

Deep Reinforcement Learning and Control

REINFORCE, Actor-critic methods

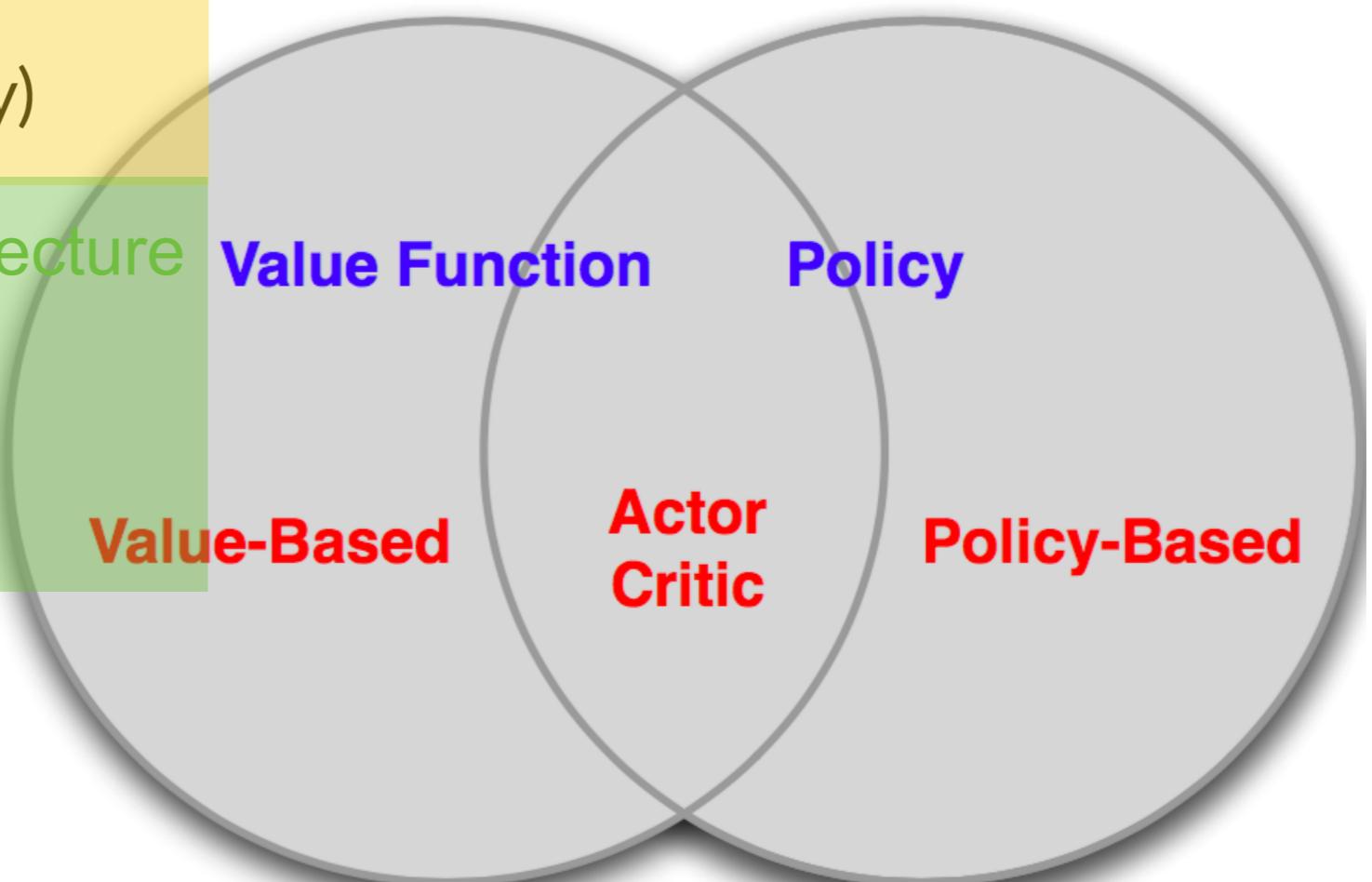
Spring 2021, CMU 10-403

Katerina Fragkiadaki



Value-Based and Policy-Based RL

- Value Based We have covered those
 - Learned Value Function
 - Implicit policy (e.g. ϵ -greedy)
- Policy Based this lecture
 - No Value Function
 - Learned Policy
- Actor-Critic
 - Learned Value Function
 - Learned Policy



Policy Optimization

- Let $U(\theta)$ be any policy objective function
 - 0. Initialize policy parameters θ
 - 1. Sample trajectories $\{\tau_i = \{s_t^i, a_t^i\}_{t=0}^T\}$ by deploying the current policy $\pi_\theta(a_t | s_t)$.
 - 2. Compute gradient vector $\nabla_\theta U(\theta)$
 - 3. $\theta \leftarrow \theta + \alpha \nabla_\theta U(\theta)$
- 
- Policy based reinforcement learning is an optimization problem: Find θ that maximizes $U(\theta)$
 - General alternatives: gradient free optimization
 - Hill climbing
 - Genetic algorithms. We have seen this! Today we will learn different policy gradient estimators.

Policy Optimization

- 
0. Initialize policy parameters θ
 1. Sample trajectories $\{\tau_i = \{s_t^i, a_t^i\}_{t=0}^T\}$ by deploying the current policy $\pi_\theta(a_t | s_t)$.
 2. Compute gradient vector $\nabla_\theta U(\theta)$
 3. $\theta \leftarrow \theta + \alpha \nabla_\theta U(\theta)$

- Advantages over value based methods:
 - Effective in high-dimensional or continuous action spaces
 - Can learn stochastic policies

Policy Gradient

- Let $U(\theta)$ be any policy objective function
- Policy gradient algorithms search for a local maximum in $U(\theta)$ by ascending the gradient of the policy, w.r.t. parameters θ

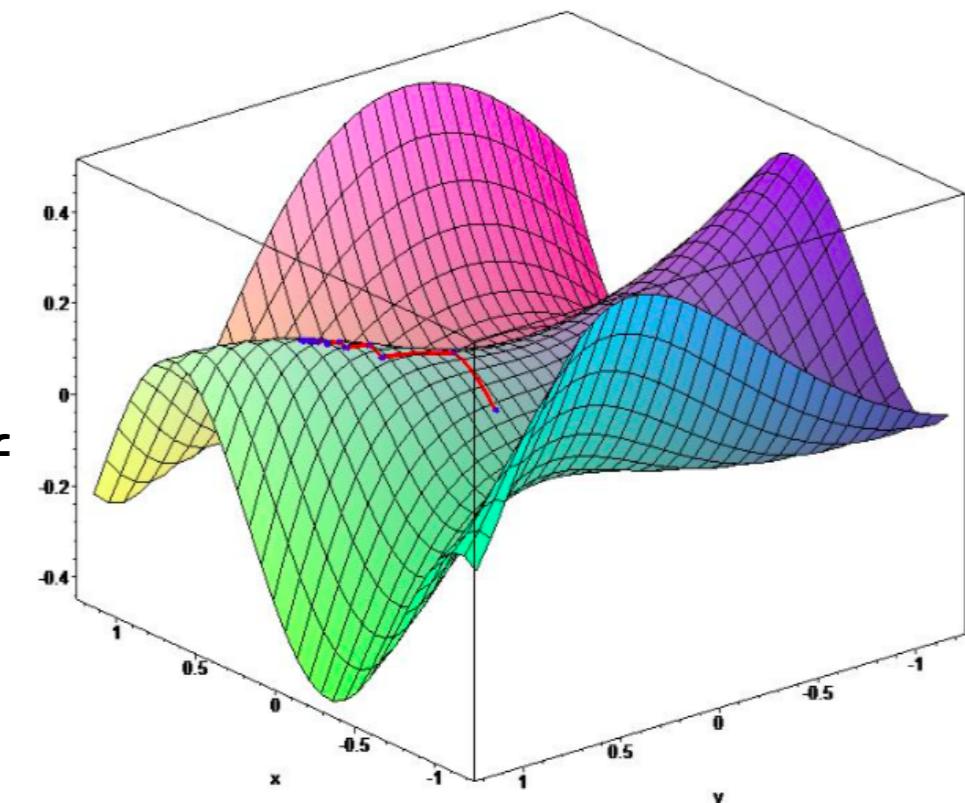
$$\theta_{new} = \theta_{old} + \Delta\theta$$

$$\Delta\theta = \alpha \nabla_{\theta} U(\theta)$$

α is a step-size parameter (learning rate)

is the **policy gradient**

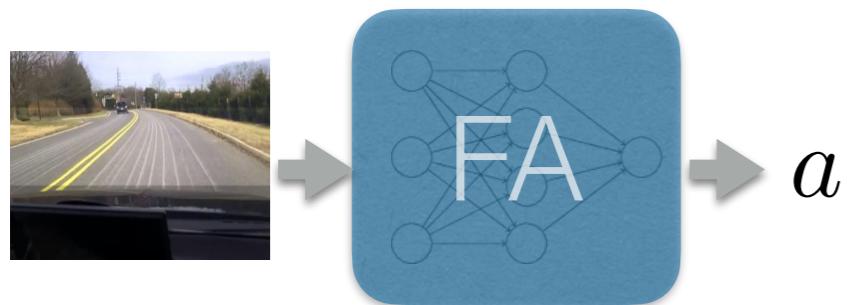
$$\nabla_{\theta} U(\theta) = \begin{pmatrix} \frac{\partial U(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial U(\theta)}{\partial \theta_n} \end{pmatrix}$$



Policy gradient: the gradient of the policy objective w.r.t. the parameters of the policy

Policy functions

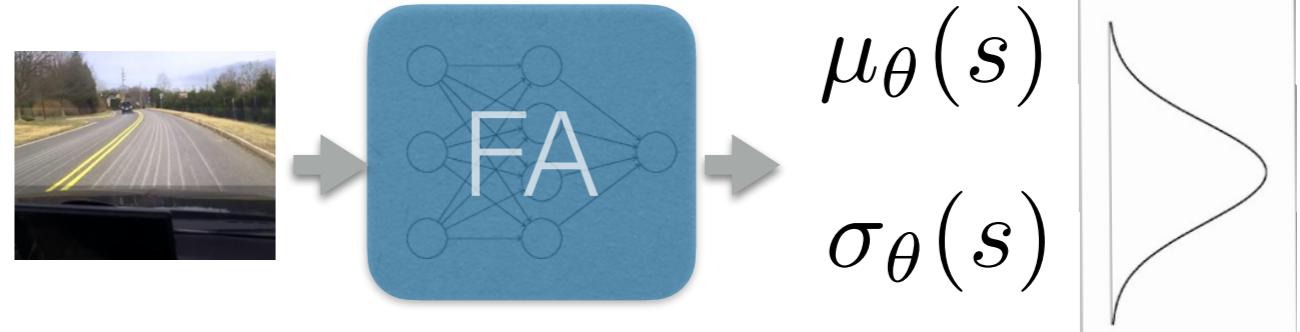
deterministic continuous policy



$$a = \pi_{\theta}(s)$$

e.g. outputs a steering angle directly

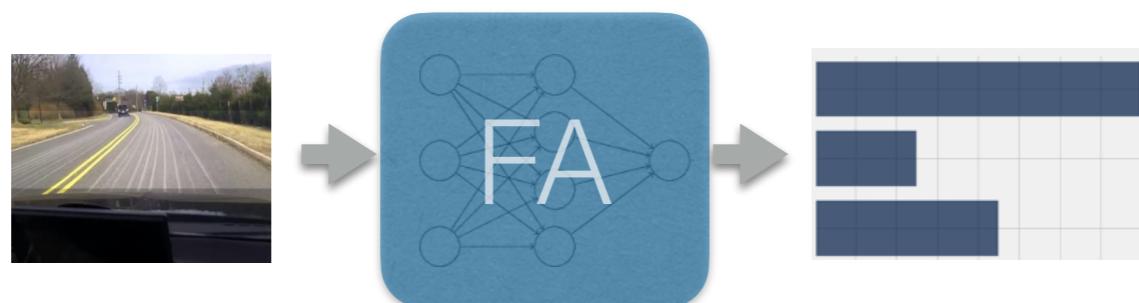
stochastic continuous policy



$$a \sim \mathcal{N}(\mu_{\theta}(s), \sigma_{\theta}^2(s))$$

FA for stochastic multimodal continuous policies is an active area of research

(stochastic) policy over discrete actions

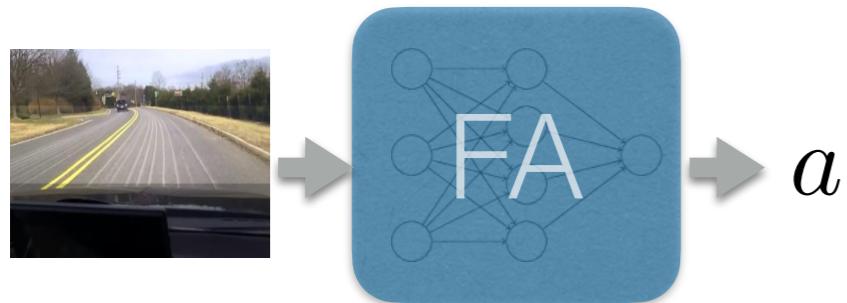


go left
go right
press brake

Outputs a distribution over a discrete set of actions

Policy functions

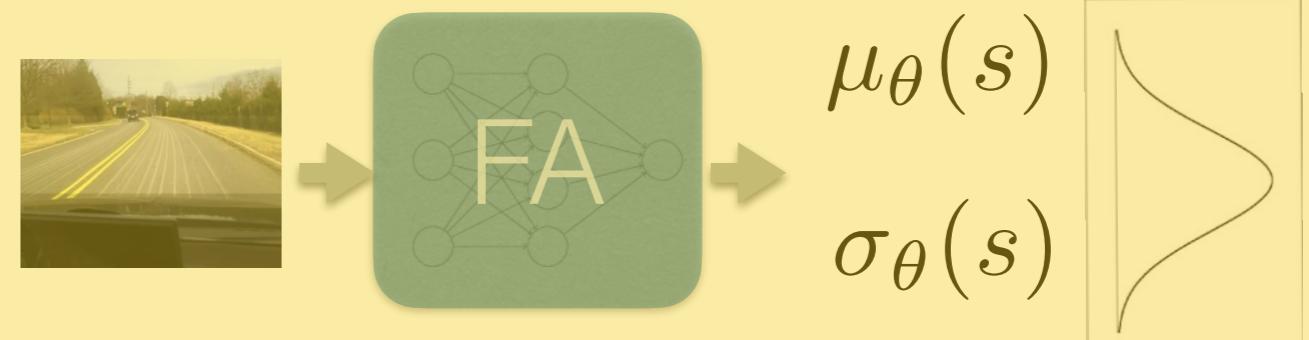
deterministic continuous policy



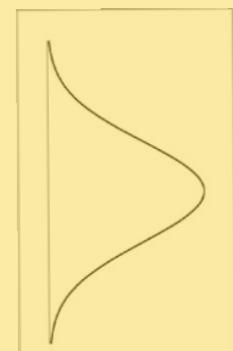
$$a = \pi_\theta(s)$$

e.g. outputs a steering angle directly

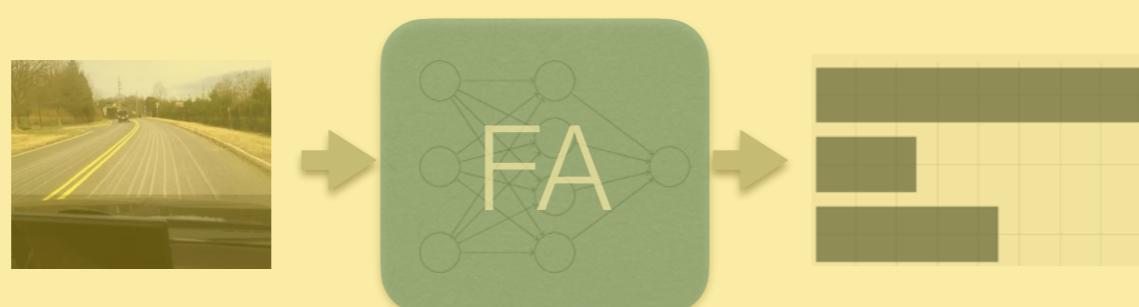
stochastic continuous policy



$$a \sim \mathcal{N}(\mu_\theta(s), \sigma_\theta^2(s))$$



(stochastic) policy over discrete actions



go left
go right
press brake

Outputs a distribution over a discrete set of actions

Policy objective

Trajectory τ is a state action sequence $s_0, a_0, s_1, a_1, \dots, s_H, a_H$

Trajectory reward: $R(\tau) = \sum_{t=0}^H R(s_t, a_t)$

A reasonable policy objective then is:

$$\max_{\theta} . \quad U(\theta) = \mathbb{E}_{\tau \sim P(\tau; \theta)} [R(\tau)] = \sum_{\tau} P(\tau; \theta) R(\tau)$$

