RAIL: Risk-Averse Imitation Learning

Extended Abstract

Anirban Santara, Abhishek Naik, Balaraman Ravindran
Dipankar Das, Dheevatsa Mudigere, Sasikanth Avancha, Bharat Kaul*

ABSTRACT

Imitation learning algorithms learn viable policies by imitating an expert's behavior when reward signals are not available. Generative Adversarial Imitation Learning (GAIL) is a state-of-the-art algorithm for learning policies when the expert's behavior is available as a fixed set of trajectories. We evaluate in terms of the expert's cost function and observe that the distribution of trajectory-costs is often more heavy-tailed for GAIL-agents than the expert at a number of benchmark continuous-control tasks. Thus, high-cost trajectories, corresponding to tail-end events of catastrophic failure, are more likely to be encountered by the GAIL-agents than the expert. This makes the reliability of GAIL-agents questionable when it comes to deployment in risk-sensitive applications like robotic surgery and autonomous driving. In this work, we aim to minimize the occurrence of tail-end events by minimizing tail risk within the GAIL framework. We quantify tail risk by the Conditional-Value-at-Risk ($CVaR$) of trajectories and develop the Risk-Averse Imitation Learning (RAIL) algorithm. We observe that the policies learned with RAIL show lower tail-end risk than those of vanilla GAIL. Thus, the proposed RAIL algorithm appears as a potent alternative to GAIL for improved reliability in risk-sensitive applications.

KEYWORDS

Reinforcement Learning; Imitation Learning; Risk Minimization; Conditional-Value-at-Risk; Reliability

ACM Reference Format:
Anirban Santara, Abhishek Naik, Balaraman Ravindran and Dipankar Das, Dheevatsa Mudigere, Sasikanth Avancha, Bharat Kaul. 2018. RAIL: Risk-Averse Imitation Learning. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 2 pages.

1 INTRODUCTION

In this paper, we study the reliability of imitation learning algorithms when it comes to learning solely from a fixed set of trajectories demonstrated by an expert with no interaction between the agent and expert during training. Risk sensitivity is integral to human learning, but much of the literature on imitation learning has been developed with average-case performance at the centre, overlooking tail-end events. The Generative Adversarial Imitation Learning (GAIL) algorithm [2] provides state-of-the-art performance at several benchmark control tasks, including those in Table 1. Additionally, this method is not prone to the issue of compounding error and it is also scalable to large environments. However, we studied the distributions of trajectory-costs (according to the expert's cost function) for the GAIL agents and experts at different control tasks and observed that the distributions for GAIL are more heavy-tailed than the expert (see Figure 1), where the tail corresponds to occurrences of high trajectory-costs. Since high trajectory-costs may correspond to events of catastrophic failure, GAIL agents are not reliable in risk-sensitive applications.

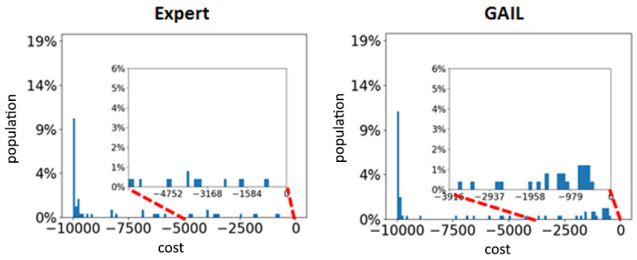

Figure 1: Histograms of the costs of 250 trajectories generated by the expert and GAIL agents for the task Humanoid-v1, from OpenAI Gym. The inset shows a zoomed-in view of the tail of the distribution (the region beyond 2σ of the mean). The GAIL agent produces tails heavier than the expert, which makes its reliability questionable for deployment in risk-sensitive applications.

