Risk-Sensitive Generative Adversarial Imitation Learning

Jonathan Lacotte Stanford University
Mohammad Ghavamzadeh Facebook AI Research
Yinlam Chow DeepMind
Marco Pavone Stanford University

Abstract

We study risk-sensitive imitation learning where the agent's goal is to perform at least as well as the expert in terms of a risk profile. We first formulate our risk-sensitive imitation learning setting. We consider the generative adversarial approach to imitation learning (GAIL) and derive an optimization problem for our formulation, which we call it risk-sensitive GAIL (RS-GAIL). We then derive two different versions of our RS-GAIL optimization problem that aim at matching the risk profiles of the agent and the expert w.r.t. Jensen-Shannon (JS) divergence and Wasserstein distance, and develop risk-sensitive generative adversarial imitation learning algorithms based on these optimization problems. We evaluate the performance of our algorithms and compare them with GAIL and the risk-averse imitation learning (RAIL) algorithms in two MuJoCo and two OpenAI classical control tasks.

1 Introduction

We study imitation learning, i.e., the problem of learning to perform a task from the sample trajectories generated by an expert. There are three main approaches to this problem: **1)** behavioral cloning (e.g., Pomerleau [1991]) in which the agent learns a policy by solving a supervised learning problem over the state-action pairs of the expert's trajectories, **2)** inverse reinforcement learning (IRL) (e.g., Ng and Russell [2000]) followed by reinforcement learning (RL), a process also referred to as RL$\circ$IRL, where we first find a cost function under which the expert is optimal (IRL) and then return the optimal policy w.r.t. this cost function (RL), and **3)** generative adversarial imitation learning (GAIL) [Ho and Ermon, 2016a] that frames the imitation learning problem as occupancy measure matching w.r.t. either the Jensen-Shannon divergence (GAIL) [Ho and Ermon, 2016a] or the Wasserstein distance (InfoGAIL) [Li et al., 2017]. Behavioral cloning algorithms are simple but often need a large amount of data to be successful. IRL does not suffer from the main problems of behavioral cloning [Ross and Bagnell, 2010, Ross et al., 2011], since it takes entire trajectories into account (instead of single time-step decisions) when learning a cost function. However, IRL algorithms are often expensive to run as they require solving a RL problem in their inner loop. This issue had restricted the use of IRL to small problems for a long while and only recently scalable IRL algorithms have been developed [Levine and Koltun, 2012, Finn et al., 2016]. On the other hand, the nice feature of the GAIL approach to imitation learning is that it bypasses the intermediate IRL step and directly learns a policy from data, as if it were obtained by RL$\circ$IRL. The resulting algorithm is closely related to generative adversarial networks (GAN) [Goodfellow et al., 2014] that has recently gained attention in the deep learning community.

In many applications, we may prefer to optimize some measure of risk in addition to the standard optimization criterion, i.e., the expected sum of (discounted) costs. In such cases, we would like to use a criterion that incorporates a penalty for the variability (due to the stochastic nature of the system) induced by a given policy. Several risk-sensitive criteria have been studied in the literature of risk-sensitive Markov decision processes (MDPs) [Howard and Matheson, 1972] including the expected exponential utility (e.g., Howard and Matheson [1972], Borkar [2001]), a variance-related measure (e.g., Sobel [1982], Tamar et al. [2012], Prashanth and Ghavamzadeh [2013, 2016]), or the tail-related measures like value-at-risk (VaR) and conditional value-at-risk (CVaR) (e.g., Rockafellar and Uryasev [2002], Chow and Ghavamzadeh [2014], Tamar et al. [2015b]).

In risk-sensitive imitation learning, the agent's goal is to perform at least as well as the expert in terms of one or more risk-sensitive objective(s), e.g., mean + λCVaR$_\alpha$, for one or more values of $\lambda \geq 0$. This goal cannot be satisfied by risk-neutral imitation learning. As we will

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

show in Section 3.3, if we use GAIL to minimize the Wasserstein distance between the occupancy measures of the agent and the expert, the distance between their CVaRs could be still large. Santara et al. [2017a] recently showed empirically that the policy learned by GAIL does not have the desirable tail properties, such as VaR and CVaR, and proposed a modification of GAIL, called risk-averse imitation learning (RAIL), to address this issue. We will discuss about RAIL in more details in Section 5 as it is probably the closest work to us in the literature. Another related work is by Singh et al. [2018] on risk-sensitive IRL in which the proposed algorithm infers not only the expert's cost function but her underlying risk measure, for a rich class of static and dynamic risk measures (coherent risk measures). The agent then learns a policy by optimizing the inferred risk-sensitive objective.

In this paper, we study an imitation learning setting in which the agent's goal is to learn a policy with minimum expected sum of (discounted) costs and with CVaR_α that is at least as well as that of the expert. We first provide a mathematical formulation for this setting and derive a GAIL-like optimization problem for our formulation, which we call it risk-sensitive GAIL (RS-GAIL), in Section 3.1. In Sections 3.2 and 3.3, we define cost function regularizers that when we compute their convex conjugates and plug them into our RS-GAIL objective function, the resulting optimization problems aim at learning the expert's policy by matching occupancy measures w.r.t. Jensen-Shannon (JS) divergence and Wasserstein distance, respectively. We call the resulting optimization problems JS-RS-GAIL and W-RS-GAIL and propose our risk-sensitive generative adversarial imitation learning algorithms based on these optimization problems in Section 4. It is important to note that unlike the risk-neutral case in which the occupancy measure of the agent is matched with that of the expert, here in the risk-sensitive case, we match two sets of occupancy measures that encode the risk profile of the agent and the expert. This will become more clear in Section 3. We present our understanding of RAIL and how it is related to our work in Section 5. In Section 6, we evaluate the performance of our algorithms and compare them with GAIL and RAIL in two MuJoCo tasks [Todorov et al., 2012a] that have also been used in the GAIL [Ho and Ermon, 2016a] and RAIL [Santara et al., 2017a] papers, as well as two OpenAI classical control problems [Brockman et al., 2016]. Finally in Section 7, we conclude the paper and list a number of future directions.