Probability of a trajectory: $P(\tau; \theta) = \prod_{t=0}^H \underbrace{P(s_{t+1} | s_t, a_t)}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t | s_t)}_{\text{policy}}$

Our problem is to compute $\nabla_{\theta} U(\theta) = \nabla_{\theta} \mathbb{E}_{\tau \sim P(\tau; \theta)} [R(\tau)]$

Policy objective

$$U(\theta) = \mathbb{E}_{\tau \sim P(\tau; \theta)}[R(\tau)]$$

Computing derivatives of expectations w.r.t. variables that parametrize the distribution, not the quantity inside the expectation

$$\max_{\theta} . \quad \mathbb{E}_{x \sim P(x; \theta)} f(x)$$

Assumptions:

- P is a probability density function that is continuous and differentiable
- P is easy to sample from

Derivatives of expectations

$$\nabla_{\theta} \mathbb{E}_x f(x) = \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)]$$

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\ &= \nabla_{\theta} \sum_x P_{\theta}(x) f(x)\end{aligned}$$

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\&= \nabla_{\theta} \sum_x P_{\theta}(x) f(x) \\&= \sum_x \nabla_{\theta} P_{\theta}(x) f(x)\end{aligned}$$

Why?

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\ &= \nabla_{\theta} \sum_x P_{\theta}(x) f(x) \\ &= \sum_x \nabla_{\theta} P_{\theta}(x) f(x)\end{aligned}$$

What is the problem here?

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\&= \nabla_{\theta} \sum_x P_{\theta}(x) f(x) \\&= \sum_x \nabla_{\theta} P_{\theta}(x) f(x) \\&= \sum_x P_{\theta}(x) \frac{\nabla_{\theta} P_{\theta}(x)}{P_{\theta}(x)} f(x)\end{aligned}$$

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\&= \nabla_{\theta} \sum_x P_{\theta}(x) f(x) \\&= \sum_x \nabla_{\theta} P_{\theta}(x) f(x) \\&= \sum_x P_{\theta}(x) \frac{\nabla_{\theta} P_{\theta}(x)}{P_{\theta}(x)} f(x) \\&= \sum_x P_{\theta}(x) \nabla_{\theta} \log P_{\theta}(x) f(x)\end{aligned}$$

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\&= \nabla_{\theta} \sum_x P_{\theta}(x) f(x) \\&= \sum_x \nabla_{\theta} P_{\theta}(x) f(x) \\&= \sum_x P_{\theta}(x) \frac{\nabla_{\theta} P_{\theta}(x)}{P_{\theta}(x)} f(x) \\&= \sum_x P_{\theta}(x) \nabla_{\theta} \log P_{\theta}(x) f(x) \\&= \mathbb{E}_{x \sim P_{\theta}(x)} [\nabla_{\theta} \log P_{\theta}(x) f(x)]\end{aligned}$$

What have we achieved?

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\&= \nabla_{\theta} \sum_x P_{\theta}(x) f(x) \\&= \sum_x \nabla_{\theta} P_{\theta}(x) f(x) \\&= \sum_x P_{\theta}(x) \frac{\nabla_{\theta} P_{\theta}(x)}{P_{\theta}(x)} f(x) \\&= \sum_x P_{\theta}(x) \nabla_{\theta} \log P_{\theta}(x) f(x) \\&= \mathbb{E}_{x \sim P_{\theta}(x)} [\nabla_{\theta} \log P_{\theta}(x) f(x)]\end{aligned}$$

I can obtain an unbiased estimator for the gradient $\nabla_{\theta} \mathbb{E}_x f(x)$ by sampling!

From the law of large numbers, it will converge to the right gradient with infinite number of samples

Derivatives of expectations

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_x f(x) &= \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \\&= \nabla_{\theta} \sum_x P_{\theta}(x) f(x) \\&= \sum_x \nabla_{\theta} P_{\theta}(x) f(x) \\&= \sum_x P_{\theta}(x) \frac{\nabla_{\theta} P_{\theta}(x)}{P_{\theta}(x)} f(x) \\&= \sum_x P_{\theta}(x) \nabla_{\theta} \log P_{\theta}(x) f(x) \\&= \mathbb{E}_{x \sim P_{\theta}(x)} [\nabla_{\theta} \log P_{\theta}(x) f(x)] \\&\approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(x^{(i)}) f(x^{(i)})\end{aligned}$$

I can obtain an unbiased estimator for the gradient $\nabla_{\theta} \mathbb{E}_x f(x)$ by sampling!

From the law of large numbers, it will converge to the right gradient with infinite number of samples

Derivatives of the policy objective

$$\max_{\theta} . \ U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

Derivatives of the policy objective

$$\max_{\theta} . \quad U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

$$\begin{aligned}\nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) R(\tau)\end{aligned}$$

Derivatives of the policy objective

$$\max_{\theta} . \ U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

$$\begin{aligned}\nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \textcolor{blue}{\nabla_{\theta} P_{\theta}(\tau)} R(\tau)\end{aligned}$$

$$\nabla_{\theta} P(\tau^{(i)}; \theta) = \nabla_{\theta} \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right]$$

$$\begin{aligned}
\nabla_{\theta} P(\tau^{(i)}; \theta) &= \nabla_{\theta} \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\
&= \prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \nabla_{\theta} \left[\prod_{t=0}^T \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\
&= \text{const.} \cdot \nabla_{\theta} \left[\prod_{t=0}^T \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \right]
\end{aligned}$$

Derivatives of the policy objective

$$\max_{\theta} . \quad U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

$$\begin{aligned}\nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \textcolor{blue}{\nabla_{\theta} P_{\theta}(\tau)} R(\tau)\end{aligned}$$

Derivatives of the policy objective

$$\max_{\theta} . \quad U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

$$\begin{aligned}\nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \frac{\nabla_{\theta} P_{\theta}(\tau)}{P_{\theta}(\tau)} R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \nabla_{\theta} \log P_{\theta}(\tau) R(\tau) \\ &= \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [\nabla_{\theta} \log P_{\theta}(\tau) R(\tau)]\end{aligned}$$

Derivatives of the policy objective

$$\max_{\theta} . \quad U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

$$\begin{aligned}\nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \frac{\nabla_{\mu} P_{\theta}(\tau)}{P_{\theta}(\tau)} R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \nabla_{\theta} \log P_{\theta}(\tau) R(\tau) \\ &= \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [\nabla_{\theta} \log P_{\theta}(\tau) R(\tau)]\end{aligned}$$

Approximate the gradient with empirical estimate from N sampled trajectories:

$$\nabla_{\theta} U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)})$$

Derivatives of the policy objective

$$\max_{\theta} . \quad U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

$$\begin{aligned}\nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \frac{\nabla_{\mu} P_{\theta}(\tau)}{P_{\theta}(\tau)} R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \nabla_{\theta} \log P_{\theta}(\tau) R(\tau) \\ &= \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [\nabla_{\theta} \log P_{\theta}(\tau) R(\tau)]\end{aligned}$$

Approximate the gradient with empirical estimate from N sampled trajectories:

$$\nabla_{\theta} U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)})$$

From trajectories to actions

$$\nabla_{\theta} \log P(\tau^{(i)}; \theta) = \nabla_{\theta} \log \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right]$$

From trajectories to actions

$$\nabla_{\theta} \log P(\tau^{(i)}; \theta) = \nabla_{\theta} \log \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right]$$
$$= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} + \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right]$$

From trajectories to actions

$$\begin{aligned}\nabla_{\theta} \log P(\tau^{(i)}; \theta) &= \nabla_{\theta} \log \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} + \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right]\end{aligned}$$

From trajectories to actions

$$\begin{aligned}\nabla_{\theta} \log P(\tau^{(i)}; \theta) &= \nabla_{\theta} \log \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} + \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})\end{aligned}$$

From trajectories to actions

$$\begin{aligned}\nabla_{\theta} \log P(\tau^{(i)}; \theta) &= \nabla_{\theta} \log \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} + \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})\end{aligned}$$

$$\nabla_{\theta} U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)}) \quad \rightarrow$$

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

From trajectories to actions

$$\begin{aligned}\nabla_{\theta} \log P(\tau^{(i)}; \theta) &= \nabla_{\theta} \log \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} + \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})\end{aligned}$$

Let's call \hat{g} the approximate gradient vector

$$\nabla_{\theta} U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)}) \quad \rightarrow$$

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

From trajectories to actions

$$\begin{aligned}\nabla_{\theta} \log P(\tau^{(i)}; \theta) &= \nabla_{\theta} \log \left[\prod_{t=0}^T \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} \cdot \underbrace{\pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log P(s_{t+1}^{(i)} | s_t^{(i)}, a_t^{(i)})}_{\text{dynamics}} + \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \nabla_{\theta} \left[\sum_{t=0}^T \underbrace{\log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\ &= \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)})\end{aligned}$$

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

$\nabla_{\theta} \log \pi_{\theta}(a)$ for Gaussian policy

Variance may be fixed, or can also be parameterized, for now let's fix it.

- For univariate Gaussian $\mathcal{N}(\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\frac{(a-\mu)^2}{\sigma^2}}$

$$\nabla_{\theta} \log \pi_{\theta}(a | s) = \text{const.} \cdot \frac{(a - \mu_{\theta}(s))}{\sigma^2} \nabla_{\theta} \mu_{\theta}(s)$$

- For multivariate Gaussian $\mathcal{N}(\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^k \det \Sigma}} e^{-\frac{1}{2}(\mathbf{a}-\mu)^{\top} \Sigma^{-1} (\mathbf{a}-\mu)}$

$$\nabla_{\theta} \log \pi_{\theta}(a | s) = \text{const.} \cdot \Sigma^{-1}(\mathbf{a} - \mu_{\theta}(s)) \frac{\partial(\mu_1 \dots \mu_k)}{\partial(\theta_1 \dots \theta_n)}$$

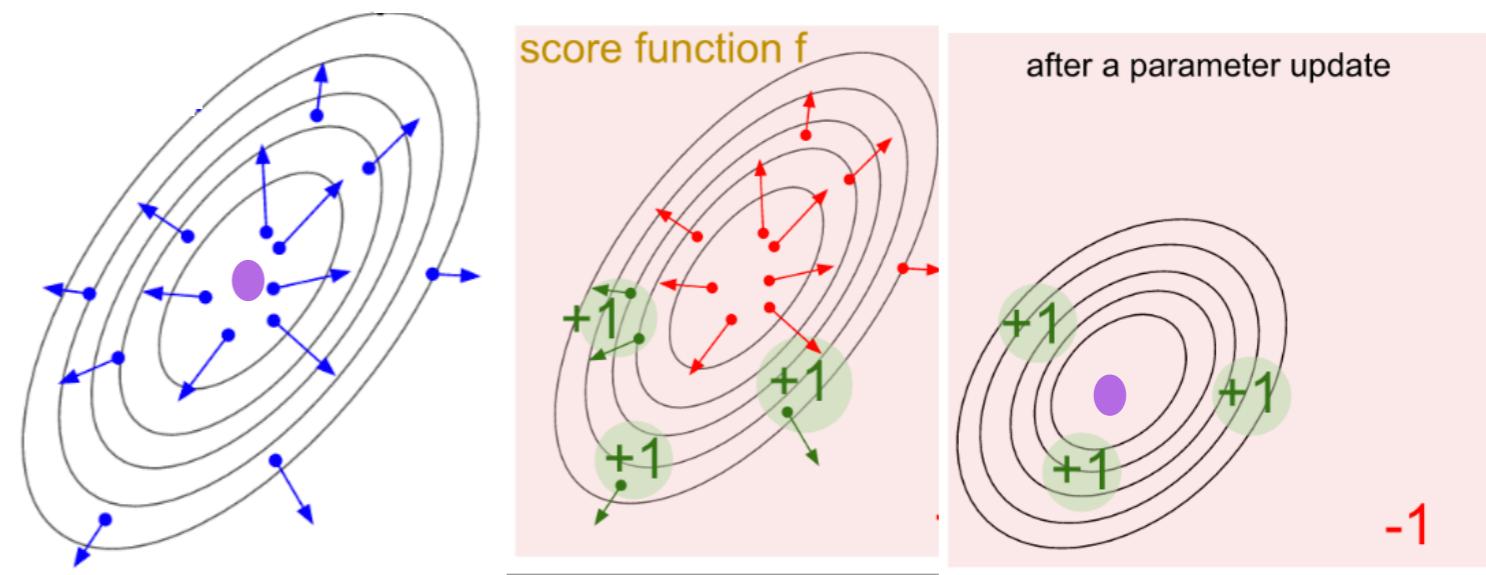
Two-dimensional Gaussian $P_\theta(x)$

$$\begin{aligned}
 \nabla_\theta \mathbb{E}_x f(x) &= \nabla_\theta \mathbb{E}_{x \sim P_\theta(x)} [f(x)] \\
 &= \nabla_\theta \sum_x P_\theta(x) f(x) \\
 &= \sum_x \nabla_\theta P_\theta(x) f(x) \\
 &= \sum_x P_\theta(x) \frac{\nabla_\theta P_\theta(x)}{P_\theta(x)} f(x) \\
 &= \sum_x P_\theta(x) \nabla_\theta \log P_\theta(x) f(x) \\
 &= \mathbb{E}_{x \sim P_\theta(x)} [\nabla_\theta \log P_\theta(x) f(x)] \\
 &\approx \frac{1}{N} \sum_{i=1}^N \nabla_\theta \log P_\theta(x^{(i)}) f(x^{(i)}) \\
 &\approx \frac{1}{N} \sum_{i=1}^N \text{const.} \Sigma^{-1} \cdot (\mathbf{a}^{(i)} - \mu) \begin{bmatrix} 10 \\ 01 \end{bmatrix} f(x^{(i)})
 \end{aligned}$$

