Machine Learning

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


Foundation of Policy Gradient
Likelihood Ratio Policy Gradient

Let \( \tau \) be state-action \( s_0, u_0, \ldots, s_H, u_H \). Utility of policy \( \pi \) parametrized by \( \theta \) is

\[
U(\theta) = \mathbb{E}_{\pi_\theta, U} \left[ \sum_{t=0}^{H} R(s_t, u_t); \pi_\theta \right] = \sum_\tau P(\tau; \theta)R(\tau). \tag{1}
\]

Our goal is to find \( \theta \):

\[
\max_\theta U(\theta) = \max_\theta \sum_\tau P(\tau; \theta)R(\tau) \tag{2}
\]
Likelihood Ratio Policy Gradient

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

Taking the gradient wrt \( \theta \):

\[ (4) \]
Likelihood Ratio Policy Gradient

\[ \sum_{\tau} p(\tau; \theta) R(\tau) \]  

(3)

Taking the gradient wrt \( \theta \):

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

(4)

(5)

Move differentiation inside sum (ignore \( R(\tau) \) and then add in term that cancels out
Likelihood Ratio Policy Gradient

\[ \sum_{\tau} p(\tau; \theta)R(\tau) \]  \hspace{1cm} (3)

Taking the gradient wrt \( \theta \):

\[ \nabla_{\theta} U(\theta) = \sum_{\tau} R(\tau) \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_{\theta} P(\tau; \theta) \]  \hspace{1cm} (4)

\[ = \sum_{\tau} p(\tau; \theta) \frac{\nabla_{\theta} P(\tau; \theta)}{P(\tau; \theta)} R(\tau) \]  \hspace{1cm} (5)

Move derivative over probability
Likelihood Ratio Policy Gradient

\[
\sum_{\tau} p(\tau; \theta) R(\tau)
\]  \hspace{1cm} (3)

Taking the gradient wrt \( \theta \):

\[
\nabla_\theta U(\theta) = \sum_{\tau} R(\tau) \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_\theta P(\tau; \theta)
\]  \hspace{1cm} (4)

\[
= \sum_{\tau} P(\tau; \theta) \frac{\nabla_\theta P(\tau; \theta)}{P(\tau; \theta)} R(\tau)
\]  \hspace{1cm} (5)

\[
= \sum_{\tau} P(\tau; \theta) \nabla_\theta \left[ \log P(\tau; \theta) \right] R(\tau)
\]  \hspace{1cm} (6)

Assume softmax form \((\nabla_\theta \log z = \frac{1}{z} \nabla_\theta z)\)
Likelihood Ratio Policy Gradient

\[ \sum_{\tau} p(\tau; \theta)R(\tau) \]  
(3)

Taking the gradient wrt \( \theta \):

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

Approximate with empirical estimate for \( m \) sample paths from \( \pi \)

\[ \nabla_{\theta} U(\theta) \approx \frac{1}{m} \sum_{i} \nabla_{\theta} \log P(r^i; \theta)R(\tau^i) \]  
(5)
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)

\[ \nabla_\theta U(\theta) \approx \frac{1}{m} \sum_{1}^{m} \nabla_\theta \log P(r^i; \theta)(R(\tau^i) - b(\tau)) \]  

- Combine with value estimation (critic)
  - Critic: Updates action-value function parameters
  - Actor: Updates policy parameters in direction suggested by critic

- Proximal policy optimization: policies should not change too much
Recap

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