

Alignment

Jordan Boyd-Graber

University of Maryland

Policy Methods

Adapted from slides by David Silver, Pieter Abbeel, and John Schulman

Reinforcement Learning is Everywhere!

- RL used to be niche subfield . . .
- Now it's all over the place
- Part of much of ML hype
- But what is reinforcement learning?

Reinforcement Learning is Everywhere!

- RL used to be niche subfield . . .
- Now it's all over the place
- Part of much of ML hype
- But what is reinforcement learning?
 - ▶ RL is a general-purpose framework for decision-making
 - ▶ RL is for an agent with the capacity to act
 - ▶ Each action influences the agent's future state
 - ▶ Success is measured by a scalar reward signal
 - ▶ Goal: select actions to maximise future reward

Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning

Ronald J. Williams
College of Computer Science
Northeastern University
Boston, MA 02115

Appears in *Machine Learning*, 8, pp. 229-256, 1992.

Foundation of Policy Gradient

Likelihood Ratio Policy Gradient

Let τ be state-action $s_0, u_0, \dots, s_H, u_H$. Utility of policy π parametrized by θ is

$$U(\theta) = \mathbb{E}_{\pi_{\theta}, U} \left[\sum_t^H R(s_t, u_t); \pi_{\theta} \right] = \sum_{\tau} P(\tau; \theta) R(\tau). \quad (1)$$

Our goal is to find θ :

$$\max_{\theta} U(\theta) = \max_{\theta} \sum_{\tau} p(\tau; \theta) R(\tau) \quad (2)$$

Likelihood Ratio Policy Gradient

Let τ be state-action $s_0, u_0, \dots, s_H, u_H$. Utility of policy π parametrized by θ is

$$U(\theta) = \mathbb{E}_{\pi_{\theta}, U} \left[\sum_t^H R(s_t, u_t); \pi_{\theta} \right] = \sum_{\tau} P(\tau; \theta) R(\tau). \quad (1)$$

Our goal is to find θ :

$$\max_{\theta} U(\theta) = \max_{\theta} \sum_{\tau} p(\tau; \theta) R(\tau) \quad (2)$$

Connecting to Generation

- τ are internal states of decoder and the tokens they produce.
- Reward is how good the output is (more on that later).
- θ are the encoder/decoder parameters

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

(4)

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

$$\nabla_{\theta} U(\theta) = \sum_{\tau} R(\tau) \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_{\theta} P(\tau; \theta) \quad (4)$$

(5)

Move differentiation inside sum (ignore $R(\tau)$ and then add in term that cancels out

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

$$\nabla_{\theta} U(\theta) = \sum_{\tau} R(\tau) \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_{\theta} P(\tau; \theta) \quad (4)$$

$$= \sum_{\tau} P(\tau; \theta) \frac{\nabla_{\theta} P(\tau; \theta)}{P(\tau; \theta)} R(\tau) \quad (5)$$

$$(6)$$

Move derivative over probability

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

$$\nabla_{\theta} U(\theta) = \sum_{\tau} R(\tau) \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_{\theta} P(\tau; \theta) \quad (4)$$

$$= \sum_{\tau} P(\tau; \theta) \frac{\nabla_{\theta} P(\tau; \theta)}{P(\tau; \theta)} R(\tau) \quad (5)$$

$$= \sum_{\tau} P(\tau; \theta) \nabla_{\theta} [\log P(\tau; \theta)] R(\tau) \quad (6)$$

Assume softmax form ($\nabla_{\theta} \log z = \frac{1}{z} \nabla_{\theta} z$)

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

$$= \sum_{\tau} P(\tau; \theta) \nabla_{\theta} [\log P(\tau; \theta)] R(\tau) \quad (4)$$

Approximate with empirical estimate for m sample paths from π

$$\nabla_{\theta} U(\theta) \approx \frac{1}{m} \sum_i^m \nabla_{\theta} \log P(x^i; \theta) R(\tau^i) \quad (5)$$

For generation

- (Gradient of) probability of outputting sequence
- How good that sequence is
- Averaged over outputs

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

$$= \sum_{\tau} P(\tau; \theta) \nabla_{\theta} [\log P(\tau; \theta)] R(\tau) \quad (4)$$

Approximate with empirical estimate for m sample paths from π

$$\nabla_{\theta} U(\theta) \approx \frac{1}{m} \sum_i^m \nabla_{\theta} \log P(x^i; \theta) R(\tau^i) \quad (5)$$

For generation

- (Gradient of) probability of outputting sequence
- How good that sequence is
- Averaged over outputs

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

$$= \sum_{\tau} P(\tau; \theta) \nabla_{\theta} [\log P(\tau; \theta)] R(\tau) \quad (4)$$

Approximate with empirical estimate for m sample paths from π

$$\nabla_{\theta} U(\theta) \approx \frac{1}{m} \sum_i^m \nabla_{\theta} \log P(x^i; \theta) R(\tau^i) \quad (5)$$

For generation

- (Gradient of) probability of outputting sequence
- How good that sequence is
- Averaged over outputs

Likelihood Ratio Policy Gradient

$$\sum_{\tau} p(\tau; \theta) R(\tau) \quad (3)$$

Taking the gradient wrt θ :

$$= \sum_{\tau} P(\tau; \theta) \nabla_{\theta} [\log P(\tau; \theta)] R(\tau) \quad (4)$$

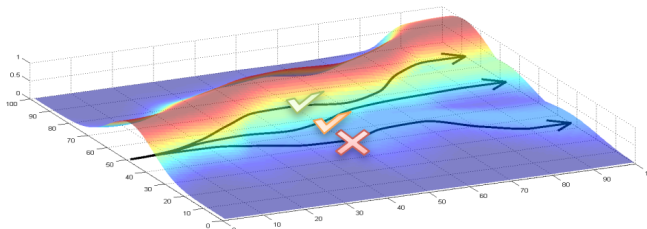
Approximate with empirical estimate for m sample paths from π

$$\nabla_{\theta} U(\theta) \approx \frac{1}{m} \sum_i^m \nabla_{\theta} \log P(x^i; \theta) R(\tau^i) \quad (5)$$

For generation

- (Gradient of) probability of outputting sequence
- How good that sequence is
- Averaged over outputs

Policy Gradient Intuition



- Increase probability of paths with positive R
- Decrease probability of paths with negative R

Extensions

- Consider baseline b (e.g., path averaging)

$$\nabla_{\theta} U(\theta) \approx \frac{1}{m} \sum_1^m \nabla_{\theta} \log P(r^i; \theta) (R(\tau^i) - b(\tau)) \quad (6)$$

- Combine with value estimation (critic)
 - ▶ Actor: What actions to take
 - ▶ Critic: How good those actions are
 - ▶ Advantage Actor Critic with temporal difference (remember TD Gammon?) term

$$A_{\pi_{\theta}} = r(s_t, a_t) + V_{\pi_{\theta}}(s_{t+1}) - V_{\pi_{\theta}}(s_t) \quad (7)$$

- Proximal policy optimization: policies should not change too much

Recap

- Reinforcement learning is active subfield of ML
- Deep learning option for learning policy / value functions
- Representation learning helps cope with large state spaces
- Still requires careful engineering and feature engineering