Imagine I am optimizing over these two parameters:

$$\mu = [\theta_1, \theta_2]^\top$$

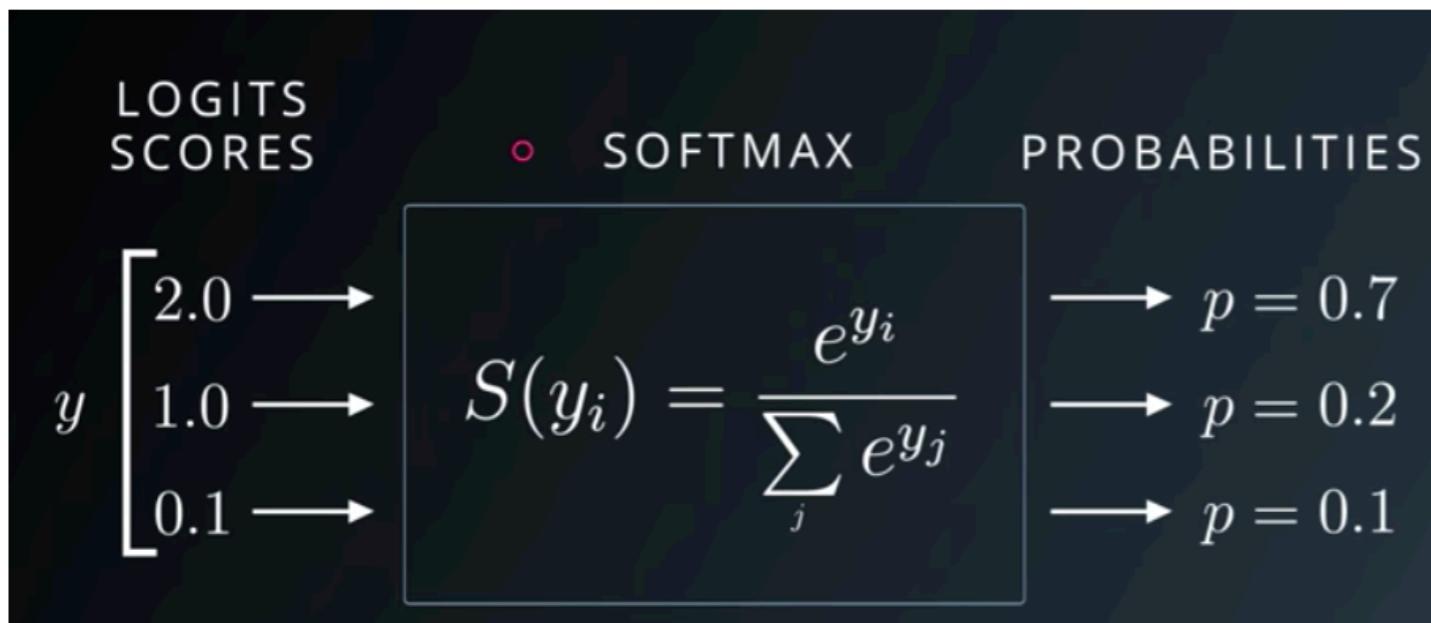
• $\mathbf{a}^{(i)}$
 $\rightarrow \Sigma^{-1}(\mathbf{a}^{(i)} - \mu) \propto \nabla_\theta \log P_\theta(\mathbf{a})$



μ

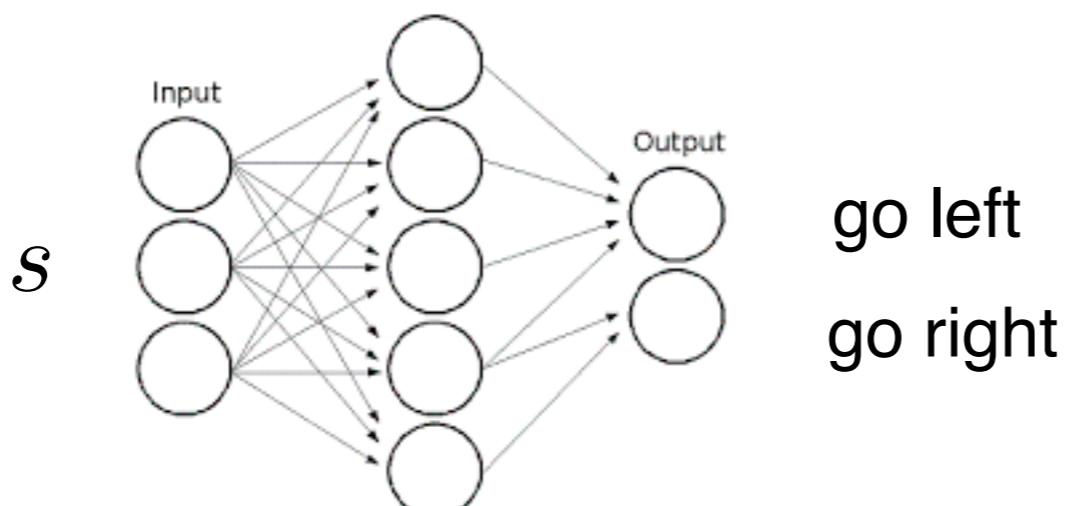
μ'

Softmax



$$\pi(a | s, \theta) \doteq \frac{e^{h(s,a,\theta)}}{\sum_b e^{h(s,b,\theta)}}$$

discrete actions



Output is a distribution over a discrete set of actions

$\nabla_{\theta} \log \pi_{\theta}(a)$ for softmax policy

$$\pi_{\theta}(a \mid s) = \frac{e^{h_{\theta}(s,a)}}{\sum_b e^{h_{\theta}(s,b)}}$$

$$\nabla_{\theta} \log \pi_{\theta}(a) =$$

$\nabla_{\theta} \log \pi_{\theta}(a)$ for softmax policy

$$\pi_{\theta}(a | s) = \frac{e^{h_{\theta}(s,a)}}{\sum_b e^{h_{\theta}(s,b)}}$$

$$\nabla_{\theta} \log \pi_{\theta}(a) = \nabla_{\theta} h_{\theta}(s, a) - \nabla_{\theta} \log \sum_b e^{h_{\theta}(s,b)}$$

$\nabla_{\theta} \log \pi_{\theta}(a)$ for softmax policy

$$\pi_{\theta}(a | s) = \frac{e^{h_{\theta}(s,a)}}{\sum_b e^{h_{\theta}(s,b)}}$$

$$\begin{aligned}\nabla_{\theta} \log \pi_{\theta}(a) &= \nabla_{\theta} h_{\theta}(s, a) - \nabla_{\theta} \log \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \nabla_{\theta} \sum_b e^{h_{\theta}(s,b)}\end{aligned}$$

$\nabla_{\theta} \log \pi_{\theta}(a)$ for softmax policy

$$\pi_{\theta}(a | s) = \frac{e^{h_{\theta}(s,a)}}{\sum_b e^{h_{\theta}(s,b)}}$$

$$\begin{aligned}\nabla_{\theta} \log \pi_{\theta}(a) &= \nabla_{\theta} h_{\theta}(s, a) - \nabla_{\theta} \log \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \nabla_{\theta} \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \sum_b \nabla_{\theta} e^{h_{\theta}(s,b)}\end{aligned}$$

$\nabla_{\theta} \log \pi_{\theta}(a)$ for softmax policy

$$\pi_{\theta}(a | s) = \frac{e^{h_{\theta}(s,a)}}{\sum_b e^{h_{\theta}(s,b)}}$$

$$\begin{aligned}\nabla_{\theta} \log \pi_{\theta}(a) &= \nabla_{\theta} h_{\theta}(s, a) - \nabla_{\theta} \log \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \nabla_{\theta} \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \sum_b \nabla_{\theta} e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \sum_b e^{h_{\theta}(s,b)} \nabla_{\theta} h_{\theta}(s, b)\end{aligned}$$

$\nabla_{\theta} \log \pi_{\theta}(a)$ for softmax policy

$$\pi_{\theta}(a | s) = \frac{e^{h_{\theta}(s,a)}}{\sum_b e^{h_{\theta}(s,b)}}$$

$$\begin{aligned}\nabla_{\theta} \log \pi_{\theta}(a) &= \nabla_{\theta} h_{\theta}(s, a) - \nabla_{\theta} \log \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \nabla_{\theta} \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \sum_b \nabla_{\theta} e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \sum_b e^{h_{\theta}(s,b)} \nabla_{\theta} h_{\theta}(s, b) \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \sum_b \frac{e^{h_{\theta}(s,b)}}{\sum_b e^{h_{\theta}(s,b)}} \nabla_{\theta} h_{\theta}(s, b)\end{aligned}$$

$\nabla_{\theta} \log \pi_{\theta}(a)$ for softmax policy

$$\pi_{\theta}(a | s) = \frac{e^{h_{\theta}(s,a)}}{\sum_b e^{h_{\theta}(s,b)}}$$

$$\begin{aligned}\nabla_{\theta} \log \pi_{\theta}(a) &= \nabla_{\theta} h_{\theta}(s, a) - \nabla_{\theta} \log \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \nabla_{\theta} \sum_b e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \sum_b \nabla_{\theta} e^{h_{\theta}(s,b)} \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \frac{1}{\sum_b e^{h_{\theta}(s,b)}} \sum_b e^{h_{\theta}(s,b)} \nabla_{\theta} h_{\theta}(s, b) \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \sum_b \frac{e^{h_{\theta}(s,b)}}{\sum_b e^{h_{\theta}(s,b)}} \nabla_{\theta} h_{\theta}(s, b) \\ &\quad \nabla_{\theta} h_{\theta}(s, a) - \sum_b \pi_{\theta}(s, b) \nabla_{\theta} h_{\theta}(s, b)\end{aligned}$$

Temporal structure

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

Temporal structure

$$\begin{aligned}\hat{g} &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) R(\tau^{(i)}) \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(\sum_{k=0}^T R(s_k^{(i)}, a_k^{(i)}) \right)\end{aligned}$$

Each action takes the blame
for the full trajectory!

Temporal structure

$$\begin{aligned}\hat{g} &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) R(\tau^{(i)}) \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(\sum_{k=0}^T R(s_k^{(i)}, a_k^{(i)}) \right) \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(\sum_{k=0}^{t-1} R(s_k^{(i)}, a_k^{(i)}) + \sum_{k=t}^T R(s_k^{(i)}, a_k^{(i)}) \right)\end{aligned}$$

Each action takes the blame
for the full trajectory!

These rewards are not caused by actions that
come after t

Can we do better than assigning
the cumulative trajectory reward
to every action in the trajectory?

Temporal structure

$$\begin{aligned}\hat{g} &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) R(\tau^{(i)}) \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(\sum_{k=0}^T R(s_k^{(i)}, a_k^{(i)}) \right) \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(\sum_{k=0}^{t-1} R(s_k^{(i)}, a_k^{(i)}) + \sum_{k=t}^T R(s_k^{(i)}, a_k^{(i)}) \right)\end{aligned}$$

Each action takes the blame for the full trajectory!

Consider instead:

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

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

Temporal structure

$$\begin{aligned}\hat{g} &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) R(\tau^{(i)}) \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(\sum_{k=0}^T R(s_k^{(i)}, a_k^{(i)}) \right) \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(\sum_{k=0}^{t-1} R(s_k^{(i)}, a_k^{(i)}) + \sum_{k=t}^T R(s_k^{(i)}, a_k^{(i)}) \right)\end{aligned}$$

Each action takes the blame for the full trajectory!

Consider instead:

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

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

We can call this the return from t onwards G_t

Likelihood ratio gradient estimator

- Let's analyze the update:

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

- Let's us rewrite is as follows: move most in the directions that favor actions that yield the **highest return**

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

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)

REINFORCE (or Monte Carlo Policy Gradient)

- 
0. Initialize policy parameters θ
 1. Sample trajectories $\{\tau_i = \{s_t^i, a_t^i\}_{t=0}^T\}$ by deploying the current policy $\pi_\theta(a_t | s_t)$.
 2. Compute gradient vector $\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t^{(i)} | s_t^{(i)}) G_t^{(i)}$
 3. $\theta \leftarrow \theta + \alpha \nabla_\theta U(\theta)$

Policy Gradients

$$\begin{aligned} \max_{\theta} . \quad U(\theta) &= \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ \nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \frac{\nabla_{\theta} P_{\theta}(\tau)}{P_{\theta}(\tau)} R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \nabla_{\theta} \log P_{\theta}(\tau) R(\tau) \\ &= \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [\nabla_{\theta} \log P_{\theta}(\tau) R(\tau)] \end{aligned}$$

Sample estimate:

$$\nabla_{\theta} U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)})$$

Evolutionary Methods

$$\max_{\mu} . \ U(\mu) = \mathbb{E}_{\theta \sim P_\mu(\theta)} [F(\theta)]$$

$$\begin{aligned}\nabla_\mu U(\mu) &= \nabla_\mu \mathbb{E}_{\theta \sim P_\mu(\theta)} [F(\theta)] \\ &= \nabla_\mu \int P_\mu(\theta) F(\theta) d\theta \\ &= \int \nabla_\mu P_\mu(\theta) F(\theta) d\theta \\ &= \int P_\mu(\theta) \frac{\nabla_\mu P_\mu(\theta)}{P_\mu(\theta)} F(\theta) d\theta \\ &= \int P_\mu(\theta) \nabla_\mu \log P_\mu(\theta) F(\theta) d\theta \\ &= \mathbb{E}_{\theta \sim P_\mu(\theta)} [\nabla_\mu \log P_\mu(\theta) F(\theta)]\end{aligned}$$

Sample estimate:

$$\nabla_\mu U(\mu) \approx \frac{1}{N} \sum_{i=1}^N \nabla_\mu \log P_\mu(\theta^{(i)}) F(\theta^{(i)})$$

Policy gradients VS Evolutionary methods

Considers distribution over actions

$$\max_{\theta} . \quad U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]$$

$$\begin{aligned}\nabla_{\theta} U(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)] \\ &= \nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \frac{\nabla_{\theta} P_{\theta}(\tau)}{P_{\theta}(\tau)} R(\tau) \\ &= \sum_{\tau} P_{\theta}(\tau) \nabla_{\theta} \log P_{\theta}(\tau) R(\tau) \\ &= \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [\nabla_{\theta} \log P_{\theta}(\tau) R(\tau)]\end{aligned}$$

