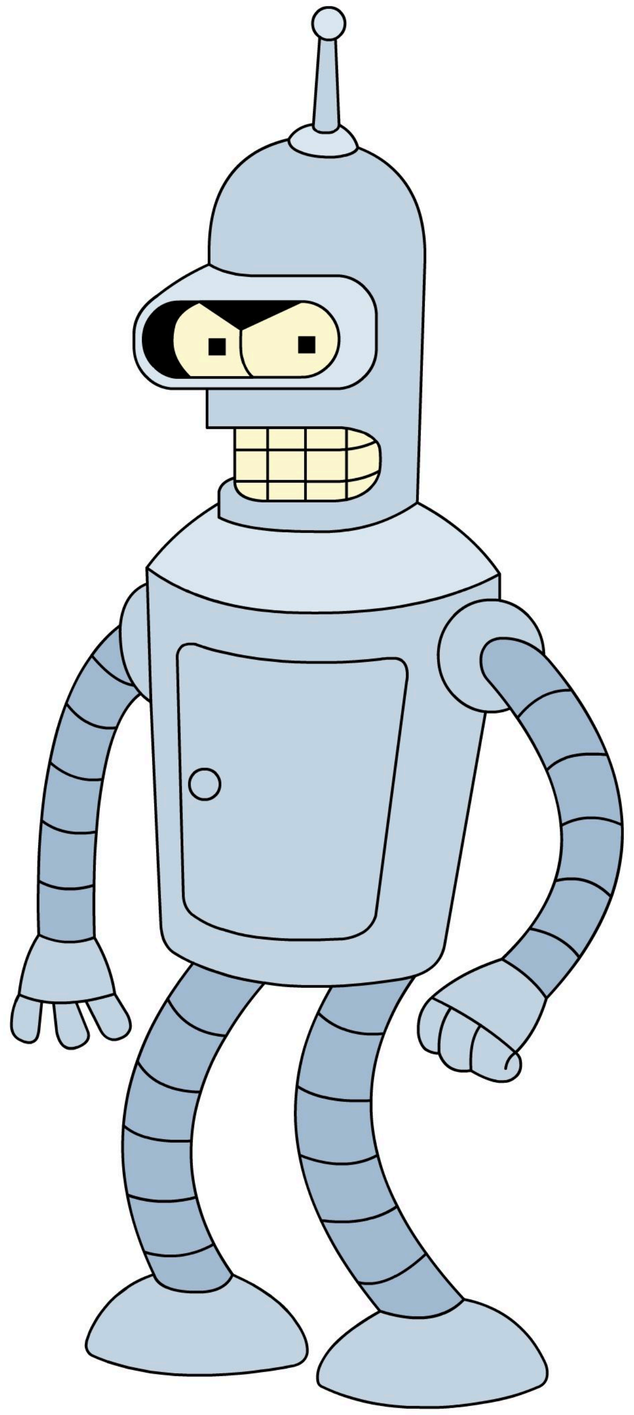
Reinforcement Learning

HSE, winter - spring 2024

Lecture 4: Policy Gradient



Sergei Laktionov
slaktionov@hse.ru
LinkedIn

Recap: Value-based Methods

1. Approximate action-value function with a neural network: $Q(s, a) \approx Q_\theta(s, a)$

2. Take action which maximises $Q_\theta(s, a)$

Recap: Value-based Methods

1. Approximate action-value function with a neural network: $Q(s, a) \approx Q_\theta(s, a)$

