Policy Gradients

Ashwin

University of Colorado Boulder

ashwin@asokan.edu

April 15, 2017

Why Policy gradients

- Recap: Policy Iterations. In interesting real world cases, we cannot practically evaluate all the states, we got to generalize.
- At any state, in every time stamp an agent gets to pick action. Policies give actions directly. Why not directly learn the policy?
- Better convergence.
- In high dimensional spaces, value space becomes difficult to fathom.
- We could learn a stochastic policy instead of being deterministic, randomness helps out.

Learning Policy

- (π, θ)
- $\rho(\pi)$
- $\pi_\theta(s, a) = \Pr[a|s, \theta]$
- $\Delta\theta = \alpha \frac{\partial \rho}{\partial \theta}$
- Maximize $E[R|\pi]$

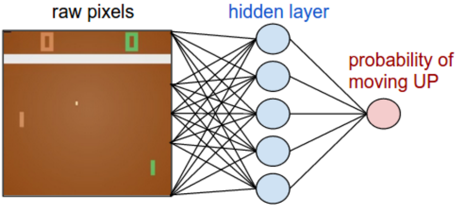

Figure: A Policy Network.

Score Function Gradient Estimator

Consider an expectation for a function $f(x)$ which we would like to maximize, x being parameterized by weight θ

$$E_{x\sim p(x|\theta)}[f(x)] \tag{1}$$

$$\nabla_\theta E_{x\sim p(x|\theta)}[f(x)] = \nabla_\theta \int p(x|\theta) f(x) dx \tag{2}$$

$$= \int \nabla_\theta p(x|\theta) \frac{p(x|\theta)}{p(x|\theta)} f(x) dx \tag{3}$$

$$= \int \nabla_\theta \log p(x|\theta) p(x|\theta) f(x) dx \tag{4}$$

$$= E_x[f(x) \nabla_\theta \log p(x|\theta)] \tag{5}$$

Score Function Gradient Estimator - Intuition

$$\hat{g} = f(x)\nabla_\theta \log p(x|\theta)$$

- We could sample $x \sim p(x|\theta)$ and compute the $\hat{g}$.
- We adjust θ based on both $f(x)$ and $\hat{g}$.
- $f(x)$ measures how good our estimate is doing. So we use it decide how much we should adjust.
- $\hat{g}$ tells us which direction we should move the weights.

Derive Policy Gradients

$x = (s_0, a_0, r_0, s_1, a_1, r_1, ...s_{T-1}, a_{T-1}, r_{T-1}, s_T)$

$$\nabla_\theta E_x[R(x)] = E_x[R(x)\nabla_\theta \log p(x|\theta)] \tag{6}$$

$$p(x|\theta) = \mu(s_0) \prod_{t=0}^{T-1} [\pi(a_t|s_t, \theta)P(s_{t+1}, r_t|s_t, a_t)] \tag{7}$$

$$\log[p(x|\theta)] = \log[\mu(s_0)] + \sum_{t=0}^{T-1} [\log \pi(a_t|s_t, \theta) + \log P(s_{t+1}, r_t|s_t, a_t)] \tag{8}$$

$$\nabla_\theta \log[p(x|\theta)] = \nabla_\theta \sum_{t=0}^{T-1} [\log \pi(a_t|s_t, \theta) \tag{9}$$

$$\nabla_\theta E_x[R(x)] = E_x[R(x)\nabla_\theta \sum_{t=0}^{T-1} [\log \pi(a_t|s_t, \theta)] \tag{10}$$

Derive Policy Gradients

For reward over a single step,

$$\nabla_\theta E[r_{t'}] = E\langle r_{t'} \sum_{t=0}^{t'} [\nabla_\theta \log \pi(a_t|s_t,\theta)]\rangle \tag{11}$$

Sum this over timesteps,

$$\nabla_\theta E[R] = E\langle \sum_{t=0}^{T-1} [r_t] \sum_{t=0}^{t'} [\nabla_\theta \log \pi(a_t|s_t,\theta)]\rangle \tag{12}$$

$$= E\langle \sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t|s_t,\theta) \sum_{t'=t}^{T-1} [r_{t'}]\rangle \tag{13}$$

Policy Gradient Theorem

Theorem (Policy Gradient Theorem)

For any differentiable policy $\pi_\theta(s, a)$, and any objective function $J(\theta)$ the policy gradient is,
$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s, a) Q^{\pi_\theta}(s, a)]$

Reduce Variance with Baseline

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

$$\nabla_\theta E_x[R] = E_x \langle \sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t|s_t, \theta)[\sum_{t'=t}^{T-1} r_{t'} - b(s_t)] \rangle \qquad (14)$$

- For any baseline b we pick the estimator remains unbiased
- Optimal baseline policy is total expected reward $b(s_t) \approx E[r_t + r_{t+1} + .. + r_{T-1}]$
- Move in the gradient direction but by how much better it is with respect to some baseline that we already know.

