Reinforcement Learning via Policy Optimization

Hanxiao Liu

November 22, 2017
Reinforcement Learning

Policy \( a \sim \pi(s) \)
Example - Mario
Example - ChatBot

Please select an option above.

Let me see main menu

See original options

Go back to all choices

This bot is stupid
Applications - Robotics
Applications - Combinatorial Problems
Supervised Learning vs RL

Supervised setup

▶ \( s_t \sim P(\cdot) \)
▶ \( a_t = \pi(s_t) \)
▶ immediate reward

Reinforcement setup

▶ \( s_t \sim P(\cdot|s_{t-1}, a_{t-1}) \)
▶ \( a_t \sim \pi(s_t) \)
▶ delayed reward

Both aims to maximize the total reward.
Markov Decision Process

What is the underlying assumption?
Landscape

Policy Optimization

DFO / Evolution
Policy Gradients
Actor-Critic Methods

Dynamic Programming
Policy Iteration
Value Iteration
Q-Learning

modified policy iteration
Monte Carlo

- Return $R(\tau) = \sum_{t=1}^{T} R(s_t, a_t)$.
- Trajectory $\tau = s_0, a_0, \ldots, s_T, a_T$

Goal: finding a good $\pi$ such that $R$ is maximized.

Policy evaluation via MC

1. Sample a trajectory $\tau$ given $\pi$.
2. Record $R(\tau)$

Repeat the above for many times and take the average.
Let’s parametrize $\pi$ using a neural network.

$\triangleright \ a \sim \pi_\theta(s)$

Expected return

$$\max_\theta U(\theta) = \max_\theta \sum_\tau P(\tau; \pi_\theta)R(\tau) \quad (1)$$

Heuristic: Raise the probability of good trajectories.

1. Sample a trajectory $\tau$ under $\pi_\theta$
2. Update $\theta$ using gradient $\nabla_\theta \log P(\tau; \pi_\theta)R(\tau)$. 
Policy Gradient

The log-derivative trick

\[ \nabla_{\theta} U(\theta) = \sum_{\tau} \nabla_{\theta} P(\tau; \pi_\theta) R(\tau) \]  

(2)

\[ = \sum_{\tau} \frac{P(\tau; \pi_\theta)}{P(\tau; \pi_\theta)} \nabla_{\theta} P(\tau; \pi_\theta) R(\tau) \]  

(3)

\[ = \sum_{\tau} P(\tau; \pi_\theta) \nabla_{\theta} \log P(\tau; \pi_\theta) R(\tau) \]  

(4)

\[ = \mathbb{E}_{\tau \sim P(\cdot; \pi_\theta)} \nabla_{\theta} \log P(\tau; \pi_\theta) R(\tau) \]  

(5)

\[ \approx \nabla_{\theta} \log P(\tau; \pi_\theta) R(\tau) \quad \tau \sim P(\cdot; \pi_\theta) \]  

(6)
Policy Gradient

\[ \nabla_{\theta} U(\theta) \approx \nabla_{\theta} \log P(\tau; \pi_{\theta}) R(\tau) \quad \tau \sim P(\cdot; \pi_{\theta}) \] 

(7)

- Analogous to SGD (so variance reduction is important), but data distribution here is a moving target (so we may want a trust region).
We can subtract any constant from the reward

\[ \nabla_\theta \sum_\tau P(\tau; \pi_\theta)(R(\tau) - b) \]  \hspace{1cm} (8)

\[ = \nabla_\theta \sum_\tau P(\tau; \pi_\theta)R(\tau) - \nabla_\theta \sum_\tau P(\tau; \pi_\theta)b \]  \hspace{1cm} (9)

\[ = \nabla_\theta \sum_\tau P(\tau; \pi_\theta)R(\tau) - \nabla_\theta b \]  \hspace{1cm} (10)

\[ = \nabla_\theta \sum_\tau P(\tau; \pi_\theta)R(\tau) \]  \hspace{1cm} (11)
Policy Gradient

The variance is magnified by $R(\tau)$

$$\nabla_\theta U(\theta) \approx \nabla_\theta \log P(\tau; \pi_\theta) R(\tau) \quad (12)$$

More fine-grained baseline?

$$\sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=0} R(s_{t'}, a_{t'}) \quad (13)$$

$$\rightarrow \sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=t} R(s_{t'}, a_{t'}) \quad (14)$$

$$\rightarrow \sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) (R_t - b_t) \quad (15)$$
Policy Gradient

\[
\sum_{t=0} \nabla_{\theta} \log \pi_{\theta}(a_t|s_t) (R_t - b_t)
\]  