2. Take action which maximises $Q_\theta(s, a)$

+ Easy to generate policy

+ Close to true objective

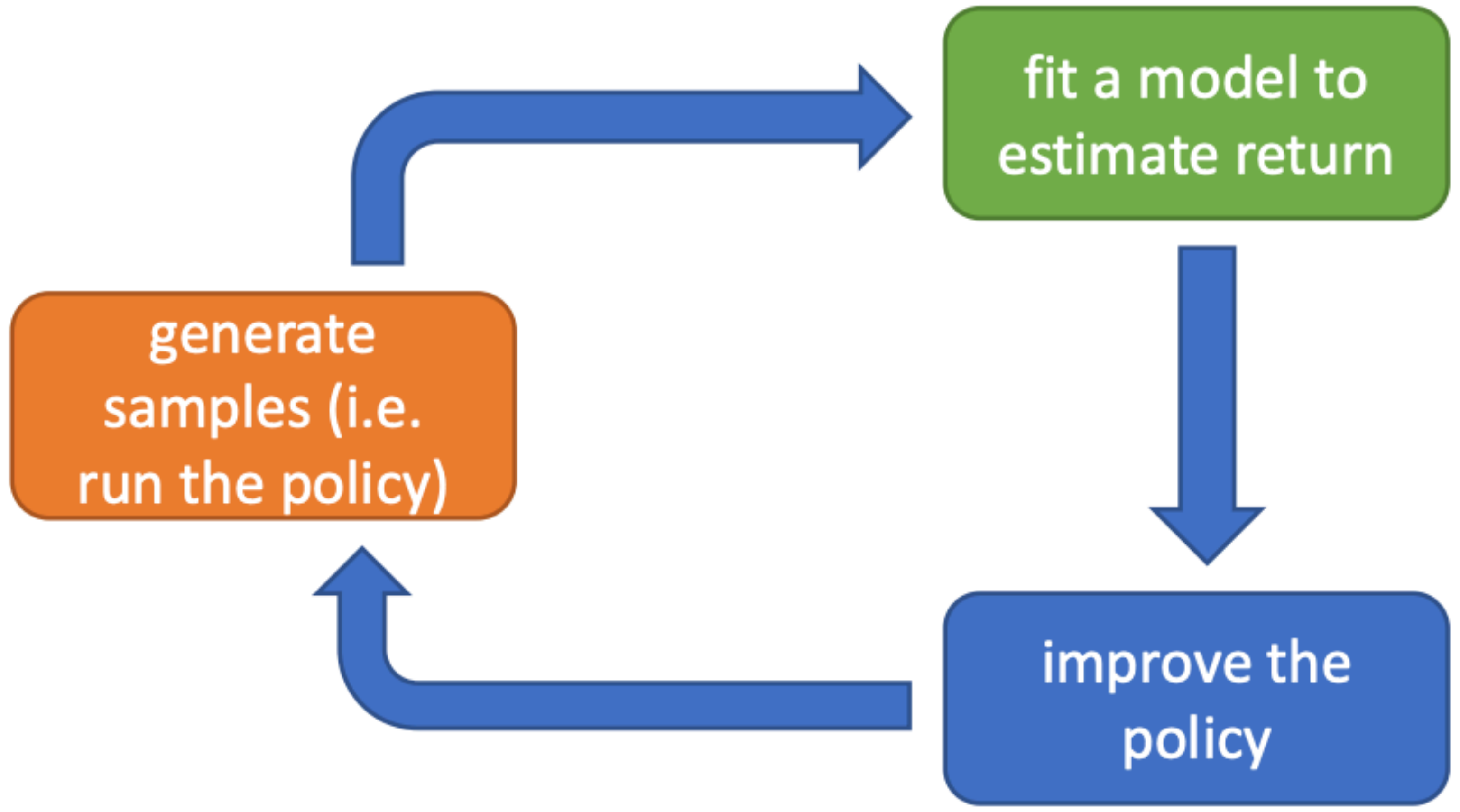
+ Fairly well-understood, good algorithms exist

- Still not the true objective

- May focus capacity on irrelevant details

- Small value error can lead to larger policy error

$Q_\phi(\mathbf{s}, \mathbf{a}) \leftarrow r(\mathbf{s}, \mathbf{a}) + \gamma \max_{\mathbf{a}'} Q_\phi(\mathbf{s}', \mathbf{a}')$



