Policy Gradient with Baselines

Alina Vereshchaka

CSE4/510 Reinforcement Learning
Fall 2019

avereshc@buffalo.edu

October 29, 2019

*Slides are adopted from Deep Reinforcement Learning and Control by Katerina Fragkiadaki (Carnegie Mellon) & Policy Gradients by David Silver

Policy Objective Functions

Goal: given policy $\pi_\theta(s, a)$ with parameters θ, find best θ

- In episodic environments we can use the start value

$$J_1(\theta) = V_{\pi_\theta}(s_1) = \mathbb{E}_{\pi_\theta}[v_1]$$

- In continuing environments we can use the average value

$$J_{avV}(\theta) = \sum_s d_{\pi_\theta}(s) V_{\pi_\theta}(s)$$

- Or the average reward per time-step

$$J_{avR}(\theta) = \sum_s d_{\pi_\theta}(s) \sum_a \pi_\theta(s, a) R_s^a$$

where $d^{\pi_\theta}(s)$ is a stationary distribution of Markov chain for π_θ

Example: Pong from Pixels

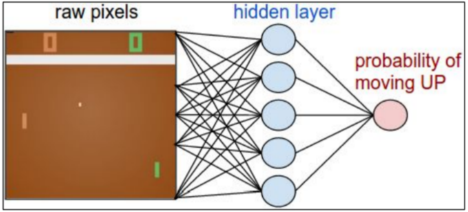
Suppose we had the training labels…
(we know what to do in any state)

(x1,UP)
(x2,DOWN)
(x3,UP)
...

Example: Pong from Pixels

Suppose we had the training labels…
(we know what to do in any state)

(x1,UP)
(x2,DOWN)
(x3,UP)
...



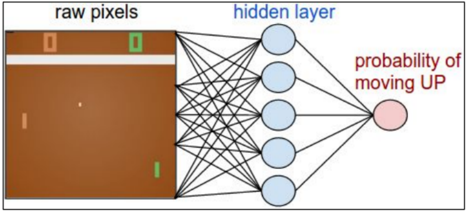
Example: Pong from Pixels

Suppose we had the training labels…
(we know what to do in any state)

(x1,UP)
(x2,DOWN)
(x3,UP)
...

maximize:

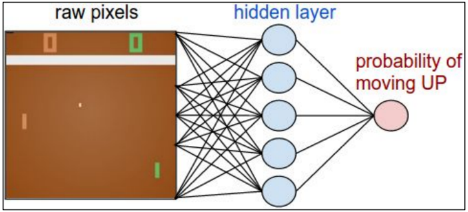
$$\boxed{\sum_i \log p(y_i \mid x_i)}$$

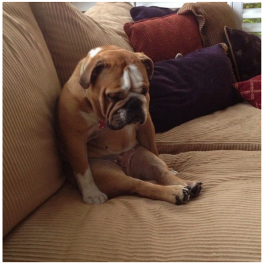


supervised learning

Example: Pong from Pixels

Except, we don't have labels...





Should we go UP or DOWN?

Example: Pong from Pixels

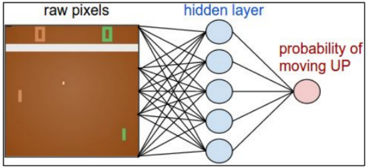
Except, we don't have labels...

"Try a bunch of stuff and see what happens. Do more of the stuff that worked in the future."

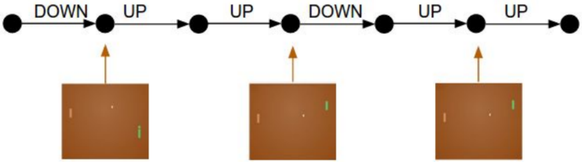
-RL

trial-and-error learning

Let's just act according to our current policy...