Good choice of \( b_t \)?

\[
b^*_t = \arg\min_b \mathbb{E} (R_t - b)^2
\]

\[
\approx V_\phi(s_t)
\]

\[
\sum_{t=0} \nabla_{\theta} \log \pi_{\theta}(a_t|s_t) (R_t - V_\phi(s_t))
\]

- Promote actions that lead to positive advantage.
Policy Gradient

\[
\sum_{t=0} \nabla_{\theta} \log \pi_{\theta}(a_t|s_t) A_t
\]  

(20)

- Sample a trajectory \( \tau \)
- For each step in \( \tau \)
  - \( R_t = \sum_{t'=t}^{t'} r_{t'} \)
  - \( A_t = R_t - V_{\phi}(s_t) \)
- take a gradient step \( \nabla_{\phi} \sum_{t=0} \| V_{\phi}(s_t) - R_t \|^2 \)
- take a gradient step \( \nabla_{\theta} \sum_{t=0} \log \pi_{\theta}(a_t|s_t) A_t \)
In PG, our opt objective a moving target defined based on trajectory samples given the current policy

- each sample is only used once

An opt objective that reuses all historical data?
TRPO

Policy gradient

\[
\sum_{t=0} \nabla_{\theta} \log \pi_{\theta}(a_t|s_t) A_t
\]  \hspace{1cm} (21)

Objective (on-policy)

\[E_{\pi} [A(s, a)] \]  \hspace{1cm} (22)

Objective (off-policy), via importance sampling

\[E_{\pi_{old}} \left[ \frac{\pi(a|s)}{\pi_{old}(a|s)} A_{old}(s, a) \right] \]  \hspace{1cm} (23)
\[
\max_{\pi} \mathbb{E}_{\pi_{old}} \left[ \frac{\pi(a|s)}{\pi_{old}(a|s)} A_{old}(s, a) \right] \approx \sum_{n=1}^{N} \frac{\pi(a_n|s_n)}{\pi_{old}(a_n|s_n)} A_n \quad (24)
\]

subject to \( \text{KL}(\pi_{old}, \pi) \leq \delta \) \quad (25)

- Quadratic approximation is used for the KL.
- Use conjugate gradient for the natural gradient direction \( F^{-1} g \).
\[
\max_{\pi} \sum_{n=1}^{N} \frac{\pi(a_n|s_n)}{\pi_{old}(a_n|s_n)} A_n - \beta KL(\pi_{old}, \pi)
\]  

Fixed $\beta$ does not work well

- shrink $\beta$ when $KL(\pi_{old}, \pi)$ is small
- increase $\beta$ otherwise
Alternative ways to penalize large change in $\pi$? Modify

$$\frac{\pi(a_n|s_n)}{\pi_{old}(a_n|s_n)} A_n = r_n(\theta) A_n$$ \hspace{1cm} (27)

As

$$\min (r_n(\theta) A_n, \text{clip}(1 - \epsilon, 1 + \epsilon, r_n(\theta)) A_n)$$ \hspace{1cm} (28)

- being pessimistic whenever “a large change in $\pi$ leads to a better obj.”
- same as the original obj otherwise.
Advantage Estimation

Currently

\[ \hat{A}_t = r_t + r_{t+1} + r_{t+2} + \cdots - V(s_t) \]  
\hfill (29)

More bias, less variance (ignoring long-term effect)

\[ \hat{A}_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots - V(s_t) \]  
\hfill (30)

Bootstrapping

\[ \hat{A}_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots - V(s_t) \]  
\hfill (31)

\[ = r_t + \gamma(r_{t+1} + \gamma r_{t+2} + \cdots) - V(s_t) \]  
\hfill (32)

\[ = r_t + \gamma V(s_{t+1}) - V(s_t) \]  
\hfill (33)

We may unroll for more than one steps.
Advantage Actor-Critic

- Sample a trajectory $\tau$
- For each step in $\tau$
  - $\hat{R}_t = r_t + \gamma V(s_{t+1})$
  - $\hat{A}_t = \hat{R}_t - V(s_t)$
- take a gradient step using
  $$\nabla_{\theta, \phi} \sum_{t=0}^{\infty} \left[ -\log \pi_\theta(a_t|s_t) \hat{A}_t + \|V_\phi(s_t) - \hat{R}_t\|^2 \right]$$

May use TROP, PPO for policy optimization.
A3C

Variance reduction via multiple actors

Figure: Asynchronous Advantage Actor-Critic