11/16/2023: Policy Gradient Methods

Previously: Q-Learning: Given (s, a), predict $Q_{opt}(s, a)$
"regression"

Today: Policy Gradient: Given s, predict a
"classification"

Input: Representation of state s → Model θ (i.e. neural networks) → probability distribution over actions $\underbrace{\pi_\theta}_{\text{probabilistic policy defined by model parameters } \theta}(a \mid s)$

How to train?

- Normal supervised learning requires knowing $\underbrace{\text{best action}}_{\text{Not known}}$ for any state at given states as training data ✗
- Policy gradient: train $\pi_\theta(a \mid s)$ to achieve high total rewards

Want to maximize value of the policy π_θ

$$\underbrace{V(\theta)}_{\text{Expected sum of rewards when using policy } \pi_\theta(a|s)} = \sum_{\text{trajectories } z} \underbrace{P(z; \theta)}_{\text{Prob of } z \text{ happening}} \cdot \underbrace{R(z)}_{\text{Reward achieved}}$$

Trajectory = possible sequences of $[s_1, a_1, r_1, s_2, a_2, r_2, s_3, \ldots]$

$$= \sum_{t=1}^{T} r_t$$

Plan: $V(\theta)$ is our training objective, maximize with gradient ascent

What is $\nabla_\theta V(\theta)$?

$$\nabla_\theta V(\theta) = \sum_{\text{traj's } z} \nabla_\theta P(z;\theta) \cdot R(z)$$

Sum over exponentially many trajectories – infeasible

Key trick: $\nabla_\theta \log(P(z;\theta)) = \frac{1}{P(z;\theta)} \underline{\nabla_\theta P(z;\theta)}$

$$\Leftrightarrow \quad \nabla_\theta P(z;\theta) = P(z;\theta) \nabla_\theta \log P(z;\theta)$$

Plug this in to $\nabla_\theta V(\theta)$!

$$\nabla_\theta V(\theta) = \underbrace{\sum_{\text{traj's } z} P(z;\theta)}_{\text{Expected value of....}} \underbrace{\nabla_\theta \log P(z;\theta) \cdot R(z)}_{\text{this quantity}}$$

$$= \mathbb{E}_\theta \left[\nabla_\theta \log P(z;\theta) \cdot R(z) \right]$$

Estimate this by sampling trajectories with $\pi_\theta(a|s)$ and taking average of $\nabla_\theta \log P(z;\theta) \cdot R(z)$

What is $\nabla_\theta \log P(z;\theta)$?

$$\log P(z;\theta) = \boxed{\underbrace{\log P(s_1)}_{\text{start state}}} + \underbrace{\log \pi_\theta(a_1|s_1)}_{\text{policy}} + \boxed{\underbrace{\log T(s_1, a_1, s_2)}_{\text{transitions}}}$$

$[s_1, a_1, r_1, s_2, \ldots]$

$$+ \underbrace{\log \pi_\theta(a_2|s_2)}_{\text{policy}} + \cdots$$

Dont depend on θ

So ∇_θ is 0

$$\text{So } \nabla_\theta \log P(z;\theta) = \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Basic Policy Gradient Algorithm:

Initialize θ randomly

For each episode:

- Sample trajectory z using $\pi_\theta(a|s)$
- Update:

$$\theta \leftarrow \theta + \eta \underbrace{R(z) \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t | s_t)}_{\approx \nabla_\theta V(\theta)}$$