2 Preliminaries

We consider the scenario in which the agent's interaction with the environment is modeled as a Markov decision process (MDP). A MDP is a tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, c, p, p_0, \gamma\}$, where $\mathcal{S}$ and $\mathcal{A}$ are state and action spaces; $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $p : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ are the cost function and transition probability distribution, with $c(s,a)$ and $p(\cdot|s,a)$ being the cost and next state probability of taking action a in state s; $p_0 : \mathcal{S} \to \Delta_{\mathcal{S}}$ is the initial state distribution; and $\gamma \in [0,1)$ is a discounting factor. A stationary stochastic policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ is a mapping from states to a distribution over actions. We denote by Π the set of all such policies. We denote by $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T) \in \Gamma$, where $a_t \sim \pi(\cdot|s_t)$, $\forall t \in \{0, \ldots, T-1\}$, a trajectory of the fixed horizon T generated by policy π, by Γ the set of all such trajectories, and by $C(\tau) = \sum_{t=0}^{T-1} \gamma^t c(s_t, a_t)$ the *loss* of trajectory τ. The probability of trajectory τ is given by $\mathbb{P}(\tau|\pi) = p^\pi(\tau) = p_0(s_0) \prod_{t=0}^{T-1} \pi(a_t|s_t) p(s_{t+1}|s_t, a_t)$. We denote by C^π the random variable of the loss of policy π. Thus, when $\tau \sim p^\pi$, $C(\tau)$ is an instantiation of the random variable C^π. The performance of a policy π is usually measured by a quantity related to the loss of the trajectories it generates, the most common would be its expectation, i.e., $\mathbb{E}[C^\pi] = \mathbb{E}_{\tau \sim p^\pi}[C(\tau)]$. We define the occupancy measure of policy π as $d^\pi(s,a) = \sum_{t=0}^{T} \gamma^t \mathbb{P}(s_t = s, a_t = a|\pi)$, which can be interpreted as the unnormalized distribution of the state-action pairs visited by the agent under policy π. Using occupancy measure, we may write the policy's performance as $\mathbb{E}[C^\pi] = \mathbb{E}_{p^\pi}[C(\tau)] = \mathbb{E}_{d^\pi}[c(s,a)] = \sum_{s,a} d^\pi(s,a) c(s,a)$.

2.1 Risk-sensitive MDPs

In risk-sensitive decision-making, in addition to optimizing the expectation of the loss, it is also important to control the variability of this random variable. This variability is often measured by the variance or tail-related quantities such as value-at-risk (VaR) and conditional value-at-risk (CVaR). Given a policy π and a confidence level $\alpha \in (0,1]$, we define the VaR at level α of the loss random variable C^π as its (left-side) $(1-\alpha)$-quantile, i.e., $\nu_\alpha[C^\pi] := \inf\{t \in \mathbb{R} \mid \mathbb{P}(C^\pi \leq t) \geq 1-\alpha\}$ and its CVaR at level α as $\rho_\alpha[C^\pi] = \inf_{\nu \in \mathbb{R}} \left\{\nu + \frac{1}{\alpha}\mathbb{E}\big[(C^\pi - \nu)_+\big]\right\}$, where $x_+ = \max(x, 0)$. We also define the *risk envelope* $\mathcal{U}^\pi = \left\{\zeta : \Gamma^\pi \to [0, \frac{1}{\alpha}] \mid \sum_{\tau \in \Gamma} \zeta(\tau) \cdot p^\pi(\tau) = 1\right\}$, which is a compact, convex, and bounded set. The quantities $p^\pi_\zeta = \zeta \cdot p^\pi$, $\zeta \in \mathcal{U}^\pi$ are called *distorted probability distributions*, and we denote by $\mathcal{P}^\pi_\zeta = \left\{p^\pi_\zeta \mid \zeta \in \mathcal{U}^\pi\right\}$ the set of such distributions. The set $\mathcal{P}^\pi_\zeta$ induces a set of *distorted occupancy measures* $\mathcal{D}^\pi_\zeta$, where each element of $\mathcal{D}^\pi_\zeta$ is the occupancy measure induced by a distorted probability distribution in $\mathcal{P}^\pi_\zeta$. The sets $\mathcal{P}^\pi_\zeta$ and $\mathcal{D}^\pi_\zeta$ characterize the risk of policy π. Given the risk envelope $\mathcal{U}^\pi$, we may define the dual representation of CVaR as $\rho_\alpha[C^\pi] = \sup_{\zeta \in \mathcal{U}^\pi} \mathbb{E}_{\tau \sim p^\pi}\big[\zeta(\tau) C(\tau)\big]$, where the supremum is attained at the density $\zeta^*(\tau) = \frac{1}{\alpha}\mathbf{1}_{\{C(\tau) \geq \nu_\alpha[C^\pi]\}}$. Hence, CVaR can be considered as

the expectation of the loss random variable, when the trajectories are generated from the distorted distribution $p^\pi_{\zeta^*} = \zeta^* \cdot p^\pi$, i.e., $\rho_\alpha[C^\pi] = \mathbb{E}_{\tau\sim p^\pi_{\zeta^*}}[C(\tau)]$. If we denote by $d^\pi_{\zeta^*} \in \mathcal{D}^\pi_\zeta$ the distorted occupancy measure induced by $p^\pi_{\zeta^*}$, then we may write the CVaR as $\rho_\alpha[C^\pi] = \mathbb{E}_{p^\pi_{\zeta^*}}[C(\tau)] = \mathbb{E}_{d^\pi_{\zeta^*}}[c(s,a)]$.

2.2 Generative Adversarial Imitation Learning

As discussed in Section 1, generative adversarial imitation learning (GAIL) [Ho and Ermon, 2016a] is a framework for directly extracting a policy from the trajectories generated by an expert policy π_E, as if it were obtained by inverse RL (IRL) followed by RL, i.e., RL$\circ$IRL(π_E). The main idea behind GAIL is to formulate imitation learning as occupancy measure matching w.r.t. the Jensen-Shannon divergence D_{JS}, i.e., $\min_\pi \big(D_{\mathrm{JS}}(d^\pi, d^{\pi_E}) - \lambda H(\pi)\big)$, where $H(\pi) = \mathbb{E}_{(s,a)\sim d^\pi}[-\log\pi(a|s)]$ is the γ-discounted causal entropy of policy π, $\lambda \geq 0$ is a regularization parameter, and $D_{\mathrm{JS}}(d^\pi, d^{\pi_E}) := \sup_{f:\mathcal{S}\times\mathcal{A}\to(0,1)} \mathbb{E}_{d^\pi}[\log f(s,a)] + \mathbb{E}_{d^{\pi_E}}[\log(1-f(s,a))]$. Li et al. [2017] proposed Info-GAIL by reformulating GAIL and replacing the Jensen-Shannon divergence $D_{\mathrm{JS}}(d^\pi, d^{\pi_E})$ with the Wasserstein distance $W(d^\pi, d^{\pi_E}) := \sup_{f\in\mathcal{F}_1} \mathbb{E}_{d^\pi}[f(s,a)] - \mathbb{E}_{d^{\pi_E}}[f(s,a)]$, where $\mathcal{F}_1$ is the set of 1-Lipschitz functions over $\mathcal{S}\times\mathcal{A}$.