Rollout the policy and collect an episode

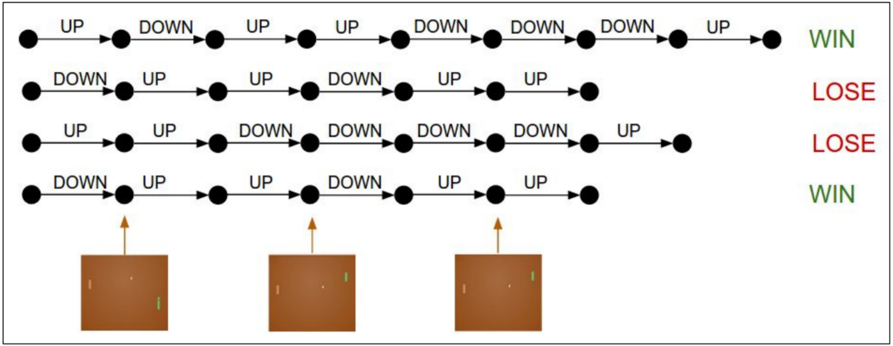


WIN

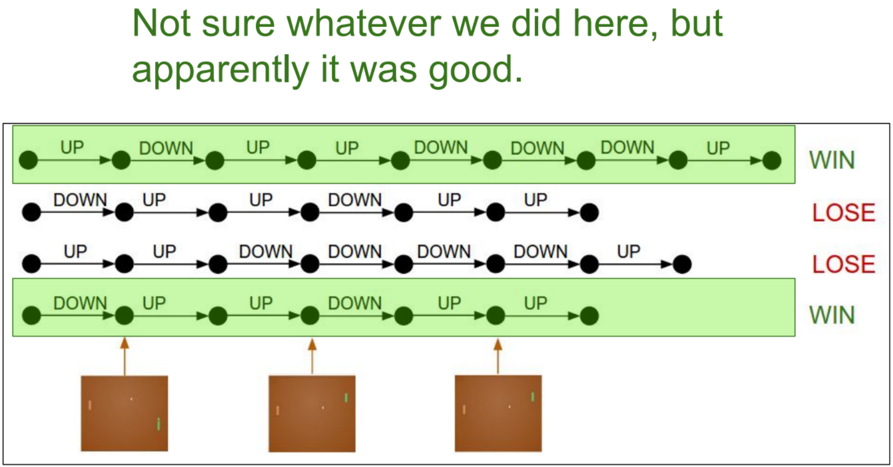
Example: Pong from Pixels

Collect many rollouts...

4 rollouts:

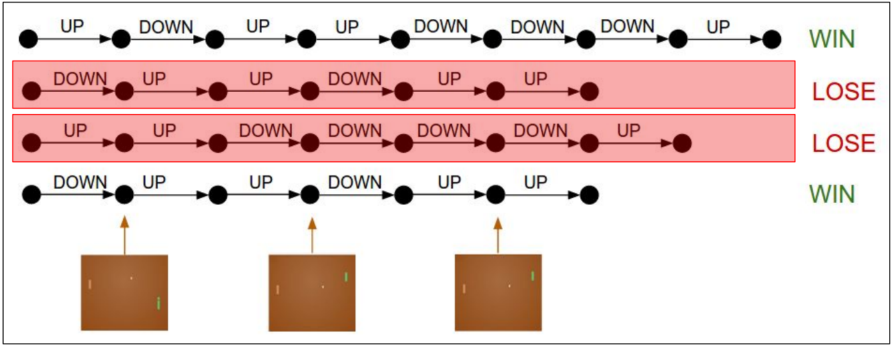


Example: Pong from Pixels



Example: Pong from Pixels

Not sure whatever we did here, but it was bad.



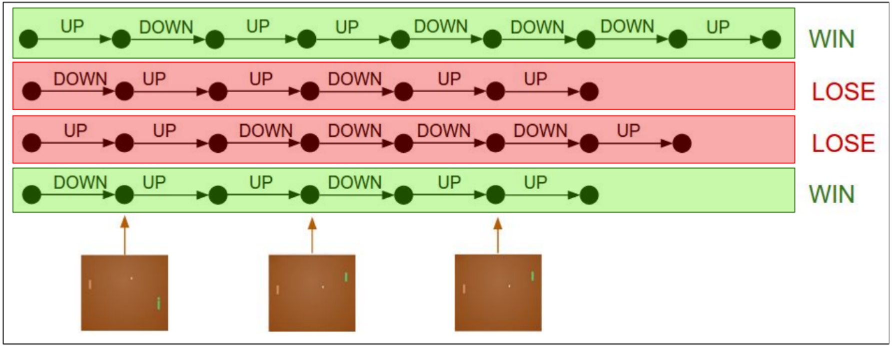
Example: Pong from Pixels

Pretend every action we took here was the correct label.

