General Review of Terms

Policy: Dictates what action to take given a state.

Deterministic, $\pi : S \to A$, maps states to actions

Stochastic, $\pi(a|s)$, conditional probability distribution

MDP: 5 tuple $\{$ S (set of states),

A (set of actions),

$P(s_{t+1}=s' | s_t=s, a_t=a)$ also $P(s'|s,a)$ (probability of state transition with action a),

$R(s'|s,a)$ (reward obtained when going to s' with a from s),

γ (discount factor, $\gamma=0$ is immediate reward, $\gamma=1$ is total reward) $\}$

Value Function: Value of state - expected reward at starting at state and continuing with policy

Written $V^\pi(s)$, $Q^\pi(s,a)$, conditional to policy π. If conditional to optimal policy, π^*, then $V^*(s)$, $Q^*(s,a)$.

$V^\pi(s)$ expected value of following π forever when agent starts at s.

$Q^\pi(s,a)$ expected value of following π forever when first taking action a in given state s.

$V^\pi(s) = E_\pi\{R_t | s_t=s\} = E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,|\, s_t=s\right\}$, expected return starting from that state given π

s_0, if $\gamma \in [0,1]$ is 0, $\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} = r_{t+1}$, immediate reward. If $\gamma=1$, then r at all time steps equally weighted

Note, $R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} \ldots$

$r_{t+1} + \gamma(r_{t+2} + \gamma r_{t+3} \cdots)$ $\quad E_\pi\{\cdot\}$ is expectation operator for policy π.

$r_{t+1} + \gamma R_{t+1} \Rightarrow$

$V^\pi(s) = E_\pi\{R_t | s_t=s\}$

$E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t=s\}$

Policy Gradient

Objective function $J(\theta) = E_\theta\left[\sum_{t=0}^{H} R(s_t, a_t); \pi_\theta\right] = \sum_\tau P(\tau;\theta) R(\tau)$ ← stochastic policy

(over all trajectories; probability of trajectory τ; reward of trajectory)

Find policy θ that creates trajectories τ $(s_1, a_1, s_2, a_2, s_3, a_3, \ldots s_H, a_H)$

that maximizes expected rewards, $\max_\theta J(\theta) = \max_\theta \sum_\tau P(\tau;\theta) R(\tau)$

Expectation in continuous state space $E_{x\sim p(x)}[S(x)] = \int p(x) S(x)\, dx$

Derivations

Note: $\nabla_\theta S(x) = S(x) \frac{\nabla_\theta S(x)}{S(x)} = S(x) \nabla_\theta \ln S(x)$

Let $J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[r(\tau)] = \int \pi_\theta(\tau) r(\tau)\, d\tau$, $\pi_\theta(\tau) = P(\tau)$ given π_θ ← Expectation of τ is Prob(τ) · reward(τ), or $\pi(\tau) \cdot r(\tau)$

Now $\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau)\, d\tau$

$= \int \pi_\theta(\tau) \underbrace{\nabla_\theta \ln \pi_\theta(\tau)}_{\text{let this be } r(\tau)} r(\tau)\, d\tau$

$= E_{\tau\sim\pi_\theta(\tau)}\left[r(\tau) \nabla_\theta \ln \pi_\theta(\tau)\right]$

Thus, policy gradient is an expectation.

$\nabla_\theta \ln \pi(\tau) \rightarrow$ First, find $\pi(\tau) = \pi(s_1, a_1, s_2, a_2, s_T, a_T) = p(s_1) \prod_{t=1}^{T} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t, a_t)$

($p(s_1)$: push starting state; $\pi_\theta(a_t|s_t)$: push we take action given state; $p(s_{t+1}|s_t,a_t)$: push we transition to next state ← environment event)

$p(s_1)\prod_{t=1}^{T} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t,a_t)$ = P(total trajectory)

Then, $\ln \pi(\tau) = \ln\left(p(s_1) \prod_{t=1}^{T} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t,a_t)\right)$

$\ln p(s_1) + \sum_t \ln \pi_\theta(a_t|s_t) + \sum_t \ln p(s_{t+1}|s_t, a_t)$

Now, apply ∇_θ

$\nabla_\theta \ln p(s_1) + \sum_t \nabla_\theta \ln \pi_\theta(a_t|s_t) + \sum_t \nabla_\theta \ln p(s_{t+1}|s_t, a_t)$

$\nabla_\theta \ln \pi(\tau) = \sum_t \nabla_\theta \ln \pi_\theta(a_t|s_t)$

$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta(\tau)}\left[r(\tau) \sum_t \nabla_\theta \ln \pi_\theta(a_t|s_t)\right]$ ← plug-in from above

$\nabla_\theta J(\theta) \approx \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \ln \pi_\theta(a_{i,t}|s_{i,t})\right)\left(\sum_{t=1}^{T} r(s_{i,t}, a_{i,t})\right)$ ← Monte-Carlo, sample τ

over all τ (N); within a τ, this is ∇_θ of expectation; likelihood of τ; reward of τ (T)

Remember:

$E_\theta\left[\sum_{t=0}^{H} R(s_t, a_t); \pi_\theta\right] = \sum_\tau P(\tau;\theta) R(\tau)$

for arbitrary states/actions given policy; for $t \in \{0, \ldots, H\}$; over τ

Can maximize rewards of θ over expectation pertaining to τ

$\theta = \theta + \alpha \nabla_\theta J$

Note: $\sum_{t=1}^{T} \nabla_\theta \ln \pi_\theta(a_{i,t}|s_{i,t})$ is max by likelihood from $\nabla_\theta s(x) = s(x)\nabla_\theta \ln s(x)$

Prevents term of $\prod_{t=1}^{T} \pi_\theta(a_t|s_t)$ from vanishing great (summation used instead of π).

