RL: Policy Search

Policy Gradient Algorithms

Marius Lindauer

Leibniz Universität Hannover

Winter Term 2021

Likelihood Ratio / Score Function Policy Gradient

$$\nabla_\theta V(\theta) = \frac{1}{m} \sum_{i=1}^{m} R(\tau^{(i)}) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)})$$

- Unbiased but very noisy
- Fixes that can make it practical
  - Temporal structure
  - Baseline
  - (and some more)

Policy Gradient: Use Temporal Structure

$$\nabla_\theta \mathbb{E}_\tau[R] = \mathbb{E}_\tau \left[\left(\sum_{t=0}^{T-1} r_t \right) \left(\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \right) \right]$$

We can repeat the same argument to derive the gradient estimator for a single reward term $r_{t'}$

$$\nabla_\theta \mathbb{E}[r_{t'}] = \mathbb{E} \left[r_{t'} \sum_{t=0}^{t'} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \right]$$

Summing this formula over t, we obtain:

$$\begin{aligned} V(\theta) = \nabla_\theta \mathbb{E}[R] \quad &= \quad \left[\sum_{t'=0}^{T-1} r_{t'} \sum_{t=0}^{t'} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \right] \\ &= \quad \left[\sum^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \sum^{T-1} r_{t'} \right] \end{aligned}$$

Policy Gradient: Use Temporal Structure

- Recall for a particular trajectory $\tau^{(i)}$, $\sum_{t'=t}^{T-1} r_{t'}^{(i)}$ is the return $G_t^{(i)}$

$$\nabla_\theta \mathbb{E}[R] \approx \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t, s_t) G_t^{(i)}$$

Monte-Carlo Policy Gradient (REINFORCE)

Leverages likelihood ratio / score function and temporal structure

$$\Delta\theta_t = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) G_t$$

REINFORCE:

1. Initialize policy parameters θ arbitrarily
2. for each episode $\{s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ do
   - for $t = 1$ to $T - 1$ do
     - $\theta := \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) G_t$
3. return θ

Policy Gradient: Introduce Baseline

- Reduce variance by introducing a baseline $b(s)$

$$\nabla_\theta \mathbb{E}_\tau[R] = \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t; \theta) \left(\sum_{t'=t}^{T-1} r_{t'} - b(s_t) \right) \right]$$

- For any choice of b, gradient estimator is unbiased
- Near optimal choice is the expected return

$$b(s_t) \approx \mathbb{E}[r_t + r_{t+1} + \ldots + r_{T-1}]$$

- Interpretation: increase logprob of action a_t proportionally to how much returns $\sum_{t'=t}^{T-1} r_{t'}$ are better than expected

"Vanilla" Policy Gradient Algorithm

- Initialize policy parameters θ and baseline b
- for iteration$= 1, 2, \dots$ do
  - Collect a set of trajectories by executing the current policy
  - At each time step t in each trajectory τ^i, compute
    - Return $G_t^i = \sum_{t'=t}^{T-1} r_{t'}^i$, and
    - Advantage estimate $\hat{A}_t^i = G_t^i - b(s_t)$
  - Re-fit the baseline by minimizing $\sum_i \sum_t ||b(s_t) - G_t^i||^2$
  - Update the policy, using a policy gradient estimate $\hat{g}$
    - which is a sum of terms $\nabla_\theta \log \pi(a_t \mid s_t; \theta) \hat{A}_t$
    - Apply gradient $\hat{g}$ by any DL-optimizer (e.g., SGD or ADAM)

Choosing the Baseline: Value Functions

- Recall Q-function:

$$Q^\pi(s,a) = \mathbb{E}_\pi[r_0 + \gamma r_1 + \gamma^2 r_2 + \dots \mid s_0 = s, a_0 = a]$$

- State-value function can serve as a great baseline

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi[r_0 + \gamma r_1 + \gamma^2 r_2 + \dots \mid s_0 = s] \\ &= \mathbb{E}_{a \sim \pi}[Q^\pi(s,a)] \end{aligned}$$

- Advantage function: Combining Q with baseline V:

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$$