3 Risk-sensitive Imitation Learning

In this section, we describe the risk-sensitive imitation learning formulation studied in the paper and derive the optimization problems that our proposed algorithms solve to learn a risk-sensitive policy from the expert's trajectories.

3.1 Problem Formulation

As described in Section 1, we consider the risk-sensitive imitation learning setting in which the agent's goal is to learn a policy with minimum loss and with CVaR that is at least as well as that of the expert. Thus, the agent solves the optimization problem

$$\min_\pi \ \mathbb{E}[C^\pi] \qquad , \qquad \text{s.t. } \rho_\alpha[C^\pi] \leq \rho_\alpha[C^{\pi_E}], \tag{1}$$

where C^π is the loss of policy π w.r.t. the expert's cost function c that is unknown to the agent. The optimization problem (1) without the loss of optimality is equivalent to the unconstrained problem

$$\min_\pi \sup_{\lambda\geq 0} \ \mathbb{E}[C^\pi] - \mathbb{E}[C^{\pi_E}] + \lambda\big(\rho_\alpha[C^\pi] - \rho_\alpha[C^{\pi_E}]\big). \tag{2}$$

Note that π_E is a solution of both (1) and (2). However, since the expert's cost function is unknown, the agent cannot directly solve (2), and thus, considers the surrogate problem

$$\min_\pi \sup_{f\in\mathcal{C}} \sup_{\lambda\geq 0} \ \mathbb{E}[C^\pi_f] - \mathbb{E}[C^{\pi_E}_f] + \lambda\big(\rho_\alpha[C^\pi_f] - \rho_\alpha[C^{\pi_E}_f]\big), \tag{3}$$

where $\mathcal{C} = \{f : \mathcal{S}\times\mathcal{A} \to \mathbb{R}\}$ and C^π_f is the loss of policy π w.r.t. the cost function f. We employ the Lagrangian relaxation procedure [Bertsekas, 1999] to swap the inner maximization over λ with the minimization over π and convert (3) to the problem

$$\sup_{\lambda\geq 0} \min_\pi \sup_{f\in\mathcal{C}} \ \mathbb{E}[C^\pi_f] - \mathbb{E}[C^{\pi_E}_f] + \lambda\big(\rho_\alpha[C^\pi_f] - \rho_\alpha[C^{\pi_E}_f]\big). \tag{4}$$

In order to enforce exploration of the imitation learning agent, we adopt maximum causal entropy IRL formulation [Ziebart et al., 2008, 2010] and add $-H(\pi)$ to the optimization problem (4). Moreover, since $\mathcal{C}$ is large, to avoid overfitting when we are provided with a finite set of expert's trajectories, we add the negative of a convex regularizer $\psi : \mathcal{C} \to \mathbb{R}\cup\{\infty\}$ to the optimization problem (4). As a result we obtain the following optimization problem for our risk-sensitive imitation learning setting, which we call it RS-GAIL:

$$\textbf{(RS-GAIL)} \qquad \sup_{\lambda\geq 0} \min_\pi \ -H(\pi) + \mathcal{L}_\lambda(\pi, \pi_E), \tag{5}$$

where $\mathcal{L}_\lambda(\pi,\pi_E) := \sup_{f\in\mathcal{C}} (1+\lambda)\big(\rho^\lambda_\alpha[C^\pi_f] - \rho^\lambda_\alpha[C^{\pi_E}_f]\big) - \psi(f)$, with $\rho^\lambda_\alpha[C^\pi_f] := \frac{\mathbb{E}[C^\pi_f] + \lambda\rho_\alpha[C^\pi_f]}{1+\lambda}$ being the coherent risk measure for policy π corresponding to mean-CVaR with the risk parameter λ. The parameter λ can be interpreted as the tradeoff between the mean performance and risk-sensitivity of the policy. The objective function $\mathcal{L}_\lambda(\pi,\pi_E)$ can be decomposed into three terms: **1)** the difference between the agent and the expert in terms of mean performance, $\mathbb{E}[C^\pi_f] - \mathbb{E}[C^{\pi_E}_f]$, which corresponds to the standard generative imitation learning objective, **2)** the difference between the agent and the expert in terms of risk $\rho_\alpha[C^\pi_f] - \rho_\alpha[C^{\pi_E}_f]$, and **3)** the convex regularizer $\psi(f)$ that encodes our belief about the expert cost function f.

For the risk-sensitive quantity $\rho^\lambda_\alpha[C^\pi]$, we define the distorted probability distributions $p^\pi_\xi = \xi \cdot p^\pi$, where $\xi = \frac{1+\lambda\zeta}{1+\lambda}$, $\zeta \in \mathcal{U}^\pi$. We denote by $\mathcal{P}^\pi_\xi$ the set of such distorted distributions and by $\mathcal{D}^\pi_\xi$ the set of distorted occupancy measures induced by the elements of $\mathcal{P}^\pi_\xi$. Similar to CVaR in Section 2.1, we may write the risk-sensitive quantity $\rho^\lambda_\alpha[C^\pi]$ as the expectation $\rho^\lambda_\alpha[C^\pi] = \mathbb{E}_{p^\pi_{\xi^*}}[C(\tau)] = \mathbb{E}_{d^\pi_{\xi^*}}[c(s,a)]$, where $\xi^* = \frac{1+\lambda\zeta^*}{1+\lambda}$ with ζ^* defined in Section 2.1 and $d^\pi_{\xi^*} \in \mathcal{D}^\pi_\xi$ is the distorted occupancy measure induced by $p^\pi_{\xi^*} \in \mathcal{P}^\pi_\xi$.

In Theorem 1, we show that the maximization problem $\mathcal{L}_\lambda(\pi,\pi_E)$ over the cost function $f \in \mathcal{C}$ can be rewritten as a sup-inf problem over the distorted occupancy measures $d \in \mathcal{D}^\pi_\xi$ and $d' \in \mathcal{D}^{\pi_E}_\xi$.

Theorem 1. *Let $\psi : \mathcal{C} \to \mathbb{R}\cup\{\infty\}$ be a convex cost function regularizer. Then,*

$$\begin{aligned}\mathcal{L}_\lambda(\pi,\pi_E) &= \sup_{f\in\mathcal{C}} (1+\lambda)\big(\rho^\lambda_\alpha[C^\pi_f] - \rho^\lambda_\alpha[C^{\pi_E}_f]\big) - \psi(f) \\ &= \sup_{d\in\mathcal{D}^\pi_\xi} \inf_{d'\in\mathcal{D}^{\pi_E}_\xi} \psi^*\big((1+\lambda)(d-d')\big),\end{aligned} \tag{6}$$

where ψ^* is the convex conjugate function of ψ, i.e., $\psi^*(d) = \sup_{f\in\mathcal{C}} d^\top f - \psi(f)$.