maximize: $\log p(y_i \mid x_i)$

Pretend every action we took here was the wrong label.

maximize: $(-1) * \log p(y_i \mid x_i)$

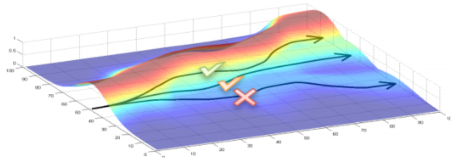


$$\nabla_\theta U(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)}) R(\tau^{(i)})$$

REINFORCE with Baseline

$$\nabla U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta) R(\tau^{(i)})$$

- Gradient tries to:
  - Increase probability of paths with positive R
  - Decrease probability of paths with negative R



! Likelihood ratio changes probabilities of experienced paths, does not try to change the paths (<-> Path Derivative)

REINFORCE (Monte-Carlo Policy Gradient)

- Update parameters by stochastic gradient ascent
- Using policy gradient theorem
- Using return G_t as an unbiased sample of $Q^{\pi_\theta}(s_t, a_t)$

$$\Delta\theta_t = \alpha G_t \nabla_\theta \log \pi_\theta(s_t, a_t)$$

REINFORCE, A Monte-Carlo Policy-Gradient Method (episodic)

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta}), \forall a \in \mathcal{A}, s \in \mathcal{S}, \boldsymbol{\theta} \in \mathbb{R}^n$
Initialize policy weights $\boldsymbol{\theta}$
Repeat forever:
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  For each step of the episode $t = 0, \ldots, T-1$:
    $G_t \leftarrow$ return from step t
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\gamma^t G_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t|S_t, \boldsymbol{\theta})$

Policy Gradient

This vanilla policy gradient update has no bias but high variance. Many following algorithms were proposed to reduce the variance while keeping the bias unchanged.

$$\nabla_\theta J(\theta) = \mathbb{E}_\pi[Q^\pi(s, a)\nabla_\theta \ln \pi_\theta(a|s)]$$

Policy Gradient Theorem

- The policy gradient theorem generalizes the likelihood ratio approach to multi-step MDPs
- Replaces instantaneous reward r with long-term value $Q^{\pi}(s,a)$
- Policy gradient theorem applies to start state objective, average reward and average value objective

---

- For any differentiable policy $\pi_\theta(s, a)$, the policy gradient is

$$\boxed{\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) \; Q^{\pi_\theta}(s,a)\right]}$$

Policy Gradient: Problems & Solutions

- Vanilla Policy Gradient
  - Unbiased but very nosy

Policy Gradient: Problems & Solutions

- Vanilla Policy Gradient
  - Unbiased but very nosy
  - Requires lots of samples to make it work

Policy Gradient: Problems & Solutions

- Vanilla Policy Gradient
  - Unbiased but very nosy
  - Requires lots of samples to make it work
- Solutions:
  - Baseline
  - Temporal Structure
  - Other (e.g. KL trust region)

Temporal structure

$$\hat{g} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\alpha_t^{(i)} \mid s_t^{(i)}) R(\tau^{(i)})$$
$$= \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\alpha_t^{(i)} \mid s_t^{(i)}) \left(\sum_{k=0}^{H} R(s_k^{(i)}, a_k^{(i)}) \right)$$

Each action takes the blame for the full trajectory!

Temporal structure

$$\hat{g} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)}) R(\tau^{(i)})$$

$$= \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)}) \left(\sum_{k=0}^{H} R(s_k^{(i)}, a_k^{(i)})\right)$$

$$= \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)}) \left(\sum_{k=0}^{t-1} R(s_k^{(i)}, a_k^{(i)}) + \sum_{k=t}^{H} R(s_k^{(i)}, a_k^{(i)})\right)$$

Each action takes the blame for the full trajectory!

