Risk-sensitive deep reinforcement learning

Gilwoo Lee
gilwoo@cs.uw.edu

Siddhartha S. Srinivasa
siddh@cs.uw.edu

1 Introduction

Adversarial Reinforcement Learning takes the following differential game-theoretic approach. It utilizes the concept of agent and disturber, and solves the following minimax game for u and w:

$$V(s) = \max_a \min_w \mathbb{E}\Big[J(s)\Big]$$

where a is the action of the agent and w is the disturbance caused by the disturber. This system is typically trained by an iterative procedure for maximizing the reward function for the agent and minimizing it for the disturber. Several approaches[1, 2, 3] have shown that such an approach results in a robust RL policy.

Another branch of RL which aims to discover robust policy w.r.t. model uncertainty and disturbance is risk-sensitive RL. One of the commonly used utility functions in Risk-Sensitive RL is an exponential reward function. Using $\frac{1}{\beta}\log(\mathbb{E}[\exp(\beta J)])$ instead of $\mathbb{E}[J]$ lets us take into account higher order moments of the reward function. Taking Taylor expansion, we get

$$\max_\theta \frac{1}{\beta}\mathbb{E}\Big[\exp(\beta J)\Big] \approx \max_\theta \mathbb{E}[J] + \beta\frac{var(J)}{2} + o(\beta^2 J^3)$$

Risk is penalized if $\beta < 0$ and encouraged if $\beta > 0$.

Consider the risk-aversive case, $\beta < 0$. If we put a disturber in this exponential utility function, a disturber would not only play the role of minimizing $\mathbb{E}[J]$ but also increase the variance of J, i.e.,

$$V(s) = \max_a \min_w \frac{1}{\beta}\mathbb{E}\Big[\exp(\beta J)\Big] \approx \max_a \min_w \mathbb{E}[J] + \beta\frac{var(J)}{2}$$

This implies that by using the exponential utility function for the Adversarial Reinforcement Learning, we train the disturber such that it not only reduces the expected total reward but also increase the variance, and train the agent to learn to minimize variance against such a disturber while maximizing the expected reward.

1.1 Risk-seeking to Risk-aversive

To encourage exploration during early training, one may consider starting from $\beta > 0$ and slowly tuning β to be positive. When $\beta > 0$, the role of disturber would be to reduce the variance, so it would only hinder the learning.

1.2 Helper to Disturber

An RL agent faces two main challenges:

1. the agent does not know the desired trajectory that maximizes rewards
2. the agenet does not know the desired actions that leads to the desired trajectory

Typical RL algorithms aim to discover both simultaneously. To aid learning in the early phase, one may consider a *fully-actuated* helper who helps the agent to quickly discover the directions which maximize reward. For example, when an agent learns to ride a bicycle, the agent needs to spend a lot of time discovering whether the bike needs to move forward, as well as the particular pedalling actions that makes the bike move forward. Instead, a fully actuated helper can first aid in discovering that moving forward is important, and gradually decrease its actuation to let the agent discover how to move forward.

References

[1] J. Morimoto and K. Doya, "Robust reinforcement learning," *Neural computation*, vol. 17, no. 2, pp. 335–359, 2005.

[2] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," *arXiv preprint arXiv:1703.02702*, 2017.

[3] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary, "Robust deep reinforcement learning with adversarial attacks," *arXiv preprint arXiv:1712.03632*, 2017.