Unbiased with Baseline

$$E_x[\nabla_\theta \log \pi(a_t|s_t,\theta) b(s_t)] \tag{15}$$

$$= E_{s_{0:t}, a0:t-1}[E_{s_{t+1:T}, at:T-1}[\nabla_\theta \log \pi(a_t|s_t,\theta) b(s_t)]] \tag{16}$$

$$= E_{s_{0:t}, a0:t-1}[b(s_t) E_{s_{t+1:T}, at:T-1}[\nabla_\theta \log \pi(a_t|s_t,\theta)]] \tag{17}$$

$$= E_{s_{0:t}, a0:t-1}[b(s_t) E_{a_t}[\nabla_\theta \log \pi(a_t|s_t,\theta)]] \tag{18}$$

$$= E_{s_{0:t}, a0:t-1}[b(s_t).0] \tag{19}$$

Reduce Variance with Discounts

$$\nabla_\theta E_x[R] \approx E_x\langle \sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t|s_t,\theta)[\sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'} - b(s_t)]\rangle \qquad (20)$$

Now baseline becomes $b(s_t) \approx E[r_t + \gamma r_{t+1} + .. + \gamma^{T-1-t} r_{T-1}]$

Reduce Variance with a Critic

- We could use a critic to estimate action value function $Q_w(s,a) \approx Q^{\pi_\theta}(s,a)$
- Now we got two parameters to learn simultaneously.
  - Critic: Updates Action value function parameters w
  - Actor: Updates Policy parameters θ in direction pointed by the critic.
- Approximate Policy Gradient $\nabla_\theta J[\theta] \approx E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)]$

Deal with Critic's Bias

Theorem (Compatible Function Approximation Theorem)

If the following two conditions are satisfied:

1. *Value function approximator is compatible to the policy*
   $\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$
2. *Parameter w minimize the mean-squared error*
   $\epsilon = E_{\pi_\theta}[(Q^{\pi_\theta}(s,a) - Q_w(s,a))^2]$

Then the policy gradient is exact
$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)]$

Proof of Compatible Function Approximation Theorem

If w is chosen to minimize mean-squared error, then its gradient with respect to w should be zero.

$$\nabla_w \epsilon = 0 \tag{21}$$

$$E_{\pi_\theta}[(Q^\theta(s,a) - Q_w(s,a))\nabla_w Q_w(s,a)] = 0 \tag{22}$$

$$E_{\pi_\theta}[(Q^\theta(s,a) - Q_w(s,a))\nabla_\theta \log \pi_\theta(s,a)] = 0 \tag{23}$$

$$E_{\pi_\theta}[(Q^\theta(s,a))\nabla_\theta \log \pi_\theta(s,a)] = E_{\pi_\theta}[Q_w(s,a))\nabla_\theta \log \pi_\theta(s,a)] \tag{24}$$

Now $Q_w(s,a)$ can be used in policy gradient

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)] \tag{25}$$

Policy Gradient Algorithm

Algorithm 1: Policy Gradient with Baseline

Initialize policy weight θ, baseline b;
for *iteration=1,2...* **do**
    Collect a set of trajectories by executing the current policy
    At each timestep in each trajectory, compute
        $R_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'} \qquad \hat{A}_t = R_t - b(s_t)$
    Refit the baseline, by minimizing $|b(s_t) - R_t|^2$ summing up over all trajectories and timesteps
    Update the policy, using the policy gradient estimate $\hat{g}$
end

Policy Gradient Algorithms - Summary

The policy gradient has many variations

- $\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s, a) V_t]$ REINFORCE
- $\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s, a) Q_w(s, a)]$ Q - Actor Critic
- $\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s, a) A_w(s, a)]$ Advantage - Actor Critic

References

- Berkeley Course on Deep RL: Lecture 9 - Reinforcement learning with policy gradients
- UCL Course on RL: Lecture 7 - Policy Gradient Methods
- Andrej Karpathy blog: Deep Reinforcement Learning - Pong from Pixels
- Policy Gradient Methods for Reinforcement Learning with Function Approximation
- Denny Britz's Policy Gradient Methods