Sample estimate:

$$\nabla_{\theta} U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)})$$

Considers distribution over policy parameters

$$\max_{\mu} . \quad U(\mu) = \mathbb{E}_{\theta \sim P_{\mu}(\theta)} [F(\theta)]$$

$$\begin{aligned}\nabla_{\mu} U(\mu) &= \nabla_{\mu} \mathbb{E}_{\theta \sim P_{\mu}(\theta)} [F(\theta)] \\ &= \nabla_{\mu} \int P_{\mu}(\theta) F(\theta) d\theta \\ &= \int \nabla_{\mu} P_{\mu}(\theta) F(\theta) d\theta \\ &= \int P_{\mu}(\theta) \frac{\nabla_{\mu} P_{\mu}(\theta)}{P_{\mu}(\theta)} F(\theta) d\theta \\ &= \int P_{\mu}(\theta) \nabla_{\mu} \log P_{\mu}(\theta) F(\theta) d\theta \\ &= \mathbb{E}_{\theta \sim P_{\mu}(\theta)} [\nabla_{\mu} \log P_{\mu}(\theta) F(\theta)]\end{aligned}$$

Sample estimate:

$$\nabla_{\mu} U(\mu) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\mu} \log P_{\mu}(\theta^{(i)}) F(\theta^{(i)})$$

Policy gradients VS Evolutionary methods

- We are sampling in both cases...
 - PG: sampling in action space
 - ES: sampling in parameter space

Pong from Pixels

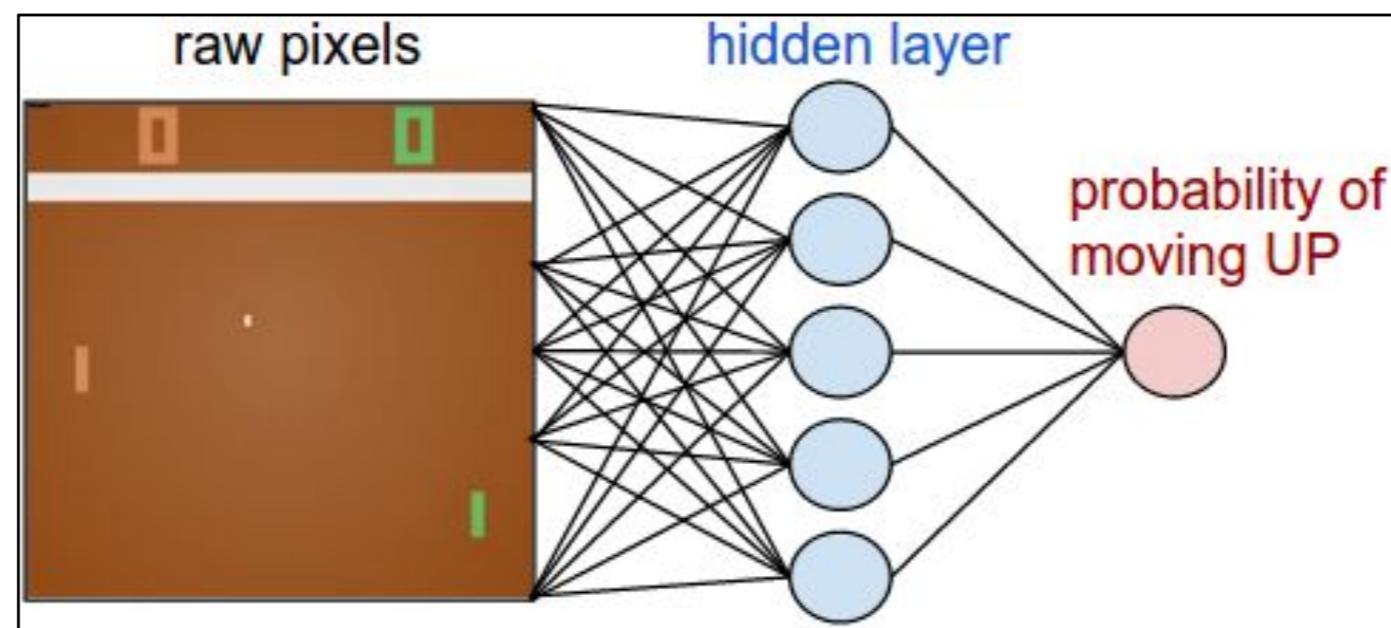
Slides from Andrei Karpathy

|



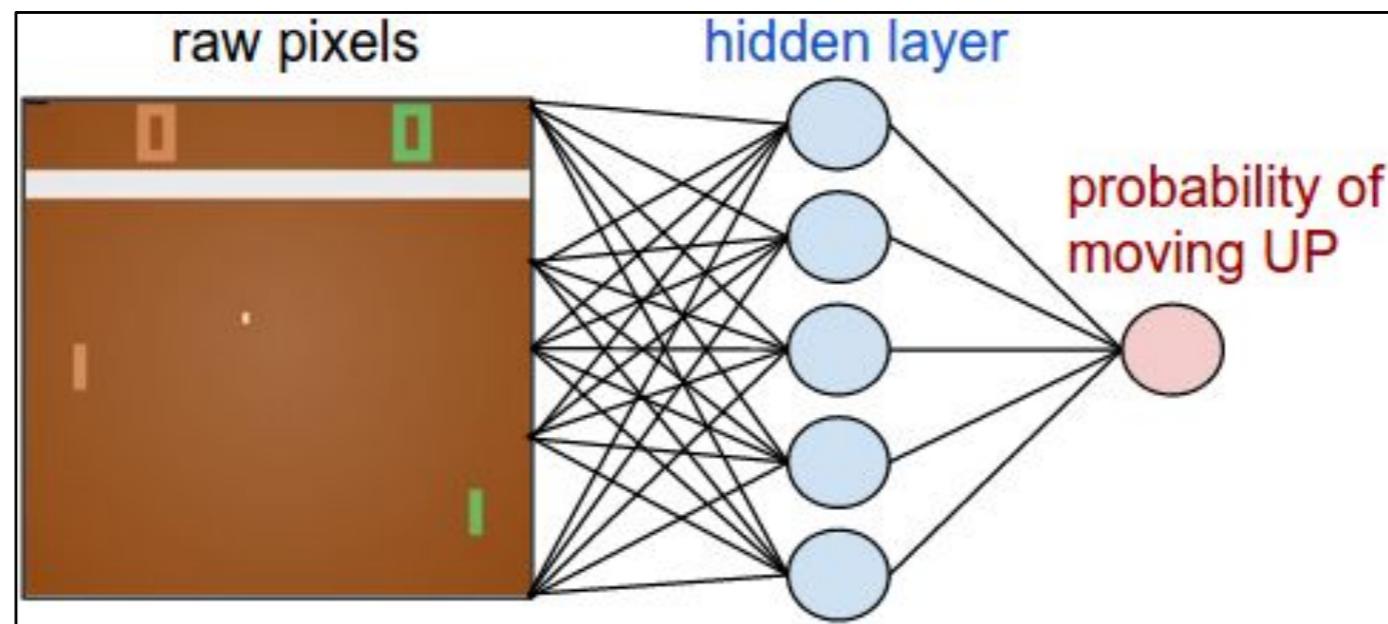
|

Policy network



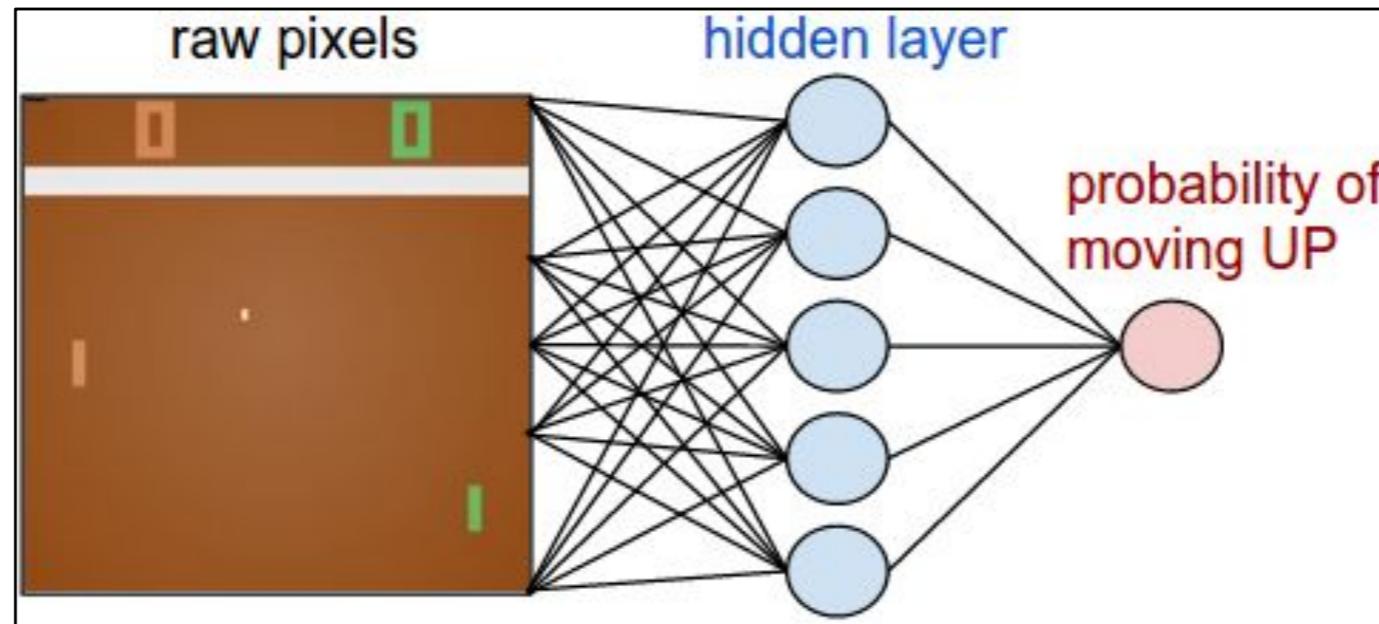
Policy network

e.g.,
height width
[80 x 80]
array



Policy network

height width
[80 x 80]
array

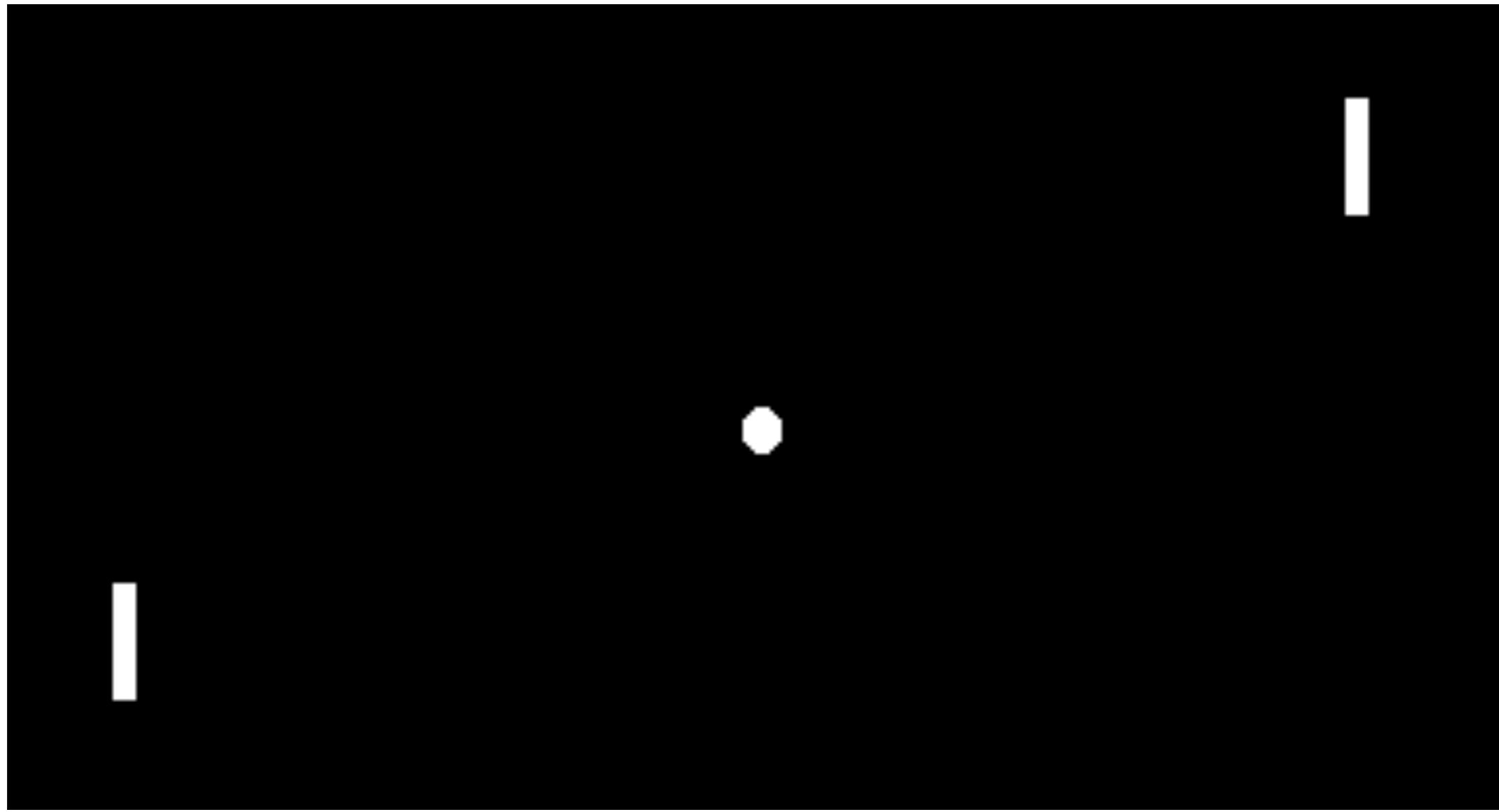


E.g. 200 nodes in the hidden network, so:

$$[(80*80)*200 + 200] + [200*1 + 1] = \sim 1.3M \text{ parameters}$$

Layer 1

Layer 2

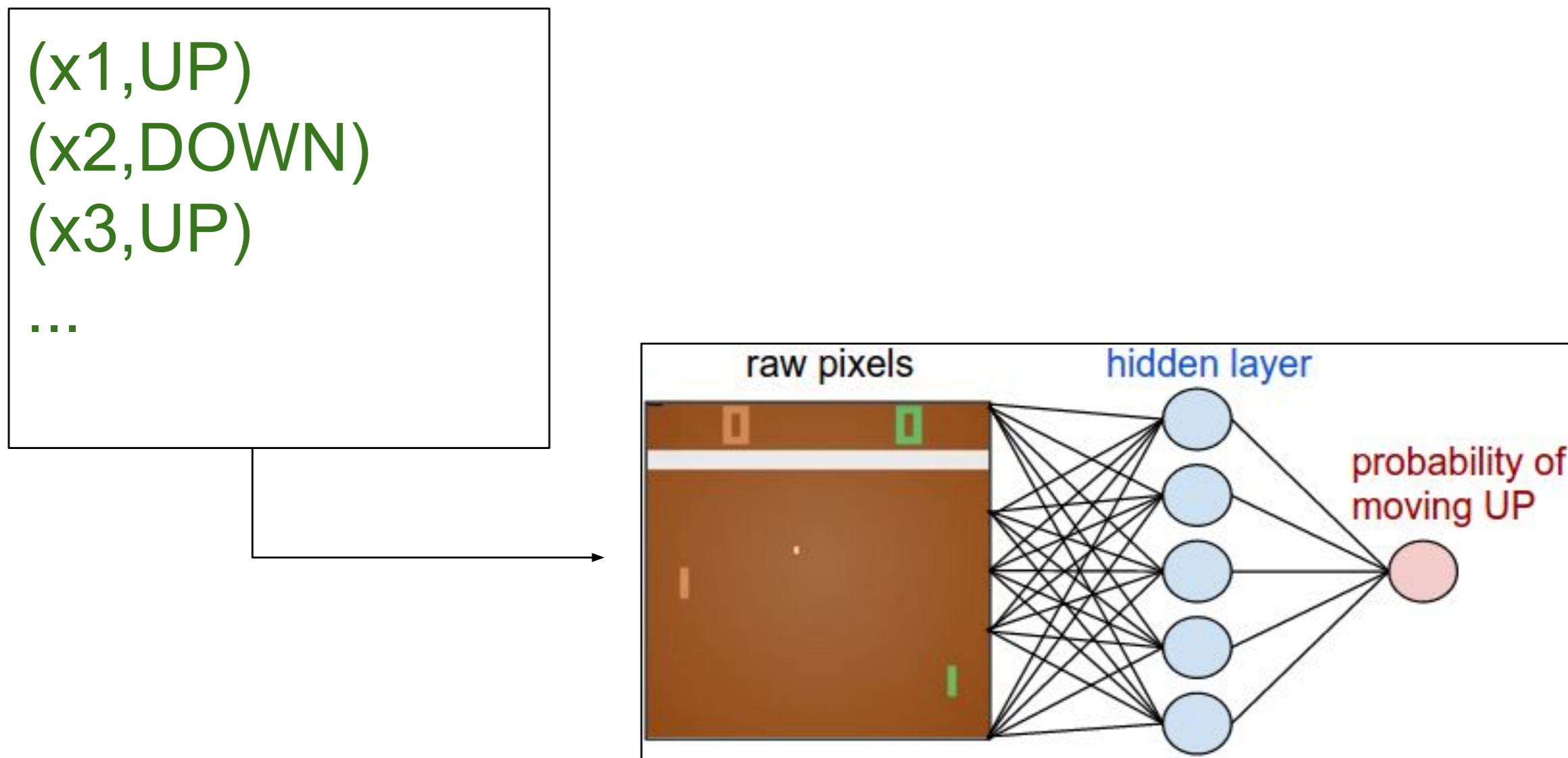


Network does not see this. Network sees $80 \times 80 = 6,400$ numbers.
It gets a reward of +1 or -1, some of the time.
Q: How do we efficiently find a good setting of the 1.3M parameters?

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

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

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

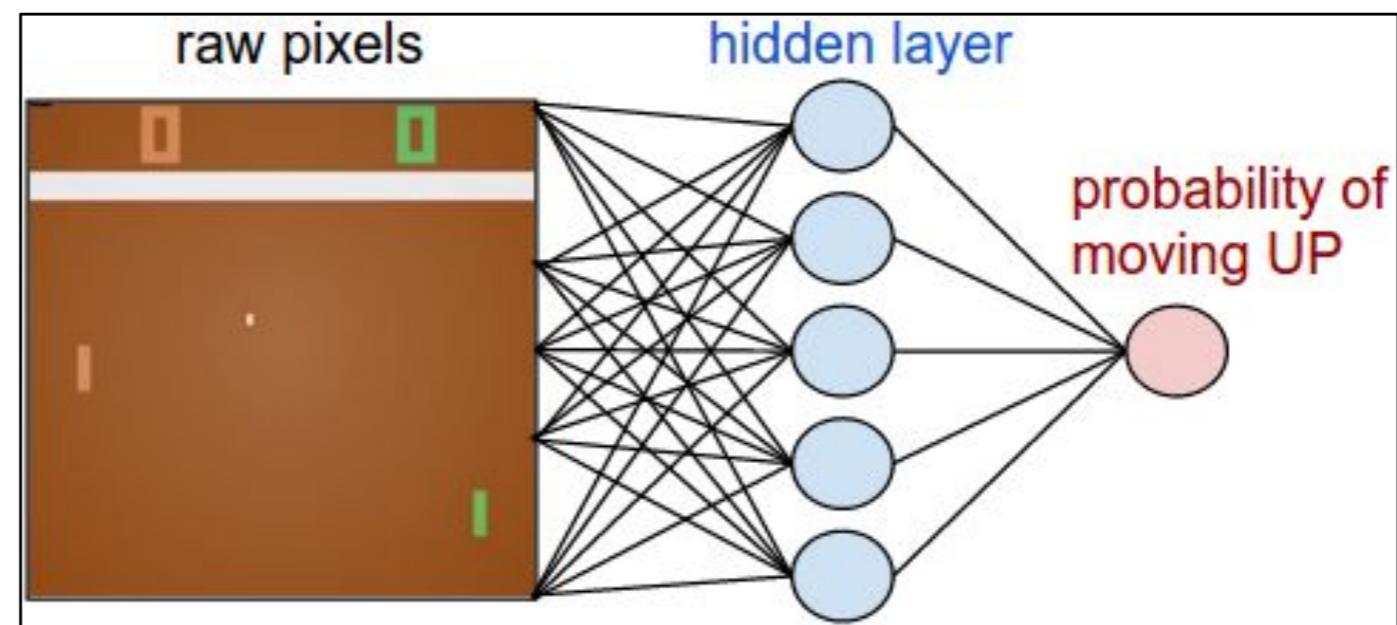


Suppose we had the training labels...
(we know what to do in any state)

(x_1, UP)
 (x_2, DOWN)
 (x_3, UP)
...

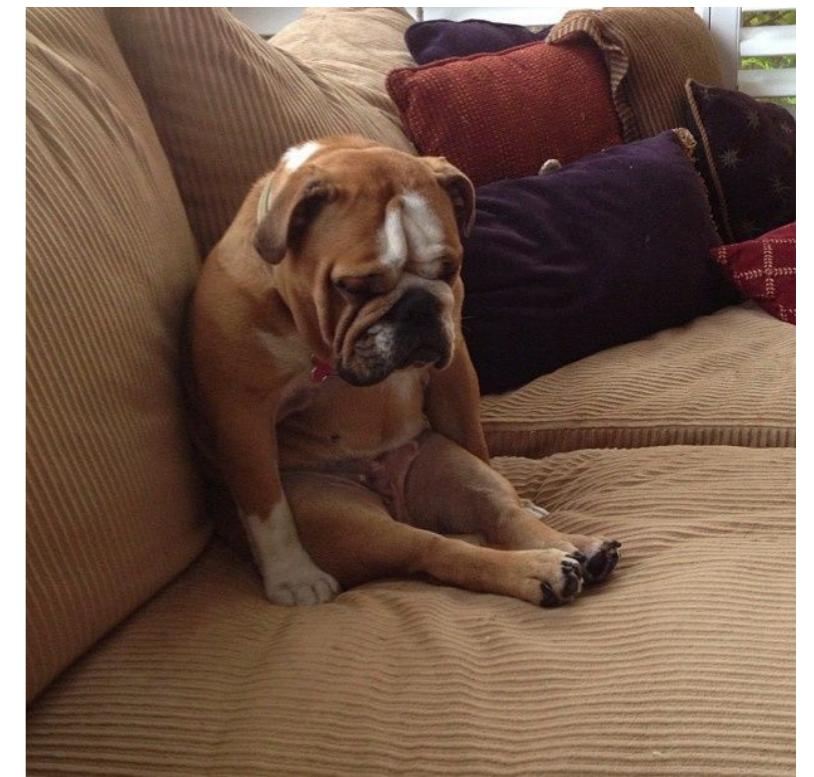
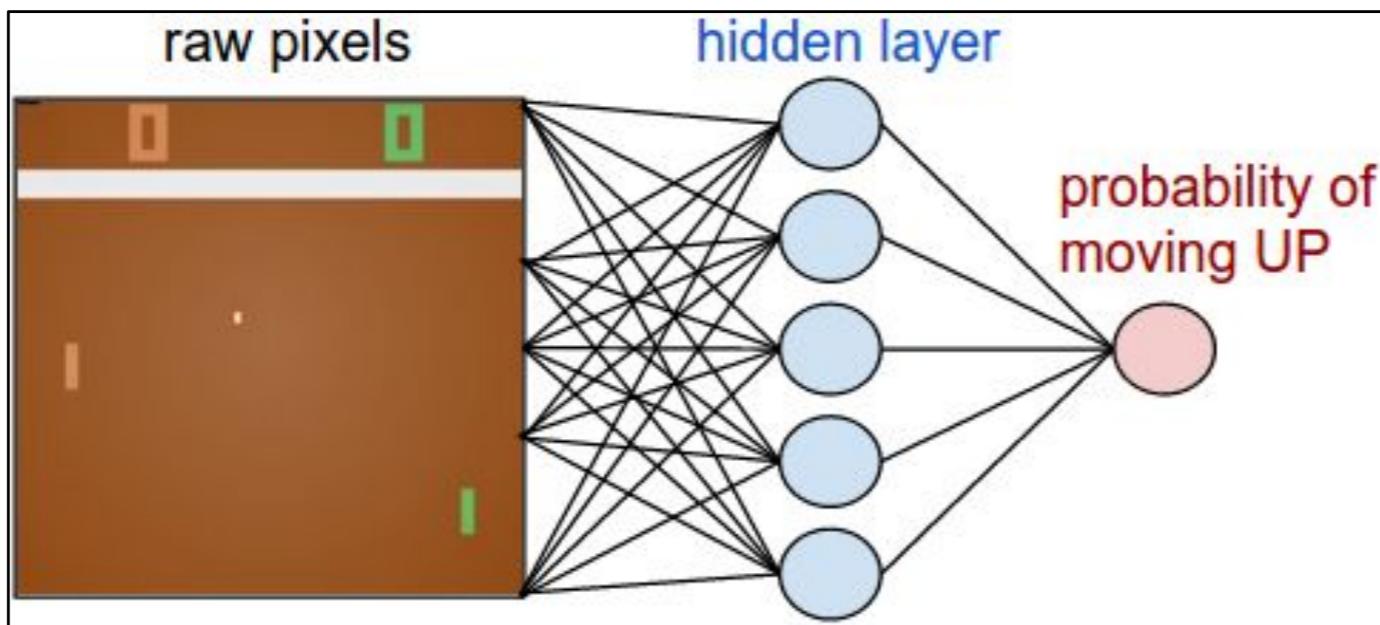
maximize:

$$\sum_i \log p(y_i | x_i)$$



supervised learning

Except, we don't have labels...



Should we go UP or DOWN?

Except, we don't have labels...

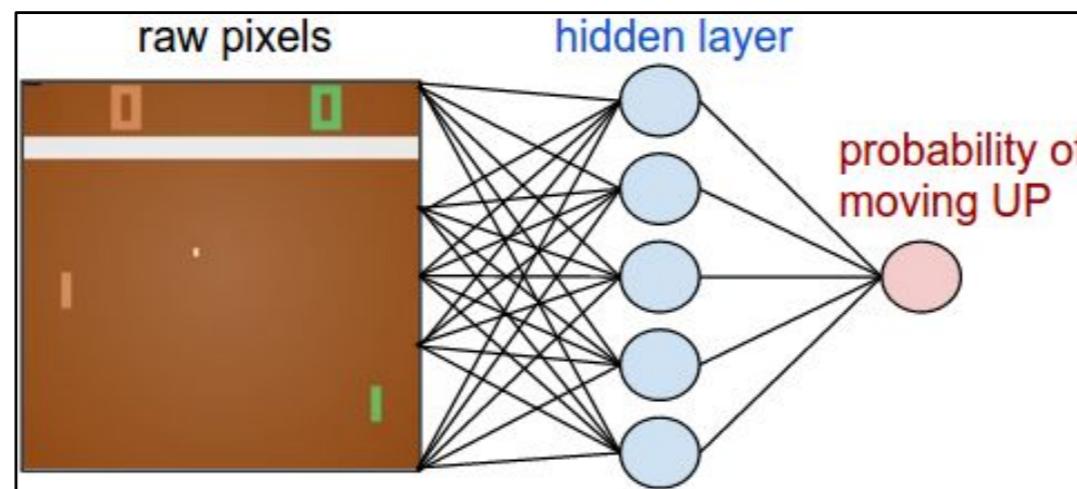


“Try a bunch of stuff and see what happens. Do more of the stuff that worked in the future.”