Proof. See Appendix A. □

From Theorem 1, we may write the RS-GAIL optimization problem (5) as

$$\textbf{(RS-GAIL)} \quad \sup_{\lambda\geq 0} \min_{\pi} \; -H(\pi) + \sup_{d\in\mathcal{D}_\xi^\pi} \inf_{d'\in\mathcal{D}_\xi^{\pi_E}} \psi^*\big((1+\lambda)(d-d')\big). \tag{7}$$

The main difference between the RS-GAIL optimization problem (7) and that of GAIL (Eq. 4 in Ho and Ermon [2016a]) is the $\sup_{\mathcal{D}_\xi^\pi} \inf_{\mathcal{D}_\xi^{\pi_E}}$ in (7). In the risk-neutral case, $\lambda = 0$, thus, the two sets of distorted occupancy measures $\mathcal{D}_\xi^\pi$ and $\mathcal{D}_\xi^{\pi_E}$ are singleton, and the RS-GAIL optimization problem is reduced to that of GAIL.

Example 1. *Let* $\psi(f) = \begin{cases} 0 & \text{if } ||f||_\infty \leq 1 \\ +\infty & \text{otherwise} \end{cases}$, *then* $\mathcal{L}_\lambda(\pi, \pi_E) = 2(1+\lambda)\sup_{d\in\mathcal{D}_\xi^\pi} \inf_{d'\in\mathcal{D}_\xi^{\pi_E}} ||d-d'||_{TV}$, *where* $||d-d'||_{TV}$ *is the total variation distance between* d *and* d'. *Note that similar to GAIL, our optimization problem aims at learning the expert's policy by matching occupancy measures. However, in order to take risk into account, it now involves matching two sets of occupancy measures (w.r.t. the total variation distance) that encode the risk profile of each policy.*

3.2 Risk-sensitive GAIL with Jensen-Shannon Divergence

In this section, we derive RS-GAIL using occupation measure matching via Jensen-Shannon (JS) divergence. We define the difference-of-convex cost function regularizer $\psi(f) := \begin{cases} (1+\lambda)\big(-\rho_\alpha^\lambda[C_f^{\pi_E}] + \rho_\alpha^\lambda[G_f^{\pi_E}]\big) & \text{if } f < 0 \\ +\infty & \text{otherwise} \end{cases}$, where $C_f^{\pi_E}$ and $G_f^{\pi_E}$ are the loss random variables of policy π_E w.r.t. the cost functions $c(s,a) = f(s,a)$ and $c(s,a) = g\big(f(s,a)\big)$, respectively, with $g(x) := \begin{cases} -\log(1-e^x) & \text{if } x < 0 \\ +\infty & \text{otherwise} \end{cases}$. To clarify, $G_f^{\pi_E}$ is a random variable whose instantiations are $G_f(\tau) = \sum_{t=0}^{T-1} \gamma^t g\big(f(s_t,a_t)\big)$, where $\tau \sim p^{\pi_E}$ is a trajectory generated by the expert policy π_E. Similar to the description in Ho and Ermon [2016a], this regularizer places low penalty on cost functions f that assign negative cost to expert's state-action pairs. However, if f assigns large costs (close to zero, which is the upper-bound of the regularizer) to the expert, then ψ will heavily penalize f. In the following theorems, whose proofs are reported in Appendix B, we derive the optimization problem of the JS version of our RS-GAIL algorithm by computing (6) for the above choice of the cost function regularizer $\psi(f)$. We prove the following results directly from the RS-GAIL optimization problem (5).

Theorem 2. *With the cost function regularizer $\psi(f)$ defined above, we may write*

$$\mathcal{L}_\lambda(\pi, \pi_E) = (1+\lambda) \sup_{f:\mathcal{S}\times\mathcal{A}\to(0,1)} \rho_\alpha^\lambda[F_{1,f}^\pi] - \rho_\alpha^\lambda[-F_{2,f}^{\pi_E}], \tag{8}$$

where F_1^π and $F_2^{\pi_E}$ are the loss random variables of policies π and π_E w.r.t. the cost functions $c(s,a) = \log f(s,a)$ and $c(s,a) = \log\big(1-f(s,a)\big)$, respectively.

Corollary 1. *We may write $\mathcal{L}_\lambda(\pi, \pi_E)$ in terms of the Jensen-Shannon (JS) divergence as*

$$\mathcal{L}_\lambda(\pi, \pi_E) = (1+\lambda) \sup_{d\in\mathcal{D}_\xi^\pi} \inf_{d'\in\mathcal{D}_\xi^{\pi_E}} D_{JS}(d, d'). \tag{9}$$

From Theorem 2, we write the optimization problem of the JS version of our RS-GAIL algorithm as

$$\textbf{(JS-RS-GAIL)} \quad \sup_{\lambda\geq 0} \min_{\pi} \; -H(\pi) + (1+\lambda) \sup_{f:\mathcal{S}\times\mathcal{A}\to(0,1)} \rho_\alpha^\lambda[F_{1,f}^\pi] - \rho_\alpha^\lambda[-F_{2,f}^{\pi_E}]. \tag{10}$$

Hence in JS-RS-GAIL, instead of minimizing the original GAIL objective, we solve the optimization problem (10) that aims at matching the sets $\mathcal{D}_\xi^\pi$ and $\mathcal{D}_\xi^{\pi_E}$ w.r.t. the JS divergence.

3.3 Risk-sensitive GAIL with Wasserstein Distance

In this section, we derive RS-GAIL using occupation measure matching via the Wasserstein distance. We define the cost function regularizer $\psi(f) := \begin{cases} 0 & \text{if } f \in \mathcal{F}_1 \\ +\infty & \text{otherwise} \end{cases}$.

Corollary 2. *For the cost function regularizer $\psi(f)$ defined above, we may write*

$$\mathcal{L}_\lambda(\pi, \pi_E) = (1+\lambda) \sup_{d\in\mathcal{D}_\xi^\pi} \inf_{d'\in\mathcal{D}_\xi^{\pi_E}} W(d, d'). \tag{11}$$

Proof. See Appendix C. □

From (6) and the cost function regularizer $\psi(f)$ defined above, we have $\mathcal{L}_\lambda(\pi, \pi_E) = \sup_{f\in\mathcal{F}_1} \rho_\alpha^\lambda[C_f^\pi] - \rho_\alpha^\lambda[C_f^{\pi_E}]$, which gives the following optimization problem for the Wasserstein version of our RS-GAIL algorithm:

$$\textbf{(W-RS-GAIL)} \quad \sup_{\lambda\geq 0} \min_{\pi} \; -H(\pi) + (1+\lambda) \sup_{f\in\mathcal{F}_1} \rho_\alpha^\lambda[C_f^\pi] - \rho_\alpha^\lambda[C_f^{\pi_E}]. \tag{12}$$

We conclude this section with a theorem that shows if we use a risk-neutral imitation learning algorithm to minimize the Wasserstein distance between the occupancy measures of the agent and the expert, the distance between their CVaRs could be still large. Thus, new algorithms, such as those developed in this paper, are needed for risk-sensitive imitation learning.