These rewards are not caused by actions that come after t

Temporal structure

$$\hat{g} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\alpha_t^{(i)} \mid s_t^{(i)}) R(\tau^{(i)})$$

$$= \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\alpha_t^{(i)} \mid s_t^{(i)}) \left(\sum_{k=0}^{H} R(s_k^{(i)}, a_k^{(i)}) \right)$$

$$= \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\alpha_t^{(i)} \mid s_t^{(i)}) \left(\sum_{k=0}^{t-1} R(s_k^{(i)}, a_k^{(i)}) + \sum_{k=t}^{H} R(s_k^{(i)}, a_k^{(i)}) \right)$$

Each action takes the blame for the full trajectory!

Consider instead:

$$\hat{g} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\alpha_t^{(i)} \mid s_t^{(i)}) \left(\sum_{k=t}^{H} R(s_k^{(i)}, a_k^{(i)}) \right)$$

Each action takes the blame for the trajectory that comes after it

We can call this the return from t onwards G_t

Likelihood ration gradient estimator

- Let's analyze the update:

$$\Delta\theta_t = \alpha G_t \nabla_\theta \log \pi_\theta(s_t, a_t)$$

- It can be further rewritten as follows:

$$\theta_{t+1} = \theta_t + \alpha G_t \frac{\nabla_\theta \pi(A_t|S_t, \theta)}{\pi(A_t|S_t, \theta}$$

- Update is proportional to:
  - the product of a return G_t
  - the gradient of the probability of taking the action actually taken
  - divided by the probability of taking that action.

Likelihood ration gradient estimator

- Let's analyze the update:

$$\Delta\theta_t = \alpha G_t \nabla_\theta \log \pi_\theta(s_t, a_t)$$

- It can be further rewritten as follows:

$$\theta_{t+1} = \theta_t + \alpha G_t \frac{\nabla_\theta \pi(A_t|S_t,\theta) \text{①}}{\pi(A_t|S_t,\theta) \text{②}}$$

① Move most in the directions that favor actions that yield the highest return

Likelihood ration gradient estimator

- Let's analyze the update:

$$\Delta\theta_t = \alpha G_t \nabla_\theta \log \pi_\theta(s_t, a_t)$$

- It can be further rewritten as follows:

$$\theta_{t+1} = \theta_t + \alpha G_t \frac{\nabla_\theta \pi(A_t|S_t,\theta) ①}{\pi(A_t|S_t,\theta) ②}$$

① Move most in the directions that favor actions that yield the highest return

② Update is inversely proportional to the action probability to fight the fact that actions that are selected frequently are at an advantage (the updates will be more often in their direction)

Policy Gradient with Baseline

- Note that in general:

$$E[b\nabla_\theta \log \pi(A_t|S_t)] = E\left[\sum_a \pi(a|S_t) b \nabla_\theta \log \pi(a|S_t)\right]$$

Policy Gradient with Baseline

- Note that in general:

$$E[b\nabla_\theta \log \pi(A_t|S_t)] = E\left[\sum_a \pi(a|S_t) b\nabla_\theta \log \pi(a|S_t)\right]$$

$$= E\left[b\nabla_\theta \sum_a \pi(a|S_t)\right]$$

Policy Gradient with Baseline

- Note that in general:

$$
\begin{aligned}
E[b\nabla_\theta \log \pi(A_t|S_t)] &= E\left[\sum_a \pi(a|S_t) b \nabla_\theta \log \pi(a|S_t)\right] \\
&= E\left[b\nabla_\theta \sum_a \pi(a|S_t)\right] \\
&= E[b\nabla_\theta 1]
\end{aligned}
$$

Policy Gradient with Baseline

- Note that in general:

$$
\begin{aligned}
E[b\nabla_\theta \log \pi(A_t|S_t)] &= E\left[\sum_a \pi(a|S_t) b \nabla_\theta \log \pi(a|S_t)\right] \\
&= E\left[b \nabla_\theta \sum_a \pi(a|S_t)\right] \\
&= E[b\nabla_\theta 1] \\
&= 0
\end{aligned}
$$

- Thus gradient remains unchanged with the additional term

Policy Gradient with Baseline

- Note that in general:

$$\begin{aligned} E[b\nabla_\theta \log \pi(A_t|S_t)] &= E\left[\sum_a \pi(a|S_t) b \nabla_\theta \log \pi(a|S_t)\right] \\ &= E\left[b\nabla_\theta \sum_a \pi(a|S_t)\right] \\ &= E[b\nabla_\theta 1] \\ &= 0 \end{aligned}$$