In order to quantify tail risk, we use Conditional-Value-at-Risk ($CVaR$) [3]. The heavier the tail, the higher the value of $CVaR$. Chow et al. [1] developed policy gradient and actor-critic algorithms for mean-$CVaR$ optimization for learning policies in the classic RL setting. We take inspiration from this work and a) formulate the Risk-Averse Imitation Learning (RAIL) algorithm which optimizes $CVaR$ in addition to the original GAIL objective; b) evaluate RAIL at a number of benchmark control tasks and demonstrate that it obtains policies with lesser tail risk at test time than GAIL.

2 PROPOSED FRAMEWORK

GAIL optimizes the following objective:

$$\underset{\pi}{argmin} \max_{\mathcal{D}} \mathbb{E}_{\pi}[log(\mathcal{D})] + \mathbb{E}_{\pi_E}[log(1-\mathcal{D})] - H(\pi) \qquad (1)$$

where, the agent's policy, $\pi : \mathcal{S} \rightarrow \mathcal{A}$, acts as a *generator* of state-action pairs and $\mathcal{D} : \mathcal{S} \times \mathcal{A} \rightarrow (0, 1)$ is a *discriminative* binary classifier which predicts the likelihood of a given a state-action pair having originated from the generator. We define the trajectory-cost variable $\mathcal{R}^{\pi}(\xi|c(\mathcal{D}))$ in the context of GAIL as:

*AS and AN contributed equally as a part of their internship at Intel Labs, India.
AS (anirban_santara@iitkgp.ac.in) is with Indian Institute of Technology Kharagpur.
AN and BR are with the Department of CSE and the Robert Bosch Centre for Data Science and AI at Indian Institute of Technology Madras.
DD, DM, SA and BK are with Parallel Computing Lab - Intel Labs, India

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

$$\mathcal{R}^{\pi}(\xi|c(\mathcal{D})) = \sum_{t=0}^{L_{\xi}-1} \gamma^{t} c(\mathcal{D}(s_t, a_t)) \quad (2)$$

where $c(\cdot)$ is an order-preserving function. Following [3], the objective of $CVaR$ optimization of $\mathcal{R}^{\pi}(\xi|c(\mathcal{D}))$ is defined as:

$$\min_{\pi, \nu} \max_{c} \; H_{\alpha}(\mathcal{R}^{\pi}(\xi|c(\mathcal{D})), \nu) \quad (3)$$

where $H_{\alpha}(Z, \nu)$, for any random variable Z, is given by:

$$H_{\alpha}(Z, \nu) \triangleq \{\nu + \frac{1}{1-\alpha}\mathbb{E}\left[(Z-\nu)^{+}\right]\}; \; (x)^{+} = max(x, 0) \quad (4)$$

Integrating this with the GAIL objective of equation 1, we have:

$$\min_{\pi, \nu} \max_{\mathcal{D}} \Big\{ -H(\pi) + \mathbb{E}_{\pi}[log(\mathcal{D})] + \mathbb{E}_{\pi_E}[log(1-\mathcal{D}(s,a))] + \lambda_{CVaR} \, H_{\alpha}(\mathcal{R}^{\pi}(\xi|c(\mathcal{D})), \nu) \Big\} \quad (5)$$

3 EVALUATION

We compare the tail risk of policies learned by GAIL and RAIL for a set of continuous control tasks listed in Table 1 that were simulated in MuJoCo [5]. Given an agent A's policy π_A we roll out $N = 50$ trajectories from it and estimate the metrics in Table 1 for comparison. Following [2], we model the generator (policy), discriminator and value function with multi-layer perceptrons of the architecture: `observationDim - fc_100 - tanh - fc_100 - tanh - outDim`. If f is the tail risk metric, in order to compare the tail risk of an agent with respect to the expert, E, we define percentage-relative f as follows:

$$f(A|E) = 100 \times \frac{f(E) - f(A)}{|f(E)|}\% \quad (6)$$

The higher these numbers, the lesser is the tail risk of agent A. We define Gain in Reliability (GR) as the difference in percentage relative tail risk between RAIL and GAIL agents.

$$GR\text{-}f = f(RAIL|E) - f(GAIL|E) \quad (7)$$

Table 1: Values of percentage relative tail risk measures and gains in reliability on using RAIL over GAIL for the different continuous control tasks. RAIL shows a remarkable improvement over GAIL in both the metrics.