Theorem 3. *Let Δ be the worst-case risk difference between the agent and the expert, given that their occupancy measures are δ-close ($\delta > 0$), i.e.,*

$$\Delta = \sup_{\pi,p,p_0} \sup_{f\in\mathcal{F}_1} \rho_\alpha[C_f^\pi] - \rho_\alpha[C_f^{\pi_E}], \quad s.t.\ W(d^\pi, d^{\pi_E}) \le \delta.$$

Then, $\Delta \ge \frac{\delta}{\alpha}$.

Theorem 3, whose proof has been reported in Appendix C, indicates that the difference between the risks can be $1/\alpha$-times larger than that between the occupancy measures (in terms of Wasserstein distance).

4 Risk-sensitive Imitation Learning Algorithms

Algorithm 1 contains the pseudocode of our JS-based and Wasserstein-based risk-sensitive imitation learning algorithms. The algorithms aim at finding a saddle-point (π, f) of the objective function (5). We use the parameterizations for the policy $\theta \mapsto \pi_\theta$ and cost function (discriminator) $w \mapsto f_w$. Similar to GAIL [Ho and Ermon, 2016a], the algorithm is TRPO-based [Schulman et al., 2015] and alternates between an Adam [Kingma and Ba, 2014] gradient ascent step for the cost function parameter w and a KL-constrained gradient descent step w.r.t. a linear approximation of the objective. The details about the algorithm, including the estimation of the gradients and the VaRs are reported in Appendix D.

In the implementation of our algorithms, we use a grid search and optimize over a finite number of the Lagrangian parameters λ. This can be seen as the agent selects among a finite number of risk profiles of the form (mean + λCVaR$_\alpha$) when she matches her risk profile to that of the expert.

5 Discussion about RAIL

We start this section by comparing the RAIL optimization problem (Eq. 9 in Santara et al. [2017a]) with that of our JS-RS-GAIL reported in Eq. 10, i.e.,

$$\textbf{(RAIL)} \quad \min_\pi\ -H(\pi) + (1+\lambda) \sup_{f:\mathcal{S}\times\mathcal{A}\to(0,1)} \rho_\alpha^\lambda[F_{1,f}^\pi] - \mathbb{E}[-F_{2,f}^{\pi_E}],$$

$$\textbf{(JS-RS-GAIL)} \quad \min_\pi\ -H(\pi) + (1+\lambda) \sup_{f:\mathcal{S}\times\mathcal{A}\to(0,1)} \rho_\alpha^\lambda[F_{1,f}^\pi] - \rho_\alpha^\lambda[-F_{2,f}^{\pi_E}].$$

If we write the above optimization problems in terms of the JS divergence, we obtain

Algorithm 1 Pseudocode of JS-RS-GAIL and W-RS-GAIL Algorithms.

1: **Input:** Expert trajectories $\{\tau_j^E\}_{j=1}^{N_E} \sim p^{\pi_E}$, Risk level $\alpha \in (0,1]$, Initial policy and cost function parameters θ_0 and w_0.
2: **for** $i = 0, 1, 2, \dots$ **do**
3: Generate N trajectories using the current policy π_{θ_i}, i.e., $\{\tau_j\}_{j=1}^{N} \sim p^{\pi_{\theta_i}}$
4: Estimate VaRs $\hat\nu_\alpha(F_{1,f_{w_i}}^\pi)$ and $\hat\nu_\alpha(-F_{2,f_{w_i}}^{\pi_E})$ **(JS)**
5: Estimate VaRs $\hat\nu_\alpha(C_{f_{w_i}}^\pi)$ and $\hat\nu_\alpha(C_{f_{w_i}}^{\pi_E})$ **(W)**
6: Update the discriminator parameter by computing a gradient ascent step w.r.t. the objective

$$w_{i+1} \mapsto (1+\lambda)\left(\rho_\alpha^\lambda[F_{1,f_{w_i}}^{\pi_{\theta_i}}] - \rho_\alpha^\lambda[-F_{2,f_{w_i}}^{\pi_E}]\right) \quad \textbf{(JS)}$$

$$w_{i+1} \mapsto (1+\lambda)\left(\rho_\alpha^\lambda[C_{f_{w_i}}^{\pi_{\theta_i}}] - \rho_\alpha^\lambda[C_{f_{w_i}}^{\pi_E}]\right) \quad \textbf{(W)}$$

7: Update the policy parameter using a KL-constrained gradient descent step w.r.t. the objective

$$\theta_{i+1} \mapsto -H(\pi_{\theta_i}) + (1+\lambda)\rho_\alpha^\lambda[F_{1,f_{w_{i+1}}}^{\pi_{\theta_i}}] \quad \textbf{(JS)}$$

$$\theta_{i+1} \mapsto -H(\pi_{\theta_i}) + (1+\lambda)\rho_\alpha^\lambda[C_{f_{w_{i+1}}}^{\pi_{\theta_i}}] \quad \textbf{(W)}$$

8: **end for**

$$\textbf{(RAIL)} \quad \min_\pi\ -H(\pi) + (1+\lambda) \sup_{d\in\mathcal{D}_\xi^\pi} D_{\mathrm{JS}}(d, d^{\pi_E}), \tag{13}$$

$$\textbf{(JS-RS-GAIL)} \quad \min_\pi\ -H(\pi) + (1+\lambda) \sup_{d\in\mathcal{D}_\xi^\pi} \inf_{d'\in\mathcal{D}_\xi^{\pi_E}} D_{\mathrm{JS}}(d, d') \quad \textit{(see Eq. 9).} \tag{14}$$

Note that while the JS in (14) matches the distorted occupancy measures (risk profiles) of the agent and the expert, the JS in (13) matches the distorted occupancy measure (risk profile) of the agent with the occupancy measure (mean) of the expert. This means that RAIL does not take the expert's risk into account in its optimization.

Moreover, the results reported in Santara et al. [2017a] indicate that GAIL optimizes the risk (VaR and CVaR) poorly. By looking at the RAIL's GitHub [Santara et al., 2017b], it seems they used the GAIL implementation from its GitHub [Ho and Ermon, 2016b]. Although we used the same GAIL implementation, we did not observe such a poor performance for GAIL, which is not that surprising since the MuJoCo domains used in the GAIL and RAIL papers are all deterministic and the policies are the only source of randomness there. This is why in our MuJoCo experiments in Section 6, we inject noise to the reward function of the problems. Finally, the gradient of the objective function reported in Eq. (A.3) of Santara et al. [2017a] is a scalar, which does not seem to be correct. We corrected this in our implementation of RAIL in Section 6.