$\mathbf{a} = \arg\max_{\mathbf{a}} Q_\phi(\mathbf{s}, \mathbf{a})$

Recap: Value-based Methods

1. Estimated the Q-function
2. Choose an action: left or right

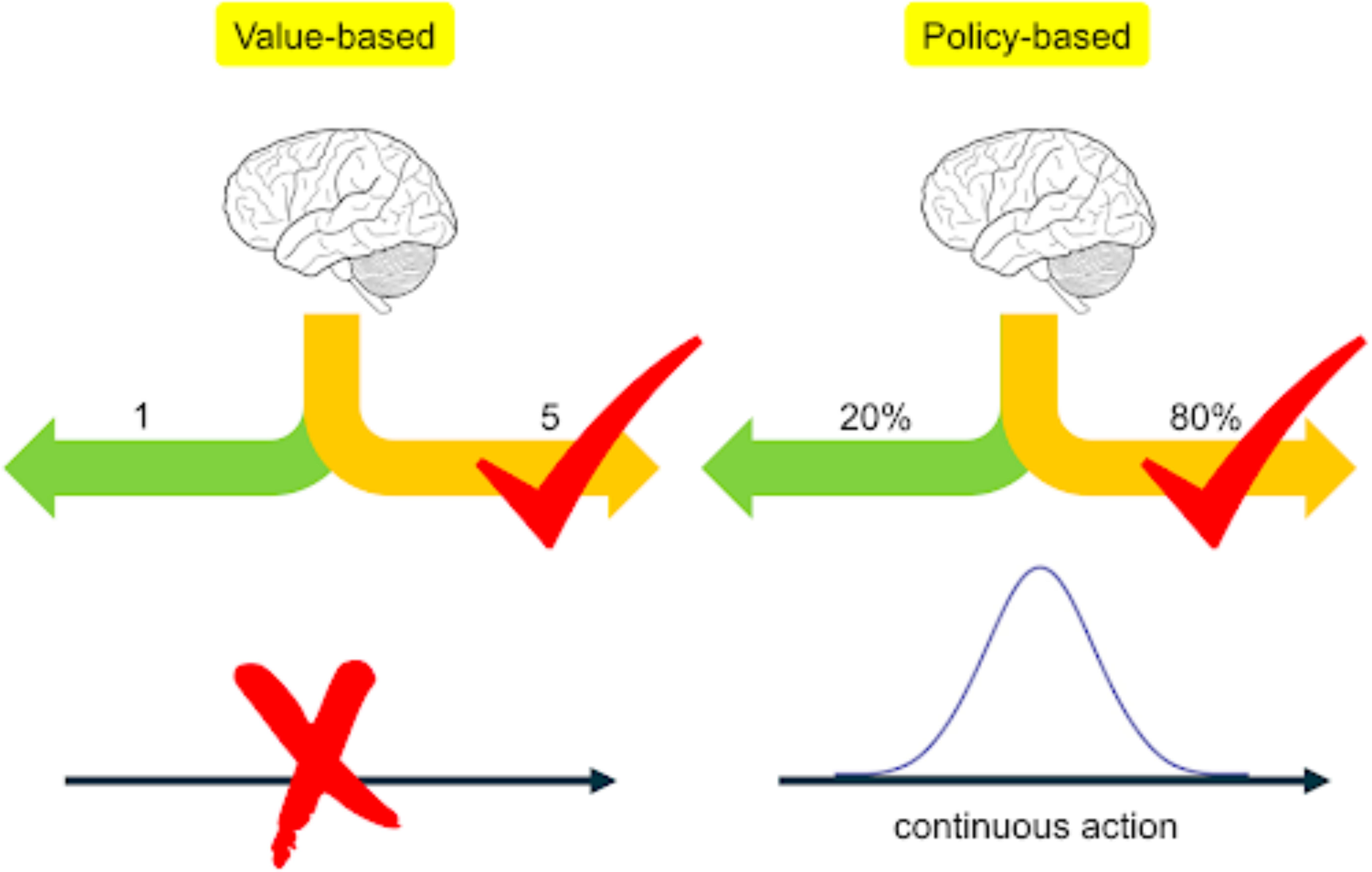
Source

Recap: Value-based Methods

"When solving a problem of interest, do not solve a more general problem as an intermediate step. Try to get the answer that you really need but not a more general one."