- Thus gradient remains unchanged with the additional term
- This holds only if b does not depend on the action (though it can depend on the state)

Policy Gradient with Baseline

- Note that in general:

$$
\begin{aligned}
E[b\nabla_\theta \log \pi(A_t|S_t)] &= E\left[\sum_a \pi(a|S_t) b \nabla_\theta \log \pi(a|S_t)\right] \\
&= E\left[b \nabla_\theta \sum_a \pi(a|S_t)\right] \\
&= E[b \nabla_\theta 1] \\
&= 0
\end{aligned}
$$

- Thus gradient remains unchanged with the additional term
- This holds only if b does not depend on the action (though it can depend on the state)
- Implies, we can substract a baseline to reduce variance

Policy Gradient with Baseline

- What is we substruct b from the rewards?

$$\nabla_\theta J(\theta) = E[\nabla_\theta \log \pi(A_t|S_t)(Q_\pi(S_t, A_t) - b]$$

Policy Gradient with Baseline

- What is we substruct b from the rewards?

$$\nabla_\theta J(\theta) = E[\nabla_\theta \log \pi(A_t|S_t)(Q_\pi(S_t, A_t) - b]$$

- A good baseline is $V_\pi(S_t)$

$$\nabla_\theta J(\theta) = E[\nabla_\theta \log \pi(A_t|S_t)(Q_\pi(S_t, A_t) - V_\pi(S_t)]$$

Policy Gradient with Baseline

- What is we substruct b from the rewards?

$$\nabla_\theta J(\theta) = E[\nabla_\theta \log \pi(A_t|S_t)(Q_\pi(S_t, A_t) - b]$$

- A good baseline is $V_\pi(S_t)$

$$\begin{aligned}\nabla_\theta J(\theta) &= E[\nabla_\theta \log \pi(A_t|S_t)(Q_\pi(S_t, A_t) - V_\pi(S_t)] \\ &= E[\nabla_\theta \log \pi(A_t|S_t)(A_\pi(S_t, A_t)]\end{aligned}$$

- Thus we can rewrite the policy gradient using the advantage function $A_\pi(S_t, A_t)$

REINFORCE with Baseline

REINFORCE with Baseline (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Algorithm parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^{d}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
    Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
        $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
        $\delta \leftarrow G - \hat{v}(S_t,\mathbf{w})$
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S_t,\mathbf{w})$
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \gamma^t \delta \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

Policy gradient methods maximize the expected total reward by repeatedly estimating the gradient $g := \nabla_\theta \mathbb{E}\left[\sum_{t=0}^{\infty} r_t\right]$. There are several different related expressions for the policy gradient, which have the form

$$g = \mathbb{E}\left[\sum_{t=0}^{\infty} \Psi_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right], \tag{1}$$

where Ψ_t may be one of the following:

1. $\sum_{t=0}^{\infty} r_t$: total reward of the trajectory.
2. $\sum_{t'=t}^{\infty} r_{t'}$: reward following action a_t.
3. $\sum_{t'=t}^{\infty} r_{t'} - b(s_t)$: baselined version of previous formula.
4. $Q^\pi(s_t, a_t)$: state-action value function.
5. $A^\pi(s_t, a_t)$: advantage function.
6. $r_t + V^\pi(s_{t+1}) - V^\pi(s_t)$: TD residual.

The latter formulas use the definitions

$$V^\pi(s_t) := \mathbb{E}_{\substack{s_{t+1:\infty},\\ a_{t:\infty}}}\left[\sum_{l=0}^{\infty} r_{t+l}\right] \qquad Q^\pi(s_t, a_t) := \mathbb{E}_{\substack{s_{t+1:\infty},\\ a_{t+1:\infty}}}\left[\sum_{l=0}^{\infty} r_{t+l}\right] \tag{2}$$

$$A^\pi(s_t, a_t) := Q^\pi(s_t, a_t) - V^\pi(s_t), \quad \text{(Advantage function).} \tag{3}$$

*Schulman, John, et al. "High-dimensional continuous control using generalized advantage estimation."