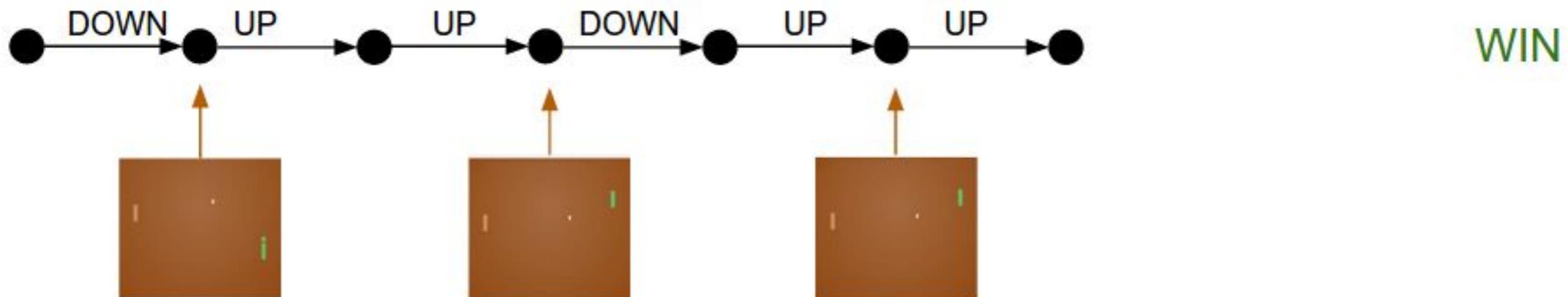
-RL

trial-and-error learning

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

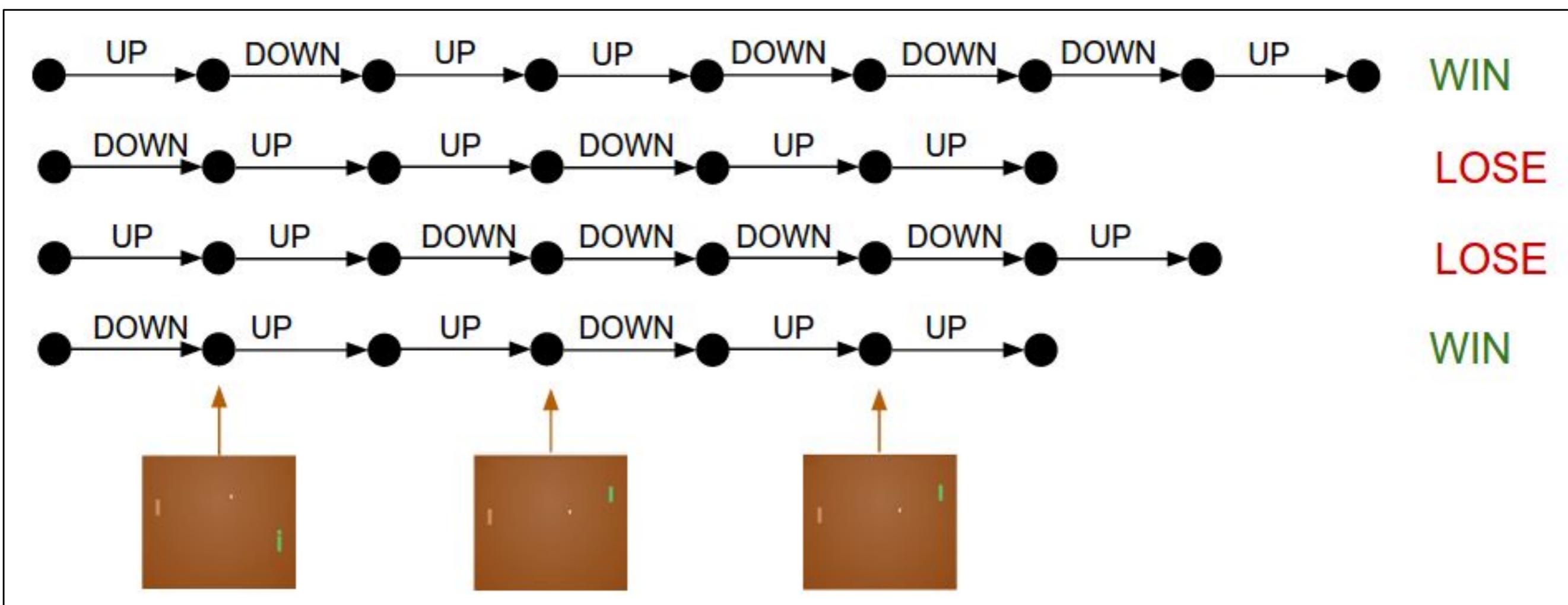


Rollout the policy
and collect an
episode

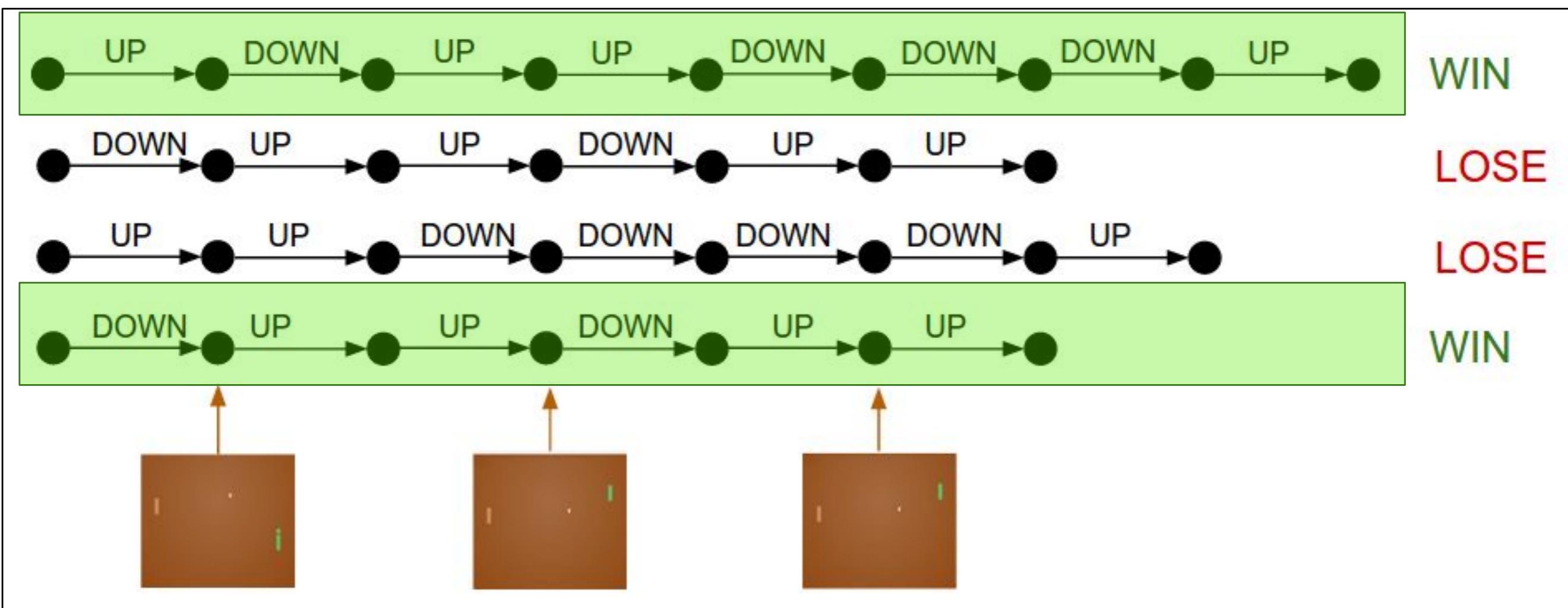


Collect many rollouts...

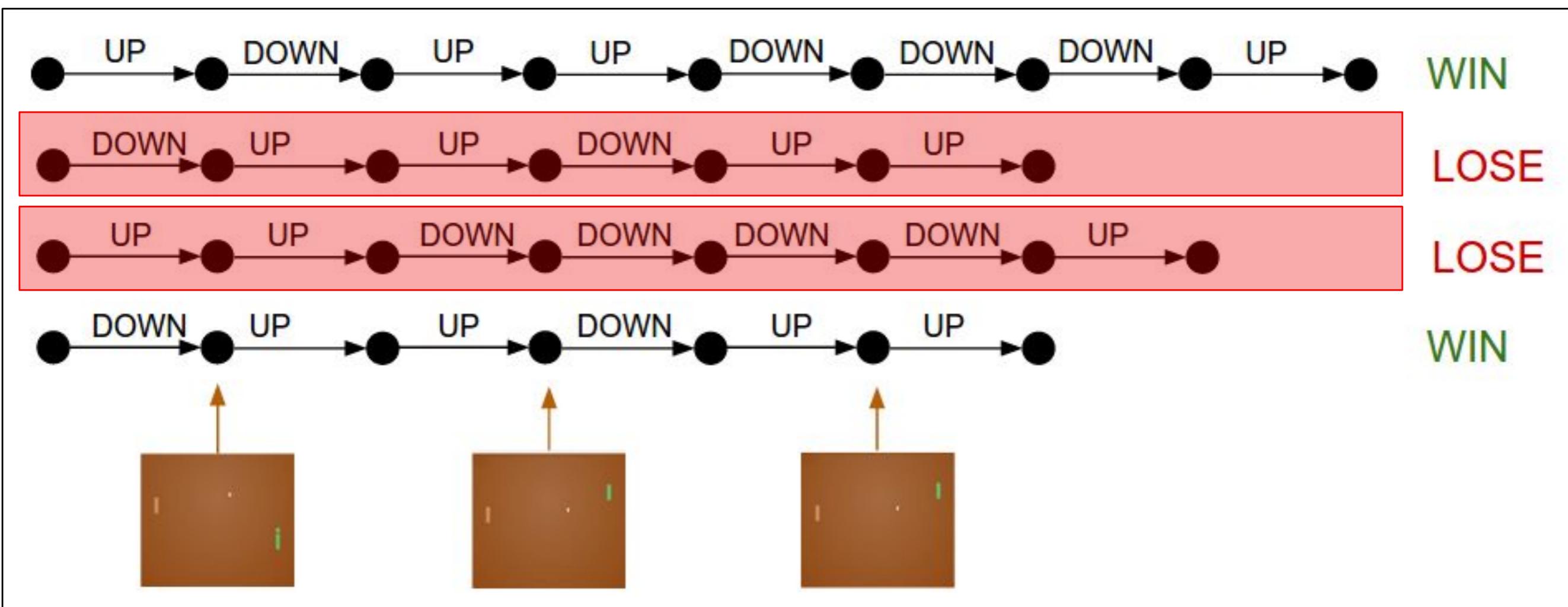
4 rollouts:



Not sure whatever we did here, but apparently it was good.



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

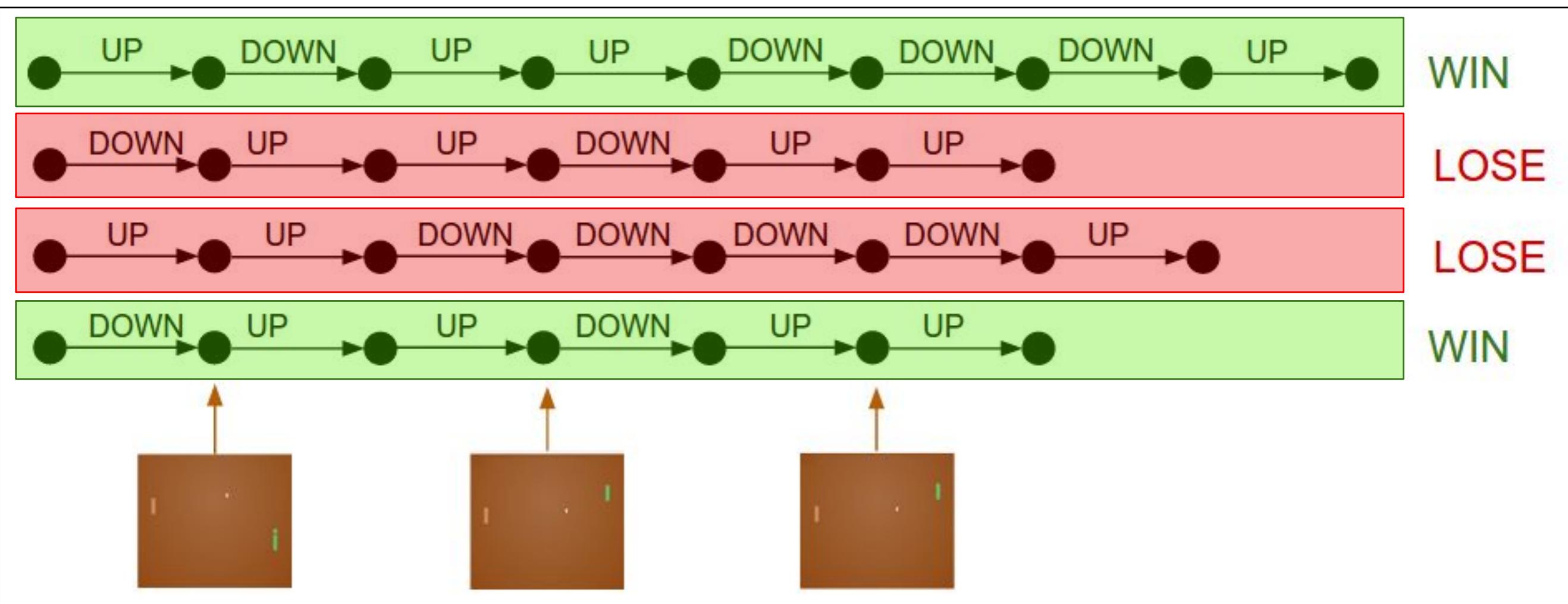


Pretend every action we took here was the correct label.

$$\text{maximize: } \log p(y_i | x_i)$$

Pretend every action we took here was the wrong label.

$$\text{maximize: } (-1) * \log p(y_i | x_i)$$



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

Supervised Learning

maximize:

$$\sum_i \log p(y_i | x_i)$$

For images x_i and their
labels y_i .

Supervised Learning

maximize:

$$\sum_i \log p(y_i | x_i)$$

For images x_i and their labels y_i .

Reinforcement Learning

Supervised Learning

maximize:

$$\sum_i \log p(y_i | x_i)$$

For images x_i and their labels y_i .

Reinforcement Learning

1) we have no labels so we sample:

$$y_i \sim p(\cdot | x_i)$$

Supervised Learning

maximize:

$$\sum_i \log p(y_i | x_i)$$

For images x_i and their labels y_i .

Reinforcement Learning

1) we have no labels so we sample:

$$y_i \sim p(\cdot | x_i)$$

2) once we collect a batch of rollouts:
maximize:

$$\sum_i A_i * \log p(y_i | x_i)$$

Supervised Learning

maximize:

$$\sum_i \log p(y_i | x_i)$$

For images x_i and their labels y_i .

Reinforcement Learning

1) we have no labels so we sample:

$$y_i \sim p(\cdot | x_i)$$

2) once we collect a batch of rollouts:
maximize:

$$\sum_i A_i * \log p(y_i | x_i)$$

We call this the **advantage**, it's a number, like +1.0 or -1.0 based on how this action eventually turned out.

Advantage is the same for all actions taken during a trajectory, and depends on the trajectory return (episode return)

Supervised Learning

maximize:

$$\sum_i \log p(y_i | x_i)$$

For images x_i and their labels y_i .

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

Reinforcement Learning

1) we have no labels so we sample:

$$y_i \sim p(\cdot | x_i)$$

2) once we collect a batch of rollouts:
maximize:

$$\sum_i A_i * \log p(y_i | x_i)$$

+ve advantage will make that action more likely in the future, for that state.

-ve advantage will make that action less likely in the future, for that state.

Variance

Here is our gradient estimator:

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

It is unbiased, i.e., with large N it will accurately approximate the true gradient.

Problem: Unfortunately, usually a very large N is required.

Can we improve the variance of our estimator?

Variance

Here is our gradient estimator:

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(\alpha_t^{(i)} | s_t^{(i)}) G_t^{(i)}$$

Variance is the trace of the covariance matrix:

$$\text{Var}(\hat{g}) = \text{tr} \left(\mathbb{E} [(\hat{g} - \mathbb{E}[\hat{g}])(\hat{g} - \mathbb{E}[\hat{g}])^T] \right) = \sum_{k=1}^n \mathbb{E} \left[(\hat{g}_k - \mathbb{E}[\hat{g}_k])^2 \right]$$

Our goal is to minimize the variance of our estimator.