6 Experiments

In this section, we evaluate the performance of our JS and Wasserstein-based algorithms and compare them with GAIL and RAIL algorithms in two MuJoCo and two OpenAI classical control tasks.

6.1 Task Specification

In our experiments, we use two OpenAI classical control tasks: CartPole and Pendulum [Brockman et al., 2016], and two MuJoCo tasks: Hopper and Walker [Todorov et al., 2012b]. Since these tasks are deterministic and the notion of risk-sensitive decision-making is closely related to the uncertainty in the system, we incorporate stochasticity into the original implementations of these tasks, as described below.

In the OpenAI classical control tasks, we inject stochasticity to the system by adding noise to the actions, which in turn adds noise to both the reward function and the transitions. In the MuJoCo tasks, we first learn a policy by running a RL agent with TRPO [Schulman et al., 2015] on the risk-neutral version of the original implementation, and then add noise to the costs as a function of the occupancy measure of the learned policy (see Appendix E for details).

CartPole: Our CartPole task is based on the CartPole-v1 environment in Brockman et al. [2016] in which at each step the agent can choose one of the two actions: either applying the force F_x (action $a = 1$) or the force $-F_x$ (action $a = 0$). In our implementation, if the agent selects action $a = 0$, it applies the force $-F_x$ w.p. 0.8 and the force $-K\,F_x$, where K is an integer uniformly drawn from $\{0, \ldots, 8\}$, w.p. 0.2.

Pendulum: Our Pendulum task is based on the Pendulum-v0 environment in Brockman et al. [2016] in which the action space consists of 3 different torque values $\{-2, 0, 2\}$ that are applied to the pendulum. In our implementation, we first extend the number of torque values to 5, and then when the agent selects an action with the torque value $u \in \{-2, -1, 0, 1, 2\}$, w.p. 0.2, the value u is multiplied by $(1 + |Z|)$, where $Z \sim \mathcal{N}(0, 1)$ is a standard Gaussian random variable truncated to be bounded between -3 and 3.

Hopper: Our Hopper task is based on Hopper-v1, a physics-based continuous control task simulated with MuJoCo [Todorov et al., 2012b], which consists of an 11-dimensional observation space, a 3-dimensional action space, a deterministic reward function $r(s, a)$, and deterministic dynamics. The goal in Hopper is to make a one-legged robot hop forward as fast as possible.

Walker2d-v1: Our Walker task is based on Walker2d-v1, a physics-based continuous control task simulated with MuJoCo [Todorov et al., 2012b], which consists of a 17-dimensional observation space, a 6-dimensional action space, a deterministic reward function $r(s, a)$, and deterministic dynamics. The goal in Walker is to make a bipedal robot walk forward as fast as possible.

6.2 Experimental Setup

In the OpenAI classical control tasks, the risk-sensitive objective is set to $\rho_\alpha^\lambda = \text{Mean} + 0.5 \times \text{CVaR}_{0.3}$, which means that the risk-sensitive parameters have been set to $\alpha = 0.3$ and $\lambda = 0.5$. We set the expert's policy to that learned by the CVaR policy gradient algorithm of Tamar et al. [2015b], which is the standard REINFORCE algorithm [Williams, 1992] adapted to the CVaR criteria. In the MuJoCo tasks, the risk-sensitive objective is set to $\rho_\alpha^\lambda = \text{Mean} + 0.05 \times \text{CVaR}_{0.3}$ and the expert policy is learned by TRPO on the standard (deterministic) implementations of these problems.

In our experiments, we use two different policy (gradient) optimization algorithms for the policy step of RAIL and our JS-RS-GAIL and W-RS-GAIL algorithms: **1)** the REINFORCE policy gradient algorithm in Tamar et al. [2015b], and **2)** the algorithm implemented in Santara et al. [2017a], which is an extension of TRPO (using KL-constrained gradient step) to risk-sensitive policy optimization. Note that the policy step of GAIL uses TRPO [Schulman et al., 2015]. In the OpenAI classical control tasks, using either of these two algorithms in the policy step of RAIL and JS-RS-GAIL did not change their performance. However, in the CartPole task, we did not obtain good results using the REINFORCE policy gradient algorithm in Tamar et al. [2015b] for the policy step of W-RS-GAIL, and thus, we conducted the experiments with the extended TRPO. In the MuJoCo tasks, we only obtained good results with the extended TRPO for all the algorithms. We conjecture that this is due to the high variance of REINFORCE gradient estimate.

We use 100 expert trajectories to train all the algorithms, which is a higher number than that used in the experiments of the GAIL paper [Ho and Ermon, 2016a] (between 1 and 20 trajectories). This is normal because the risk-sensitive algorithms require more samples than their risk-neutral counterparts, particularly those that optimize tail-related risk criteria, such as VaR and CVaR. Sample efficiency is one of the most important problems of the tail-related risk-sensitive optimization algorithms and has been reported in the literature (Chow and Ghavamzadeh [2014], Tamar et al. [2015b], Chow et al. [2018]), and it is mainly due to the fact that these algorithms require to learn a tail-related quantity, often VaR, for which only the trajectories whose return belongs to the tail can be used. There have been work to address this issue and to use the trajectories whose return does not belong to the tail to

learn about tail-related quantities (Bardou et al. [2009], Tamar et al. [2015b]), but this is still an open problem and we do not use any of these techniques in this paper.

We pre-train the risk-sensitive algorithms RAIL and JS-RS-GAIL with 100 iterations of GAIL. As it was noted by Tamar et al. [2015b], pre-training risk-sensitive policy gradient algorithms with their risk-neutral counterpart is a useful technique to avoid getting stuck in local minima. In these algorithms, we use the same network architecture as in Ho and Ermon [2016a] and Santara et al. [2017a], which consists of 2 hidden layers with 32 units each in the OpenAI tasks, with 100 units each in the MuJoCo tasks, and, tanh activations, for both the policy and discriminator networks. At each iteration, all the algorithms are given the same amount of interaction with the environment by sampling 100 trajectories. Our algorithms and RAIL use 1 update step for both the generator and discriminator at each iteration, while GAIL uses 3 update steps for the generator and 1 for the discriminator. We found these hyper-parameters by grid search for each algorithm.

We do not pre-train W-RS-GAIL, as we did not observe any improvement due to pre-training, but train it with the same number of iterations (pre-train + train) as the other algorithms. We use a more complex architecture for both the policy and discriminator networks in the Wasserstein-based algorithms: W-GAIL[1] and W-RS-GAIL. This architecture consists of 3 hidden layers with 64, 64, and 32 units, tanh activation, and clipping thresholds of -0.05 and 0.05.

