normalized by the average of popularity-score of user's tweets. The retweet proximity is computed as $\delta_{\text{time}} = 0.1(t - t_0) - 1$ where t is the earliest time at which the agent receives a tweet that decides to retweet after having retweeted at time $t_0 < t$. When an agent performs a retweet, she goes to a next state s' that is composed of the *popularity* of the new tweet, $\delta_{\text{time}} = 0$ (since the last action was a retweet) and the number of retweet % 10. The reward features are the same as the state ones but the Popularity-score is set to 0 when the agent does not re-tweet the tweet.

[7] It is worth noting that MLIRL (Babes et al., 2011) cannot be applied in this experiment as we are in a fully batch setting and we cannot interact with the environment.

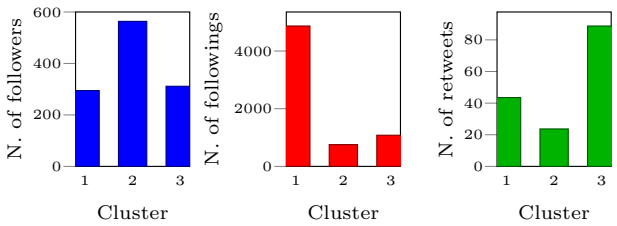

Figure 6: Twitter clustering statistics. Average number of followers (left), followings (center) and retweets (right) for each cluster.

Clustering results We perform behavioral cloning on the agents' demonstrations employing a two-layer neural network (8 neurons each). Then, we divide the demonstrations in trajectories of size 10, to have one retweet window in every trajectory. We apply Multiple-Intention Σ-GIRL with $k = 3$ clusters. The results are shown in Table 2, while Figure 6 reports some statistics on the three clusters found. The results underline that the first cluster is interested in retweeting posts with high popularity at a high frequency. Indeed, this cluster represents a standard Twitter user that follows many users and has a lower number of followers. The second cluster shows a different behavior: these agents do not want to retweet too often. They have not used the social network much, as they have few retweets and follow a small number of people. The last cluster is the most interesting one: these agents tend to retweet all popular tweets. Inspecting those we discover that they are commercial accounts (a bot, a company, and two HR managers). It is not surprising that they show the intention to post popular tweets, but they are uninterested in following other accounts.

8 Discussion and Conclusions

We presented a novel fully batch model-free IRL algorithm, Σ-GIRL, that accounts for the uncertainty in the gradient estimation, and provides the reward function that maximizes the likelihood of the estimated policy gradient, with no need of accessing the environment. We extended our algorithm to MI-IRL setting to simultaneously recover the agent-cluster assignment and the reward weights, using an EM alternating process. The experimental evaluation allowed us to illustrate the benefits of modeling the uncertainty both in the single-intent and multiple-intent setting, achieving an improvement over the considered baselines. Finally, we tested Σ-GIRL on a Twitter dataset showing qualitative results on inferring and clustering social network users' intentions. As future work, we plan to extend Σ-GIRL to the non-parametric setting and to use these learned intentions to predict the behavior of multiple users in multi-agent environments.

Acknowledgements

We thank Giuseppe Mascellaro for contributing to the idea of Σ-GIRL algorithm.

References

Almingol, J. and Montesano, L. (2015). Learning multiple behaviours using hierarchical clustering of rewards. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4608–4613. IEEE.

Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483.

Babes, M., Marivate, V., Subramanian, K., and Littman, M. L. (2011). Apprenticeship learning about multiple intentions. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 897–904.

Baxter, J. and Bartlett, P. L. (2001). Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350.

Bilmes, J. A. et al. (1998). A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126.

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*, pages 182–189.

Casella, G. and Berger, R. L. (2002). *Statistical inference*, volume 2. Duxbury Pacific Grove, CA.

Choi, J. and Kim, K.-E. (2012). Nonparametric bayesian inverse reinforcement learning for multiple reward functions. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 305–313, USA. Curran Associates Inc.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Dorato, P., Cerone, V., and Abdallah, C. (2000). *Linear quadratic control: an introduction*. Krieger Publishing Co., Inc.

Gupta, A. K. and Nagar, D. K. (2018). *Matrix variate distributions*. Chapman and Hall/CRC.

Hussein, A., Gaber, M. M., Elyan, E., and Jayne, C. (2017). Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):21.

Jain, V., Doshi, P., and Banerjee, B. (2019). Model-free irl using maximum likelihood estimation.

Klein, E., Geist, M., Piot, B., and Pietquin, O. (2012). Inverse reinforcement learning through structured classification. In *Advances in Neural Information Processing Systems*, pages 1007–1015.

Klein, E., Piot, B., Geist, M., and Pietquin, O. (2013). A cascaded supervised learning approach to inverse reinforcement learning. In *Proceedings of the 2013th European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part I*, ECMLPKDD'13, pages 1–16, Berlin, Heidelberg. Springer-Verlag.

Knyazev, A., Jujunashvili, A., and Argentati, M. (2010). Angles between infinite dimensional subspaces with applications to the rayleigh–ritz and alternating projectors methods. *Journal of Functional Analysis*, 259(6):1323–1345.

Ledoit, O. and Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2):365–411.

Manton, J. H., Mahony, R., and Hua, Y. (2003). The geometry of weighted low-rank approximations. *IEEE Transactions on Signal Processing*, 51(2):500–514.

Metelli, A. M., Pirotta, M., and Restelli, M. (2017). Compatible reward inverse reinforcement learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 2050–2059. Curran Associates, Inc.

Ng, A. Y. and Russell, S. J. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 663–670, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Nocedal, J. and Wright, S. (2006). *Numerical optimization*. Springer Science & Business Media.

Osa, T., Pajarinen, J., Neumann, G., Bagnell, J. A., Abbeel, P., Peters, J., et al. (2018). An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 7(1-2):1–179.

Peters, J. and Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697.

Pirotta, M. (2016). *Reinforcement learning: from theory to algorithms*. PhD thesis, Italy.

Pirotta, M. and Restelli, M. (2016). Inverse reinforcement learning through policy gradient minimization. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 1993–1999. AAAI Press.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA.

Rajasekaran, S., Zhang, J., and Fu, J. (2017). Inverse reinforce learning with nonparametric behavior clustering. *CoRR*, abs/1712.05514.

Spokoiny, V. et al. (2012). Parametric estimation. finite sample theory. *The Annals of Statistics*, 40(6):2877–2909.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. A Bradford book. Bradford Book.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In Solla, S., Leen, T., and Müller, K., editors, *Advances in Neural Information Processing Systems 12*, pages 1057–1063. MIT Press.

Taslaman, L. (2014). The principal angles and the gap.

Tateo, D., Pirotta, M., Restelli, M., and Bonarini, A. (2017). Gradient-based minimization for multi-expert inverse reinforcement learning. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Ziebart, B. D., Maas, A., Bagnell, J. A., and Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, pages 1433–1438.