Reducing variance by subtracting a baseline

What if we **subtract a constant b** from the rewards:

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) (R(\tau^{(i)}) - b)$$
$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)}) - \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) b$$

Reducing variance by subtracting a baseline

What if we **subtract a constant b** from the rewards:

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) (R(\tau^{(i)}) - b)$$

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)}) - \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) b$$

original formulation

new term

Reducing variance by subtracting a baseline

What if we **subtract a constant b** from the rewards:

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) (R(\tau^{(i)}) - b)$$
$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)}) - \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) b$$

original formulation

new term

$$\sum_{\tau} P(\tau; \theta) \nabla_{\theta} \log P_{\theta}(\tau) b = \sum_{\tau} P(\tau; \theta) \frac{\nabla_{\theta} P_{\theta}(\tau)}{P(\tau; \theta)} b$$
$$= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) b$$
$$= b \left(\sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) \right)$$
$$= b \left(\nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) \right)$$
$$= 0$$

Reducing variance by subtracting a baseline

What if we subtract a constant b from the rewards:

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) (R(\tau^{(i)}) - b)$$
$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) R(\tau^{(i)}) - \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \log P_{\theta}(\tau^{(i)}) b$$

original formulation

new term

$$\begin{aligned} \sum_{\tau} P(\tau; \theta) \nabla_{\theta} \log P_{\theta}(\tau) b &= \sum_{\tau} P(\tau; \theta) \frac{\nabla_{\theta} P_{\theta}(\tau)}{P(\tau; \theta)} b \\ &= \sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) b \\ &= b \left(\sum_{\tau} \nabla_{\theta} P_{\theta}(\tau) \right) \\ &= b \left(\nabla_{\theta} \sum_{\tau} P_{\theta}(\tau) \right) \\ &= 0 \end{aligned}$$

- new term equals to 0 in expectation
- We are still unbiased

Baseline choices

Constant Baseline (single scalar): $b = \mathbb{E}[R(\tau)]$

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) (G_t^{(i)} - \textcolor{blue}{b})$$

Time-dependent Baseline (a vector length T): $b_t = \sum_{i=1}^N G_t^{(i)}$

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) (G_t^{(i)} - \textcolor{blue}{b}_t)$$

State-dependent Baseline (a function):

$$b(s) = \mathbb{E} [r_t + r_{t+1} + r_{t+2} + \dots + r_{T-1} \mid s_t = s] = \textcolor{red}{V}_{\pi}(s)$$

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) (G_t^{(i)} - \textcolor{blue}{b}(s_t^{(i)}))$$

Variance

Subtracting a state-dependent Baseline:

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) G_t^{(i)}$$

$$\text{Var}(\hat{g}) = \text{tr} \left(\mathbb{E} [(\hat{g} - \mathbb{E}[\hat{g}])(\hat{g} - \mathbb{E}[\hat{g}])^T] \right) = \sum_{k=1}^n \mathbb{E} \left[(\hat{g}_k - \mathbb{E}[\hat{g}_k])^2 \right]$$

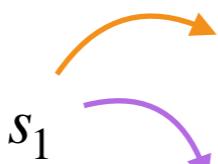
- Imagine in some state S_1 the rewards of all actions are ~ 3000 and in some state S_2 the rewards of all actions are ~ -4000 .

Action advantages Versus Action returns

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \underbrace{(G_t^{(i)} - b(s_t^{(i)}))}_{\text{advantage}}$$

$$G(a_1) = 3030$$

s_1

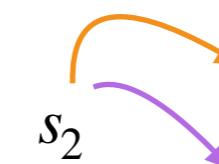


$$v(s_1) = 3000$$

$$G(a_2) = 2970$$

$$G(a_1) = -3970$$

s_2



$$v(s_2) = -4000$$

$$G(a_2) = -4030$$

- Imagine in some state s_1 the rewards of all actions are ~ 3000 and in some state s_2 the rewards of all actions are ~ -4000 . Bad actions in s_1 will much influence the gradient way more than good actions in s_2 . But our goal is to select the right actions across both bad and good states!
- Now imagine you have $b(s_1) = 3000$ and $b(s_2) = -4000$ and you subtract those values from the sample returns.

Action advantages Versus Action returns

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \underbrace{(G_t^{(i)} - b(s_t^{(i)}))}_{\text{advantage}}$$



- Imagine in some state s_1 the rewards of all actions are ~ 3000 and in some state s_2 the rewards of all actions are ~ -4000 . Bad actions in s_1 will much influence the gradient way more than good actions in s_2 . But our goal is to select the right actions across both bad and good states!
- Now imagine you have $b(s_1) = 3000$ and $b(s_2) = -4000$ and you subtract those values from the sample returns.
- We want to encourage an action not when it has high return, but when it has **higher return than the other actions from that state**, i.e., when it has an **advantage** over other actions. It may well be that a state is bad and all actions have low returns in that state, we care to find the actions that have higher returns than the rest, and we need to calibrate for the goodness of state using state dependent baselines.

Estimate state dependent baseline $V_\phi^\pi(s_t)$

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t^{(i)} | s_t^{(i)}) \left(G_t^{(i)} - V_\phi^\pi(s_t^{(i)}) \right)$$

- Monte Carlo estimation
1. Initialize ϕ
 2. Collect trajectories: τ_1, \dots, τ_N
 3. Regress against empirical return, e.g., using gradient descent:

$$\phi \leftarrow \arg \min_{\phi} \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^{T-1} \left(V_\phi^\pi(s_t^{(i)}) - \left(\sum_{k=t}^{T-1} R(s_k^{(i)}, a_k^{(i)}) \right) \right)^2$$

Estimate state dependent baseline $V_\phi^\pi(s_t)$

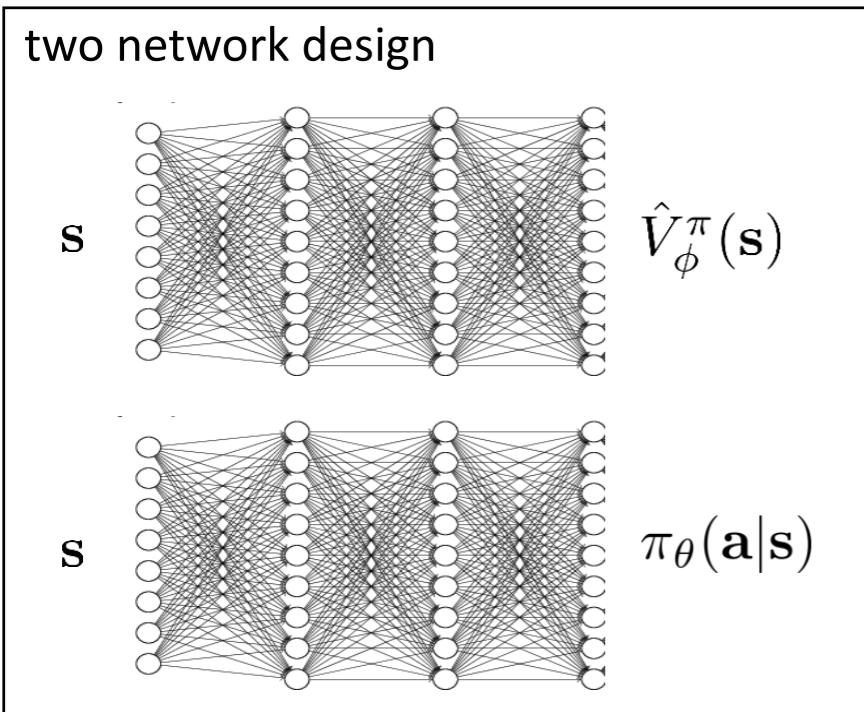
$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t^{(i)} | s_t^{(i)}) \left(G_t^{(i)} - V_\phi^\pi(s_t^{(i)}) \right)$$

- TD estimation
1. Initialize ϕ
 2. Collect data for: s, a, s', r
 3. Fitted V iteration

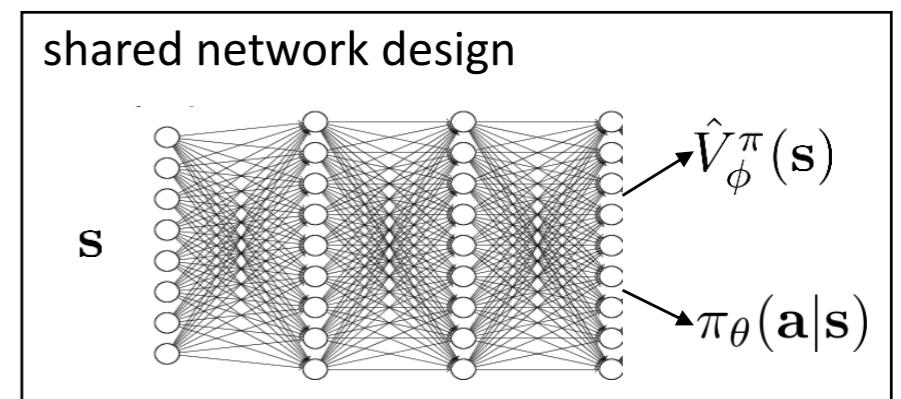
$$\phi_{\ell+1} \leftarrow \arg \min_{\phi} \sum_{(s,a,s',r)} \left\| (r + V_{\phi_\ell}^\pi(s')) - V_\phi(s) \right\|_2^2$$

Actor-Critic

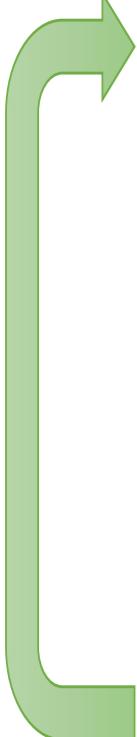
- ▶ Actor-critic algorithms maintain two sets of parameters
 - Critic Updates action-value function parameters w
 - Actor Updates policy parameters θ , in direction suggested by critic



+ simple & stable
- no shared features between actor & critic



Actor-Critic

- 
0. Initialize policy parameters θ and critic parameters ϕ .
 1. Sample trajectories $\{\tau_i = \{s_t^i, a_t^i\}_{i=0}^T\}$ by deploying the current policy $\pi_\theta(a_t | s_t)$.
 2. Fit value function $V_\phi^\pi(s)$ by MC or TD estimation (update ϕ)
 3. Compute action advantages $A^\pi(s_t^i, a_t^i) = G_t^{(i)} - V_\phi^\pi(s_t^i)$
 4. $\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t^i | s_t^i) A^\pi(s_t^i, a_t^i)$
 5. $\theta \leftarrow \theta + \alpha \nabla_\theta U(\theta)$

Sample returns are Q estimates!

Whether we use a baseline:

$$\hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \left(G_t^{(i)} - V_{\phi}^{\pi}(s_t^{(i)}) \right)$$

or not:

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

returns $G_t^{(i)} = \sum_{k=t}^T R(s_k^{(i)}, a_k^{(i)})$ are trying to estimate Q values from a single rollout:

$$Q^{\pi}(s, a) = \mathbb{E}[R_0 + R_1 + \dots | S_0 = s, A_0 = a]$$

Sample returns are Q estimates!

Returns $G_t^{(i)} = \sum_{k=t}^T R(s_k^{(i)}, a_k^{(i)})$ are trying to estimate Q values from a single rollout:

$$Q^\pi(s, a) = \mathbb{E}[R_0 + R_1 + \dots | S_0 = s, A_0 = a]$$

Minimize variance of such Q estimates by:

- discounting
- introducing a **learnt approximation for the Q**, as opposed to returns from single rollouts

By definition of the Q function

$$Q^{\pi, \gamma}(s, a) = \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 R_2 \dots | S_0 = s, A_0 = a]$$

By definition of the Q function

$$\begin{aligned}Q^{\pi, \gamma}(s, a) &= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 R_2 \dots | S_0 = s, A_0 = a] \\&= \mathbb{E}[R_0 + \gamma V^\pi(S_1) | S_0 = s, A_0 = a]\end{aligned}$$

By definition of the Q function

$$\begin{aligned}Q^{\pi, \gamma}(s, a) &= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 R_2 \dots | S_0 = s, A_0 = a] \\&= \mathbb{E}[R_0 + \gamma V^\pi(S_1) | S_0 = s, A_0 = a] \\&= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 V^\pi(S_2) | S_0 = s, A_0 = a]\end{aligned}$$

By definition of the Q function

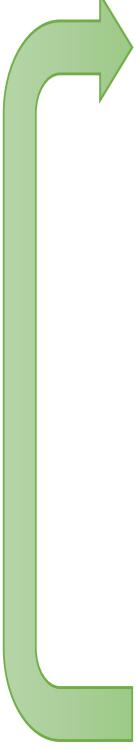
$$\begin{aligned}Q^{\pi, \gamma}(s, a) &= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 R_2 \dots | S_0 = s, A_0 = a] \\&= \mathbb{E}[R_0 + \gamma V^\pi(S_1) | S_0 = s, A_0 = a] \\&= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 V^\pi(S_2) | S_0 = s, A_0 = a] \\&= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 R_2 + \gamma^3 V^\pi(S_3) | S_0 = s, A_0 = a] \\&= \dots\end{aligned}$$

If I have estimated $V_\phi^\pi(S)$, I can use it to estimate the Q values, I do not need a separate Q function approximate!

Advantage Actor-Critic

- 
0. Initialize policy parameters θ and critic parameters ϕ .
 1. Sample trajectories $\{\tau_i = \{s_t^i, a_t^i\}_{t=0}^T\}$ by deploying the current policy $\pi_\theta(a_t | s_t)$.
 2. Fit value function $V_\phi^\pi(s)$ by MC or TD estimation (update ϕ)
 3. Compute action advantages $A^\pi(s_t^i, a_t^i) = R(s_t^i, a_t^i) + \gamma V_\phi^\pi(s_{t+1}^i) - V_\phi^\pi(s_t^i)$
 4. $\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t^i | s_t^i) A^\pi(s_t^i, a_t^i)$
 5. $\theta \leftarrow \theta + \alpha \nabla_\theta U(\theta)$

Advantage Actor-Critic

- 
0. Initialize policy parameters θ and critic parameters ϕ .
 1. Sample trajectories $\{\tau_i = \{s_t^i, a_t^i\}_{t=0}^T\}$ by deploying the current policy $\pi_\theta(a_t | s_t)$.
 2. Fit value function $V_\phi^\pi(s)$ by MC or TD estimation (update ϕ)
 3. Compute action advantages $A^\pi(s_t^i, a_t^i) = R(s_t^i, a_t^i) + \gamma V_\phi^\pi(s_{t+1}^i) - V_\phi^\pi(s_t^i)$
 4. $\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t^i | s_t^i) A^\pi(s_t^i, a_t^i)$
 5. $\theta \leftarrow \theta + \alpha \nabla_\theta U(\theta)$

How can we compute this in automatic differentiation packages, such as Tensorflow?

