Reward Elicitation

Soheil

Introduction

- Inverse Reinforcement Learning and Apprenticeship Learning
- Bayesian IRL
- Risk in RL

Background

- MDP definition
- $< S, A, P, R, \gamma >$
- policy
- value function
- linear representation of reward function
- μ (expected feature counts)
- D: demonstrations
- Bayesian IRL
- applying Bayes theory

Markov Decision Processes

Bayesian IRL

Inverse Reinforcement Learning (IRL) is the problem of finding a reward function that can represent the intention of an expert who has provided the MDP with demonstrations. In Bayesian IRL this problem is solved with Bayesian probability approach. The goal in Bayesian IRL is to find the posterior probability over the reward function [2].

In BIRL we assume that the policy is stationary, and each sample is in fact independent of the previous one.

$$\Pr(D|R) = \prod_{(s,a)\in D} \Pr((s,a)|R)$$

In addition, we model the likelihood of the expert being at (s,a) in terms of an exponential distribution function of Q^*. This exponential function is called softmax.

$$\Pr((s,a)|\mathbf{R}) = \frac{1}{Z} e^{cQ^*(s,a,\mathbf{R})}$$

Therefore

$$\begin{aligned} \Pr(D|\mathbf{R}) &= \prod_{(s,a)\in D} \frac{e^{cQ^*(s,a,\mathbf{R})}}{\sum_{a'\in\mathcal{A}} e^{cQ^*(s,a',\mathbf{R})}} \\ &= \frac{1}{Z} e^{cE(D,\mathbf{R})} \end{aligned}$$

where $E(D,\mathbf{R}) = \sum_i Q^*(s_i,a_i,\mathbf{R})$

Now, by applying the Bayes rule we can compute the posterior probability of reward function

$$\begin{aligned} \Pr(\mathbf{R}|D) &= \frac{\Pr(D|\mathbf{R})\Pr(\mathbf{R})}{\Pr(D)} \\ &= \frac{1}{Z'} e^{cE(D,\mathbf{R})} \Pr(\mathbf{R}) \end{aligned}$$

Z' is a normalizing factor. Computing Z' is not an easy task. However, we can address this problem using sampling algorithms that can deal with the ratio of the densities at two points instead of the absolute value of the density at a point.

Priors

We have different options for priors; we can use a uniform distribution over space $[R_{min}, R_{max}]$. Some MDPs have parsimonious reward function, with sparse reward structure. In these situations Gaussian and Laplacian prior should work better. In some situations where we are planning to reach to a goal, a Beta distribution can be used.

Robust Optimization for MDPs

- V@R and CV@R
- CV@R motivation
- Robust MDP
- LP representation

Let X be a random variable, and the cumulative distribution function of X is defined as $F(x) = \Pr(X \leq x)$. The *value-at-risk* (VaR) at $\alpha \in (0, 1)$ confidence level is the $1 - \alpha$ quantile of X

$$\mathrm{VaR}_\alpha(X) = \min\{t | \Pr(X \leq x) \leq t\}$$

$\mathrm{VaR}_\alpha(X)$ cannot fully representing the real worst case scenario, which might happen somewhere at the end of a long tail, since it neglects the long tails with low probabilities. In addition, $\mathrm{VaR}_\alpha(X)$ is not a convex function. In order to fix these problems, the *average value-at-risk* and *conditional value-at-risk* were introduced.

The *conditional value-at-risk* at confidence level $\alpha \in (0, 1)$ is defined as:

$$\mathrm{CVaR}_\alpha(X) = \min_t\{t + \frac{1}{\alpha}\mathbb{E}[(X - t)^-]\}$$

where $(x)^- = -\max(-x, 0)$ is the negative part of x. This definition is true for both non-atomic and atomic probability distribution. However, in case of non-atomic distributions, we can simplify the formulation to

$$\mathrm{CVaR}_\alpha(X) = \mathbb{E}[X | X < \mathrm{VaR}_\alpha(X)]$$

This representation, unlike the previous one, is not coherent. The dual of the coherent representation can be formulated in terms of linear programs as:

$$\begin{array}{ll} \max_{t,y} & t + \frac{1}{\alpha} p^\top y \\ s.t. & y \leq X - \mathbf{1}t \\ & y \leq 0 \end{array}$$

Robust MDP

Let $\rho(u, r) = r^\top u$ denote the total discounted return for the MDP M. When we are looking for an optimal policy, we simply maximize the return over all policies:

$$\begin{array}{ll} \max_{\mathbf{u}} & \mathbf{r}^\top \mathbf{u} \\ & \mathbf{A}^\top \mathbf{u} = \alpha^\top \\ & \mathbf{u} \geq 0 \end{array}$$

But, we are dealing with an uncertain reward function. In this situation, we are usually interested into figuring out how good we can do in the worst-case scenario. This leads to an adversarial situation, where one agent is trying to maximize the return, and the advisory is doing the opposite by minimizing the return.

$$\begin{array}{ll} \max_{\mathbf{u}} \min_{\mathbf{r}} & \mathbf{r}^\top \mathbf{u} \\ & \mathbf{A}^\top \mathbf{u} = \alpha^\top \\ & \mathbf{u} \geq 0 \end{array}$$

The best worst-case can be a good baseline, but it is too conservative. A less conservative alternative can be $\text{CVaR}_\alpha(X)$. So, the risk-sensitive discounted-return problem we are solving can be represented by [1]:

$$\begin{array}{ll} \max_u & \text{CVaR}_\alpha(\rho(u,r)) \\ & A^\top u = \alpha^\top \\ & u \geq 0 \end{array}$$

Adding the $\text{CVaR}_\alpha(X)$ guarantees the robustness, and at the same time it avoids the worst-case, which might be very unlikely to happen.

Using the $\text{CVaR}_\alpha(\rho(u,r))$ term with it's dual representation, we can expand the linear program and have

$$\begin{array}{ll} \max\limits_{u,t,y} & t + \frac{1}{\alpha} p^\top y \\ s.t. & y \leq \mathbf{R}^\top u - \mathbf{1}t \\ & y \leq 0 \\ & \mathbf{A}^\top \mathbf{u} = \alpha^\top \\ & \mathbf{u} \geq 0 \end{array}$$

where $\mathbf{R}$ is a matrix consist of column vectors $[\mathbf{r_1}, \mathbf{r_2}, ..., \mathbf{r_k}]$, and $\mathbf{r_i} \sim P(\mathbf{R}|D)$, which means $\mathbf{r}$ is drawn from the posterior probability distribution over reward function.

Questions

References

[1] Yinlam Chow, Aviv Tamar, Shie Mannor, and Marco Pavone. Risk-Sensitive and Robust Decision-Making: a CVaR Optimization Approach. pages 1–21, jun 2015.

[2] Deepak Ramachandran and Eyal Amir. Bayesian Inverse Reinforcement Learning. *IJCAI International Joint Conference on Artificial Intelligence*, 10(5):2586–2591, 2007.