Environment	Dimensionality		$GR\text{-}VaR_{0.9}$ (%)	$GR\text{-}CVaR_{0.9}$ (%)
	Obs	Action		
Reacher-v1	11	2	38.61	60.57
Hopper-v1	11	3	52.94	89.00
HalfCheetah-v1	17	6	13.46	21.60
Walker-v1	17	6	1.66	25.13
Humanoid-v1	376	17	67.19	72.78

4 DISCUSSION

We make the following observations about the performance of RAIL after detailed experimentation (please refer to the full paper [4] for an extended discussion)[1] :

[1] All code and hyperparameters available at `https://github.com/Santara/RAIL`

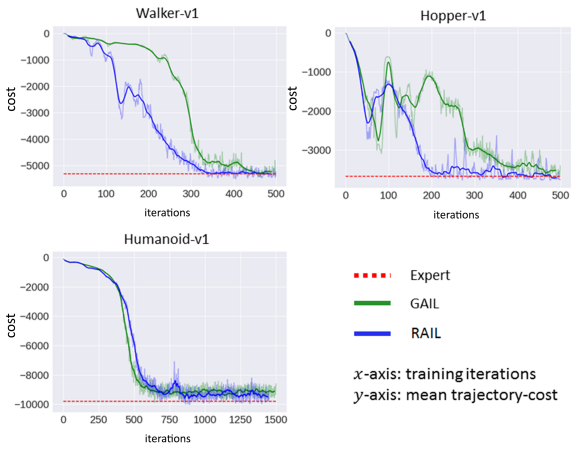

Figure 2: Convergence of mean trajectory-cost. RAIL converges almost as fast as GAIL at all the continuous-control tasks in discussion, and at times, even faster.

- RAIL obtains superior performance than GAIL at both tail risk measures – $VaR_{0.9}$ and $CVaR_{0.9}$ – across a wide range of continuous-control tasks, without increasing the sample complexity or degrading the mean performance of GAIL.
- The applicability of RAIL is not limited to environments in which the distribution of trajectory-cost is heavy-tailed for GAIL. In the absence of a heavy tail, minimization of $CVaR_{\alpha}$ of the trajectory cost aids in learning better policies by contributing to the minimization of the mean and standard deviation of trajectory cost [3, 4]. Thus we can use RAIL instead of GAIL irrespective of whether the distribution of trajectory costs is heavy-tailed for GAIL or not.
- RAIL converges almost as fast as GAIL at all the continuous-control tasks in discussion, and at times, even faster (see Figure 2 for some sample learning curves).
- Scalability is one of the salient features of GAIL. The success of RAIL in learning a viable policy for Humanoid-v1 suggests that RAIL preserves the scalability of GAIL.

In conclusion, our study establishes that RAIL is a superior choice than GAIL for learning low-risk policies via imitation learning in complex, risk-sensitive environments. We plan to test RAIL on fielded robotic applications in the future.

REFERENCES

[1] Yinlam Chow and Mohammad Ghavamzadeh. 2014. Algorithms for CVaR optimization in MDPs. In *Advances in neural information processing systems*. 3509–3517.
[2] Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*. 4565–4573.
[3] R Tyrrell Rockafellar and Stanislav Uryasev. 2000. Optimization of conditional value-at-risk. *Journal of risk* 2 (2000), 21–42.
[4] Anirban Santara, Abhishek Naik, Balaraman Ravindran, Dipankar Das, Dheevatsa Mudigere, Sasikanth Avancha, and Bharat Kaul. 2017. RAIL: Risk-Averse Imitation Learning. *arXiv preprint arXiv:1707.06658* (2017).
[5] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. MuJoCo: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 5026–5033.