6.3 Experimental Results

In this section, we compare the performance of our algorithm JS-RS-GAIL with RAIL and GAIL (Tables 1–4), and our algorithm W-RS-GAIL with W-GAIL (Table 5) in terms of their mean, VaR_α, CVaR_α, and more importantly ρ_α^λ, which is the main target of our risk-sensitive algorithms. In all of these algorithms, we aim at minimizing the sum of the costs (the lower the better). We also report the performance of the expert and a random policy for reference.

We report the performance of the algorithms in terms of each criterion for the OpenAI control tasks in Table 1. For each task, we run each algorithm for a fixed number of iterations, 200 for CartPole and 300 for Pendulum (after 100 pre-training iterations). After that we run the algorithm for another 100 iterations and evaluate the performance of each of these 100 policies by generating 300 trajectories from that policy. We then average each performance criterion over the 100 policies. We average over 100 policies generated after our algorithms stop to show how well each algorithm converges in terms of each performance criterion. We repeat this process for 10 random seeds and take the average. We then report the average and 95% confidence interval *(empirical mean*$\pm$ $1.96 \times$ *empirical standard deviation*$/\sqrt{n=10}$*)*.

Table 2 contains the exact same results for CartPole and Pendulum, except this time we first average each performance criterion over the top 10 policies of the last 100 policies (instead of averaging over the last 100 policies). Note that the top 10 policies are different for each performance criterion.

The results of Tables 1 and 2 show that JS-RS-GAIL achieves the best performance (compared to GAIL and RAIL) in terms of the risk-sensitive criteria, in particular ρ_α^λ. This advantage becomes statistically significant when we average over the top 10 policies (see Table 2). We conjecture that if we average over more (than 10) random seeds, we will see statistically significant advantage for JS-RS-GAIL even when we average over the last 100 iterations. Note that in Pendulum, none of the algorithms achieve the expert's performance, but they perform better than the random policy. This means that they are learning and expert's performance can be achieved with more iterations and parameter tuning.

Tables 3 and 4 contain the exact same results as in Tables 1 and 2 for the MuJoCo tasks: Hopper and Walker. Similar to the OpenAI classical control problems, here JS-RS-GAIL also achieves the best performance in terms of the risk-sensitive criteria and the advantage becomes statistically significant when we average over the top 10 policies (see Table 4).

Table 5 shows the performance of W-RS-GAIL and compares it with that of W-GAIL in the CartPole problem. We do not compare the Wasserstein-based algorithms with the JS-based ones because they are solving different optimization problems. However, our results indicate that the JS-based algorithms perform better than their Wasserstein-based counterparts in terms of the relevant criteria (mean for GAIL and ρ_α^λ for RS-GAIL algorithms). We conjecture that the reason is the small size of the networks. When we use Wasserstein with a small network, we end up having a very limited representation power due to clipping the weights at certain thresholds in order to maintain the Lipschitz smoothness of the network. This is why we think that the Wasserstein-based algorithms could perform better in more complex problems that require more complex networks. Verifying this conjecture requires more experiments that we leave as a future work.

7 Conclusions and Future Work

In this paper, we first formulated a risk-sensitive imitation learning setting in which the agent's goal is to have

[1]Note that the W-GAIL algorithm in our experiments is just the Wasserstein version of GAIL and is simpler than InfoGAIL [Li et al., 2017].

Table 1: Performance of the policies learned by the algorithms for $\alpha = 0.3$ and $\lambda = 0.5$. Results are averaged over the last 100 iterations and 10 random seeds.

Criteria	Random	Expert	GAIL	RAIL	JS-RS-GAIL	Random	Expert	GAIL	RAIL	JS-RS-GAIL
	CartPole					**Pendulum**				
Mean	-12	-333	-296±12	-315±3	-319±3	1410	162	907±41	1150±81	908±89
VaR_α	-3	-301	-151±37	-193±19	-231±16	1760	341	1485±44	1517±59	1409±60
CVaR_α	-2	-294	-109±36	-163±19	-208±17	1812	401	1495±46	1527±56	1419±58
ρ_α^λ	-13	-479	-350±31	-398±11	-425±11	2296	362	1656±63	1973±106	1616±109

Table 2: Performance of the policies learned by the algorithms for $\alpha = 0.3$ and $\lambda = 0.5$. Results are averaged over the top 10 policies of the last 100 iterations and 10 random seeds.

Criteria	Random	Expert	GAIL	RAIL	JS-RS-GAIL	Random	Expert	GAIL	RAIL	JS-RS-GAIL
	CartPole					**Pendulum**				
Mean	-12	-333	-326±3	-319±6	-325±4	1410	162	656±54	961±135	436±84
VaR_α	-3	-301	-269±6	-249±30	-282±7	1760	341	1403±27	1325±74	1152±69
CVaR_α	-2	-294	-258±8	-229±32	-278±8	1812	401	1411±26	1335±81	1175±85
ρ_α^λ	-13	-479	-451±6	-434±21	-465±6	2296	362	1362±68	1629±188	1023±138

Table 3: Performance of the policies learned by the algorithms for $\alpha = 0.3$ and $\lambda = 0.05$. Results are averaged over the last 100 iterations and 10 random seeds.

Criteria	Random	Expert	GAIL	RAIL	JS-RS-GAIL	Random	Expert	GAIL	RAIL	JS-RS-GAIL
	Hopper					**Walker**				
Mean	-10	-6096	-5428±191	-5638±220	-5622±198	-1	-7651	-6542±252	-6894±241	-6921±230
VaR_α	-5	-6129	-5576±228	-5621±202	-5709±210	0	-7875	-6674±187	-6605±201	-6702±199
CVaR_α	- 3	-5590	-4913±231	-5141±215	-5202±222	0	-6440	-5341±352	-6012±215	-6111±202
ρ_α^λ	-10	-6375	-5673±202	-5895±231	-5882±209	1	-7973	-6809±269	-7194±251	-7226±239

Table 4: Performance of the policies learned by the algorithms for $\alpha = 0.3$ and $\lambda = 0.05$. Results are averaged over the top 10 policies of the last 100 iterations and 10 random seeds.

