Literature review: Reinforcement Learning

Daniel Hernandez

1 Appendix

subfiles

$$U(\theta) = \sum_{\tau} P(\tau;\theta)R(\tau) \tag{1}$$

Takin the gradient w.r.t θ gives:

$$\begin{aligned} \nabla_\theta U(\theta) &= \nabla_\theta \sum_{\tau} P(\tau;\theta)R(\tau) \\ &= \sum_{\tau} \nabla_\theta P(\tau;\theta)R(\tau) \end{aligned} \tag{2}$$

Let's add the term $\frac{P(\tau;\theta)}{P(\tau;\theta)}$ to the right hand side of the equation

$$\begin{aligned} &= \sum_{\tau} \nabla_\theta \frac{P(\tau;\theta)}{P(\tau;\theta)} P(\tau;\theta)R(\tau) \\ &= \sum_{\tau} P(\tau;\theta) \frac{\nabla_\theta P(\tau;\theta)}{P(\tau;\theta)} R(\tau) \\ &= \sum_{\tau} P(\tau;\theta) \nabla_\theta \log P(\tau;\theta) R(\tau) \\ \nabla_\theta U(\theta) &= E[\nabla_\theta \log P(\tau;\theta) R(\tau)] \end{aligned} \tag{3}$$

This leaves us with an expectation for the term $\nabla_\theta \log P(\tau;\theta)R(\tau)$. We can compute an empirical approximation of that expresion by taking m sample trajectories (or paths) under the policy π_θ. Note that as of now we have not discussed how to calculate $P(\tau;\theta)$. Thus we use a Monte Carlo approach to approximate the gradient of the utility of π_θ:

$$\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=0}^{m} \nabla_\theta \log P(\tau^{(i)};\theta) R(\tau^{(i)}) \tag{4}$$

This works even if the reward function R is unkown and/or discontinuous. This works in discrete state spaces. The likelihood ratio changes the probability of experienced paths. That is, the probability of sampling trajectories.

Let's define the probability of a trajectory under a policy π_θ as:

$$\mathrm{P}\ (\tau;\theta) = \Pi_{t=0}^{H} \underbrace{P(s_{t+1}|s_t,u_t)}_{\text{dynamics models}} \underbrace{\pi_\theta(u_t|s_t)}_{\text{policy}} \quad (5)$$

From here we can calculate the term $\nabla_\theta \log P(\tau;\theta)$ present in equation 3

$$\begin{aligned}\nabla_\theta \log P(\tau;\theta) &= \nabla_\theta \log[\Pi_{t=0}^{H} P(s_{t+1}|s_t,u_t)\pi_\theta(u_t|s_t)] \\ &= \nabla_\theta[(\sum_{t=0}^{H} \log P(s_{t+1}|s_t,u_t)) + (\sum_{t=0}^{H} \log \pi_\theta(u_t|s_t))] \\ &= \sum_{t=0}^{H} \underbrace{\nabla_\theta \log \pi_\theta(u_t|s_t)}_{\text{no dynamics required!}}\end{aligned} \quad (6)$$

Which, if we plug into the original equation, we get:

$$\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{m}\sum_{i=0}^{m}\sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})(\sum_{k=t}^{H} R(s_t^{(i)},u_t^{(i)})) \quad (7)$$

This is an unbiased estimate and it works in theory. However it requries an impractical amount of samples, otherwise the approximation is very noisy. In order to overcome this limitation we can do the following tricks:

- Add a baseline
- Add temporal structure
- Use trust region and natural gradient.

1.1 Add a baseline

Substract a baseline from the equation **??** to reduce variance without introducing varience (cite Williams 1992):

$$\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{m}\sum_{i=0}^{m}\sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})(\sum_{k=t}^{H} R(s_t^{(i)},u_t^{(i)}) - b) \quad (8)$$

The problem with this equation is that each action u_i is being scaled by the whole sum of rewards $R(\tau)$. This means that, for instance, the last action

in a trajectory is taking into account rewards that happened much earlier in the episode. A way to compensate for this is to scale the value of an action u_i (clarify what is meant by this) only by rewards that depend on u_i. So the equation becomes:

$$\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=0}^{m} \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(u_t^{(i)} | s_t^{(i)}) (\sum_{k=t}^{H} R(s_t^{(i)}, u_t^{(i)})) \tag{9}$$

There are other proposed variance reduction techniques via tweaking the baseline which some researchers have studied (cite Greensmith)