Now for Cart-Pole-v1...

$$\nabla_\theta J(\theta) \approx \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \ln \pi_\theta(a_{i,t}|s_{i,t})\right)\left(\sum_{t=1}^{T} r(s_{i,t}, a_{i,t})\right)$$

Could use logistic since $|A|=2$. But, softmax generalizes.

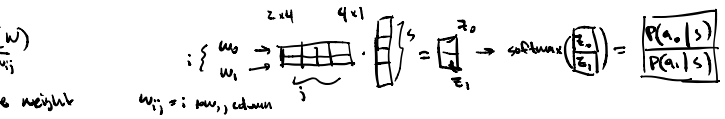
So, need: parametrized policy which tells us probability of action given state

$A = \{0,1\}$, but $\pi(a_0|s)$, $\pi(a_1|s)$ = softmax with $\pi_w(a_i|s) = \frac{e^{w_i \cdot s}}{\sum_{i=0}^{|A|} e^{w_i \cdot s}}$

Stochastic policy $\pi_w(a|s)$

Trying to derive: $\frac{\partial J(w)}{\partial w_{ij}}$

Update rule for single weight of system.



$\frac{\partial}{\partial w_{ij}} \ln \pi(a_1|s)$ ← π dependent on $k=i$, $k \neq i$ due to bottom of softmax term. Could use Kronecker delta, implement with one-hot encoding,

$\frac{\partial}{\partial w_{ij}} \left(\ln e^{w_k \cdot s} - \ln \sum_{p=0}^{|A|} e^{w_p \cdot s}\right)$ For $i=k$ Note: i = action, j = state dimension For $i \neq k$ $\frac{\partial}{\partial w_{ij}} \left(\ln e^{w_k \cdot s} - \ln \sum_{p=0}^{|A|} e^{w_p \cdot s}\right)$

$s_j - \left(\sum_{p=0}^{|A|} e^{w_p \cdot s}\right)^{-1} \frac{\partial}{\partial w_{ij}} \left(\sum_{p=0}^{|A|} e^{w_p \cdot s}\right)$

$s_j - \left(\sum_{p=0}^{|A|} e^{w_p \cdot s}\right)^{-1} s_j e^{w_i \cdot s}$

$s_j - s_j \pi(a_k|s)$

$s_j(1 - \pi(a_i|s))$

$-\left(\sum_{p=0}^{|A|} e^{w_p \cdot s}\right)^{-1} s_j e^{w_i \cdot s}$

$-s_j \pi(a_i|s)$

Thus, $\frac{\partial}{\partial w_{ij}} \ln \pi(a_k|s) = \begin{cases} s_j(1-\pi(a_i|s)) & \text{if } i=k \\ -s_j \pi(a_i|s) & \text{if } i \neq k \end{cases}$

$\forall w_{ij} \in \pi_w$, $\frac{\partial}{\partial w_{ij}} \ln \pi(a_k|s) = s_j(\delta - \pi(a_i|s))$, where $\begin{cases} \delta = 1 & \text{if } i=k \\ \delta = 0 & \text{if } i \neq k \end{cases}$

Update Rule: $w_{ij} = w_{ij} + \alpha \frac{\partial J}{\partial w_{ij}}$

for single τ $= w_{ij} + \alpha \sum_t^T \left[s_{jt}(\delta - \pi(a_i|s_t)) r(s_t, a_t)\right]$

Vectorized: $w = w + \alpha \sum_t^T \left[\delta - \pi(a|s_t)\right] r(s_t, a_t)] s^T$, where δ is one-hot of action taken on step t at τ

Given $w \in \mathbb{R}^{|A| \times |S|}$, $s \in \mathbb{R}^{|S| \times 1}$, $\pi(a|s_t) = \mathbb{R}^{|A| \times 1}$, where s_t is feature of s.

(w: softmax parameter matrix; s: state input; $\pi(a|s_t)$: output of softmax)

Problem: $w = w - \alpha \sum_t^T \left[\delta - \pi(a|s_t) r(s_t, a_t)\right] s^T$ seems to converge (subtracting gradient)!?

$(\delta - \pi(a|s_t))$ should say how far our policy was from action

$\pm$ existing weights to get to that action

$r(s_t, a_t)$ tells us how much we want to get to that action

if high, amplifies weight change, if low, reduces weight change

Adding this to weights increases prob. of high reward choices... so why does subtracting work?!