Criteria	Random	Expert	GAIL	RAIL	JS-RS-GAIL	Random	Expert	GAIL	RAIL	JS-RS-GAIL
	Hopper					**Walker**				
Mean	-10	-6096	-5743±145	-6049±60	-6032±51	-1	-7651	-7221±214	-7405±65	-7621±63
VaR_α	-5	-6129	-6130±91	-6268±10	-6355±13	0	-7875	-7377±133	-7535±30	-7925±30
CVaR_α	-3	-5590	-5361±226	-5541±96	-5595±83	0	-6440	-5590±335	-6172±136	-6451±129
ρ_α^λ	-10	-6375	-6011±156	-6340±64	-6325±55	-1	-7973	-7527±230	-7714±72	-7953±70

Table 5: Performance of the policies learned by the algorithms for $\alpha = 0.3$ and $\lambda = 0.5$. Results are averaged over the last 100 iterations and 10 random seeds (W-GAIL1 and W-RS-GAIL1), as well as over the top 10 policies of the last 100 iterations and 10 random seeds (W-GAIL2 and W-RS-GAIL2).

Criteria	Random	Expert	W-GAIL1	W-RS-GAIL1	W-GAIL2	W-RS-GAIL2
			CartPole			
Mean	-12	-333	-275±8	-282±8	-284±5	-309±4
VaR_α	-3	-301	-43±26	-89±32	-71±22	-171±31
CVaR_α	-2	-294	-30±14	-59±27	-60±17	-149±31
ρ_α^λ	-13	-479	-290±14	-312±12	-314±9	-384±10

a risk profile as good as the expert's. We then derived a GAIL-like optimization problem for our formulation, which we termed it risk-sensitive GAIL (RS-GAIL). We proposed two risk-sensitive generative adversarial imitation learning algorithms based on two variations of RS-GAIL that match the agent's and the expert's risk profiles w.r.t. Jensen-Shannon (JS) divergence and Wasserstein distance. We experimented with our algorithms and compared their performance with that of GAIL [Ho and Ermon, 2016a] and RAIL [Santara et al., 2017a] in two MuJoCo and two OpenAI control tasks.

Future directions include **1)** extending our results to other popular risk measures, such as expected exponential utility and the more general class of coherent risk measures, **2)** investigating other risk-sensitive imitation learning settings, especially those in which the agent can tune its risk profile w.r.t. the expert, e.g., being a more risk averse/seeking version of the expert, **3)** reducing variance of the gradient estimate in extended TRPO, and **4)** more experiments, particularly with our Wasserstein-based algorithm, in more complex problems and in problems with intrinsic stochasticity.

References

E. Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

O. Bardou, N. Frikha, and G. Pagès. Computing VaR and CVaR using stochastic approximation and adaptive unconstrained importance sampling. *Monte Carlo Methods and Applications*, 15(3):173–210, 2009.

D. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

V. Borkar. A sensitivity formula for the risk-sensitive cost and the actor-critic algorithm. *Systems & Control Letters*, 44:339–346, 2001.

J. Borwein. A very complicated proof of the minimax theorem. *Minimax Theory and its Applications*, 1 (1):21–27, 2016.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI gym. *arXiv:1606.01540*, 2016.

Y. Chow and M. Ghavamzadeh. Algorithms for CVaR optimization in MDPs. In *Advances in Neural Information Processing Systems*, pages 3509–3517, 2014.

Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research*, 18(167):1–51, 2018.

C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 49–58, 2016.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Proceedings of Advances in neural information processing systems*, pages 2672–2680, 2014.

J. Ho and S. Ermon. Generative adversarial imitation learning. In *Proceedings of Advances in Neural Information Processing Systems*, pages 4565–4573, 2016a.

J. Ho and S. Ermon. Gail github. `https://github.com/openai/imitation`, 2016b.

R. Howard and J. Matheson. Risk sensitive Markov decision processes. *Management Science*, 18(7):356–369, 1972.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

S. Levine and V. Koltun. Continuous inverse optimal control with locally optimal examples. In *Proceedings of the 29th International Conference on Machine Learning*, pages 41–48, 2012.

Y. Li, J. Song, and S. Ermon. InfoGAIL: Interpretable imitation learning from visual demonstrations. In *Proceedings of Advances in Neural Information Processing Systems*, 2017.

A. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 663–670, 2000.

X. Nguyen, M. Wainwright, and M. Jordan. On surrogate loss functions and f-divergences. *The Annals of Statistics*, 37(2):876–904, 2009.

A. Pichler. Evaluations of risk measures for different probability measures. *SIAM Journal on Optimization*, 23(1):530–551, 2013.

D. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

L. Prashanth and M. Ghavamzadeh. Actor-critic algorithms for risk-sensitive MDPs. In *Proceedings of Advances in Neural Information Processing Systems 26*, pages 252–260, 2013.

L. Prashanth and M. Ghavamzadeh. Variance-constrained actor-critic algorithms for discounted and average reward mdps. *Machine Learning Journal*, 105(3):367–417, 2016.

R. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 26:1443–1471, 2002.

S. Ross and D. Bagnell. Efficient reductions for imitation learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 661–668, 2010.

S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

A. Santara, A. Naik, B. Ravindran, D. Das, D. Mudigere, S. Avancha, and B. Kaul. Rail: Risk-averse imitation learning. *arXiv:1707.06658*, 2017a.

A. Santara, A. Naik, B. Ravindran, D. Das, D. Mudigere, S. Avancha, and B. Kaul. Rail github. `https://github.com/Santara/RAIL`, 2017b.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory, Second Edition*. Society for Industrial and Applied Mathematics, 2014.

S. Singh, J. Lacotte, A. Majumdar, and M. Pavone. Risk-sensitive inverse reinforcement learning via semi- and non-parametric methods. *Int. Journal of Robotics Research*, 2018.

M. Sobel. The variance of discounted Markov decision processes. *Applied Probability*, pages 794–802, 1982.

R. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of Advances in Neural Information Processing Systems*, pages 1057–1063, 2000.

A. Tamar, D. D. Castro, and S. Mannor. Policy gradients with variance related risk criteria. In *Proceedings of the Twenty-Ninth International Conference on Machine Learning*, pages 387–396, 2012.

A. Tamar, Y. Chow, M. Ghavamzadeh, and S. Mannor. Policy gradient for coherent risk measures. In *Advances in Neural Information Processing Systems*, pages 1468–1476, 2015a.

A. Tamar, Y. Glassner, and S. Mannor. Optimizing the CVaR via sampling. In *AAAI Conference on Artificial Intelligence*, 2015b.

A. Tamar, Y. Chow, M. Ghavamzadeh, and S. Mannor. Sequential decision-making with coherent risk. *IEEE Transaction on Automatic Control*, 62(7):3323–3338, 2017.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *Proceedings of IROS*, pages 5026–5033, 2012a.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012b.

R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

B. Ziebart, A. Maas, J. Bagnell, and A. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of AAAI*, pages 1433–1438, 2008.

B. Ziebart, J. Bagnell, and A. Dey. Modeling interaction via the principle of maximum causal entropy. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1255–1262, 2010.