We need to write the expression that when differentiated will give that gradient.

$$\hat{U} = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \log \pi_\theta(a_t^i | s_t^i) A^\pi(s_t^i, a_t^i)$$

REINFORCE/Actor-critic training

- Stability of training neural networks requires the **gradient updates to be de-correlated**
- This is not the case if data arrives **sequentially**
- Gradient updates computed from some part of the space can cause the value (Q) function approximator to **oscillate**
- Our solution so far has been: **Experience buffers** where experience tuples are mixed and sampled from. Resulting sampled batches are more stationary than the ones encountered online (without buffer)
- This limits deep RL to **off-policy** methods, since data from an older policy are used to update the weights of the value approximator (critic) (except if we take care and weight such data under our current stochastic policy->importance sampling)

Asynchronous Deep RL for on policy learning

Asynchronous Methods for Deep Reinforcement Learning

Volodymyr Mnih¹

Adrià Puigdomènech Badia¹

Mehdi Mirza^{1,2}

Alex Graves¹

Tim Harley¹

Timothy P. Lillicrap¹

David Silver¹

Koray Kavukcuoglu¹

¹ Google DeepMind

² Montreal Institute for Learning Algorithms (MILA), University of Montreal

VMNIH@GOOGLE.COM

ADRIAP@GOOGLE.COM

MIRZAMOM@IRO.UMONTREAL.CA

GRAVESEA@GOOGLE.COM

THARLEY@GOOGLE.COM

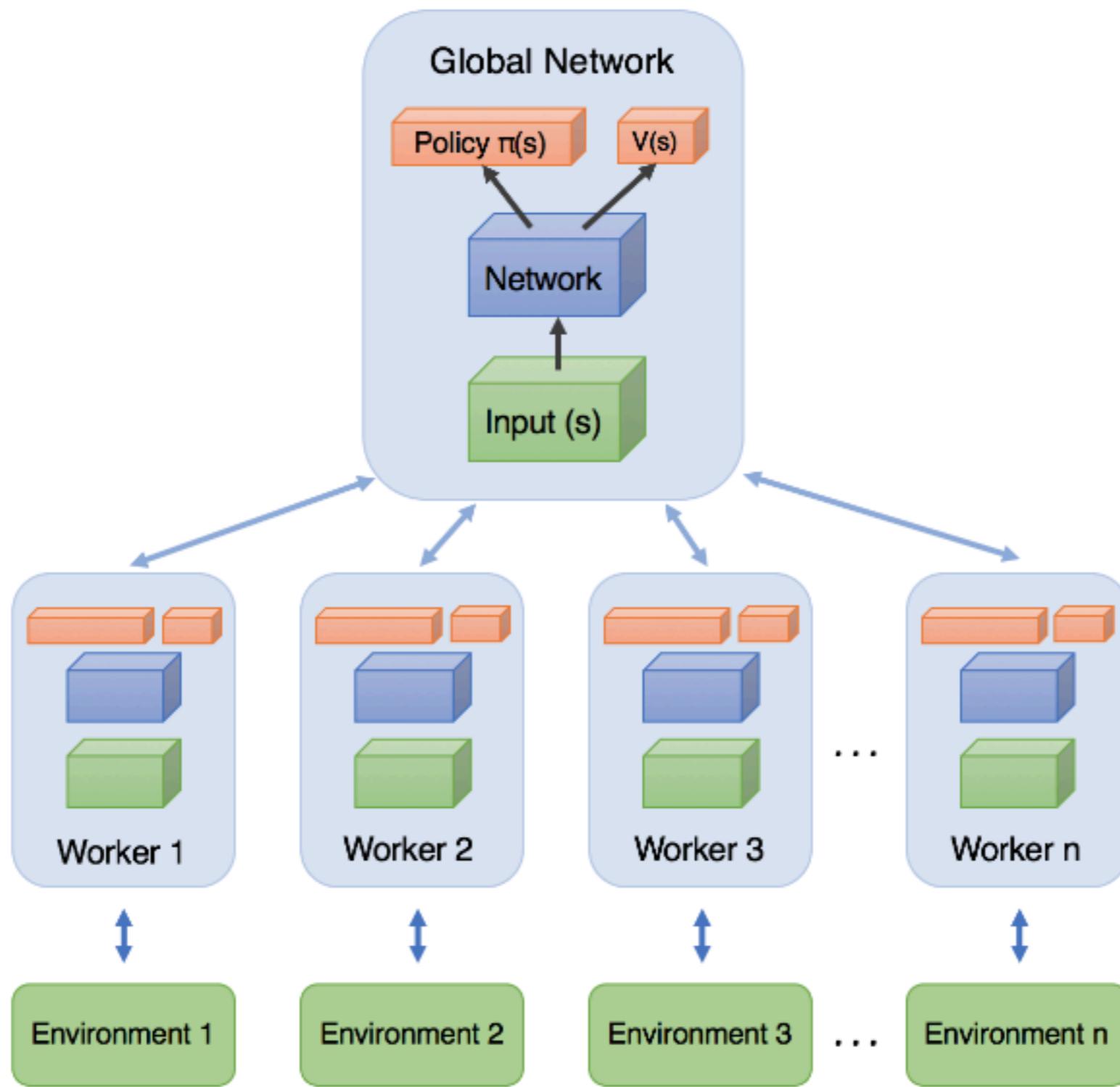
COUNTZERO@GOOGLE.COM

DAVIDSILVER@GOOGLE.COM

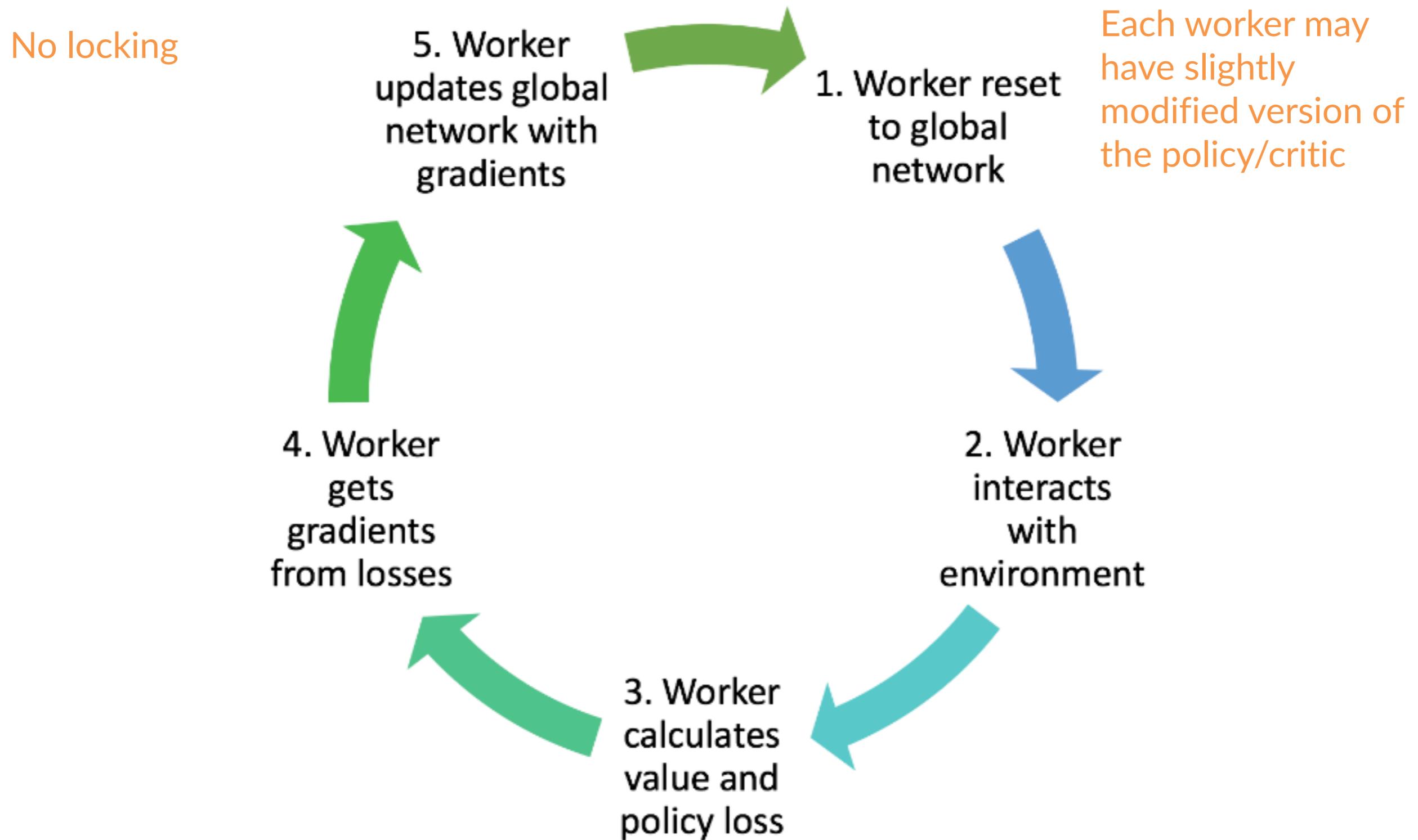
KORAYK@GOOGLE.COM

- Alternative: parallelize the collection of experience and stabilize training **without experience buffers**
- **Multiple threads of experience**, one per agent, each exploring in different part of the environment contributing experience tuples
- **Different exploration strategies** (e.g., various ϵ values) in different threads increase diversity
- Now you can train on-policy using any of our policy gradient methods

Distributed RL



Distributed Asynchronous RL-A3C



Algorithm S3 Asynchronous advantage actor-critic - pseudocode for each actor-learner thread.

// Assume global shared parameter vectors θ and θ_v , and global shared counter $T = 0$

// Assume thread-specific parameter vectors θ' and θ'_v

Initialize thread step counter $t \leftarrow 1$

repeat

 Reset gradients: $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.

 Synchronize thread-specific parameters $\theta' = \theta$ and $\theta'_v = \theta_v$ **Copying the weights**

$t_{start} = t$

 Get state s_t

repeat

 Perform a_t according to policy $\pi(a_t|s_t; \theta')$

Rollout

 Receive reward r_t and new state s_{t+1}

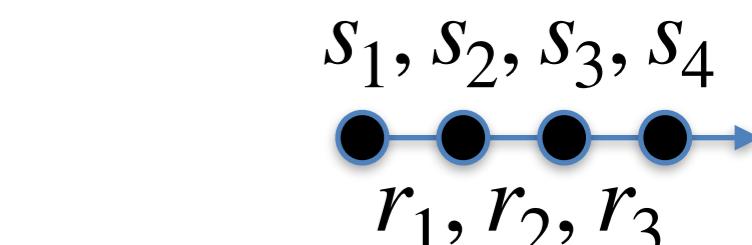
$t \leftarrow t + 1$

$T \leftarrow T + 1$

until terminal s_t **or** $t - t_{start} == t_{max}$

$R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta'_v) & \text{for non-terminal } s_t // \text{Bootstrap from last state} \end{cases}$

for $i \in \{t - 1, \dots, t_{start}\}$ **do**



Advantage

$R \leftarrow r_i + \gamma R$

 Accumulate gradients wrt θ' : $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i; \theta') (R - V(s_i; \theta'_v))$

 Accumulate gradients wrt θ'_v : $d\theta_v \leftarrow d\theta_v + \partial (R - V(s_i; \theta'_v))^2 / \partial \theta'_v$

end for

 Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.

Learning the critic

until $T > T_{max}$

What is the approximation used for the advantage?

$$R_3 = r_3 + \gamma V(s_4, \theta'_v)$$

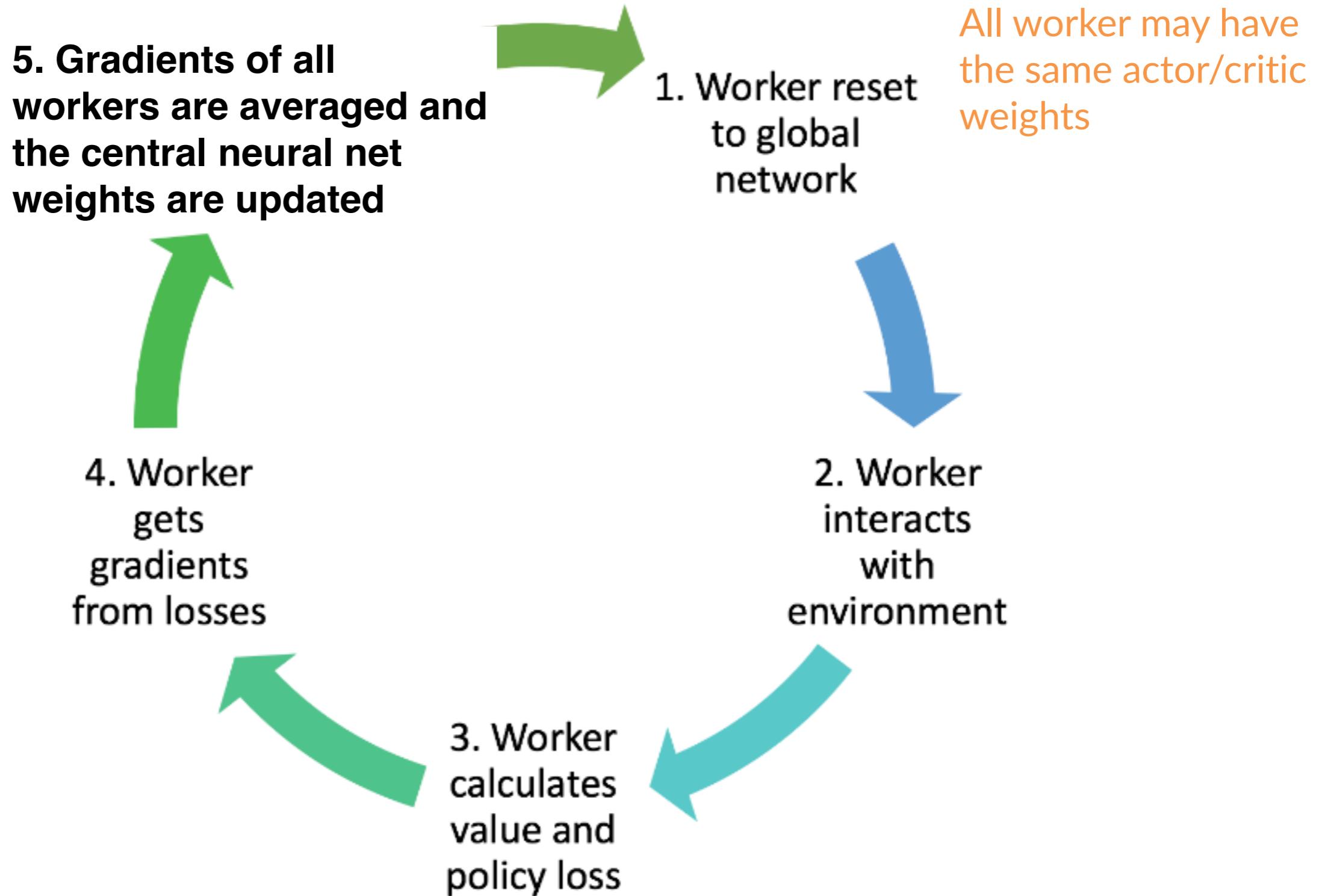
$$A_3 = R_3 - V(s_3; \theta'_v)$$

$$R_2 = r_2 + \gamma r_3 + \gamma^2 V(s_4, \theta'_v)$$

$$A_2 = R_2 - V(s_2; \theta'_v)$$

$R_3 - V(s_3)$

Distributed Synchronous RL-A2C



Algorithm S3 Asynchronous advantage actor-critic - pseudocode for each actor-learner thread.

// Assume global shared parameter vectors θ and θ_v , and global shared counter $T = 0$

// Assume thread-specific parameter vectors θ' and θ'_v

Initialize thread step counter $t \leftarrow 1$

repeat

 Reset gradients: $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.

 Sampling thread specific parameters $\theta' \leftarrow \theta$ and $\theta'_v \leftarrow \theta_v$

 ``We also found that adding the *entropy* of the policy π to the objective function improved exploration by discouraging premature convergence to suboptimal deterministic policies.” So you need to add to the policy gradient: $+ \beta \nabla_{\theta} H(\pi_{\theta}(a_t | s_t; \theta))$

We will look into the entropy as part of the reward in later lecture

 Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.
until $T > T_{max}$

What is the approximation used for the advantage?

$$R_3 = r_3 + \gamma V(s_4, \theta'_v)$$

$$R_2 = r_2 + \gamma r_3 + \gamma^2 V(s_4, \theta'_v)$$

$$A_3 = R_3 - V(s_3; \theta'_v)$$

$$A_2 = R_2 - V(s_2; \theta'_v)$$