—Vladimir Vapnik

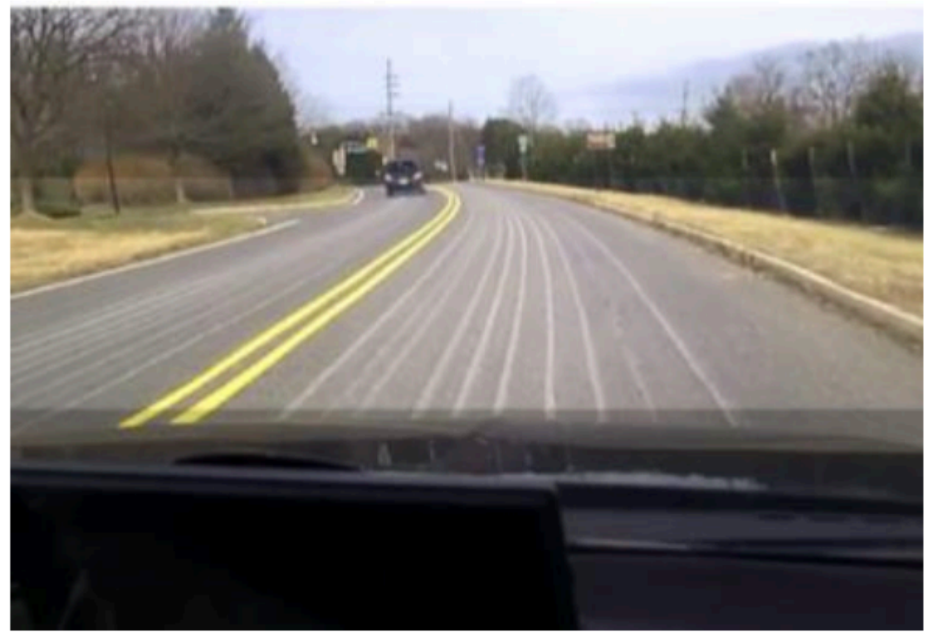
Value-based vs Policy-based



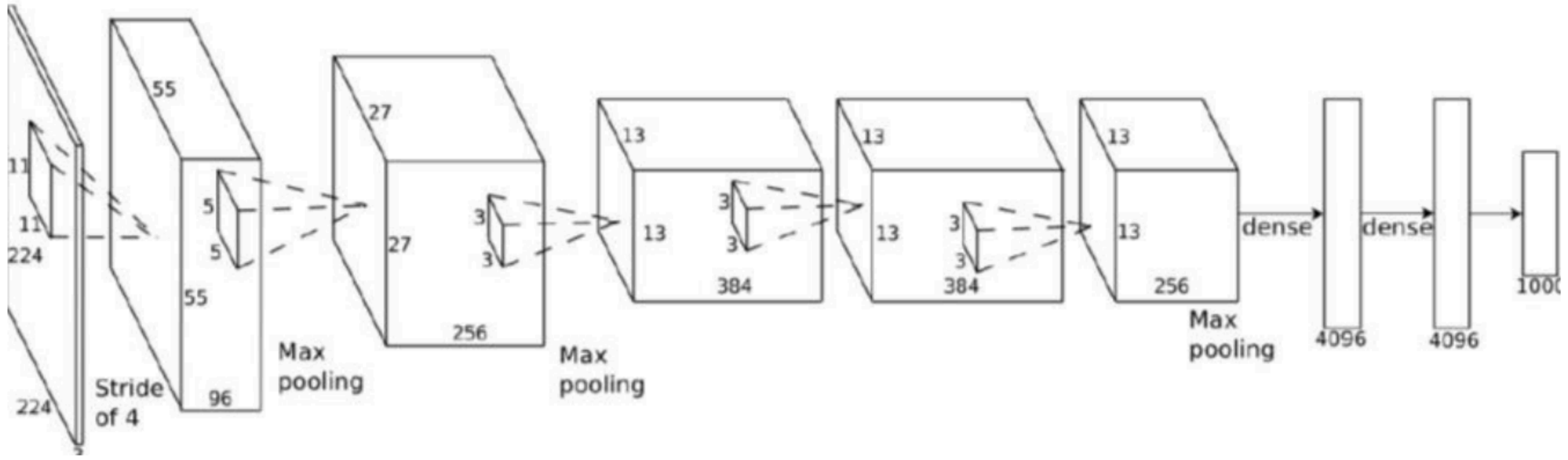
Parametric Policy

Suppose that since now we are living in the class of parametric policies:

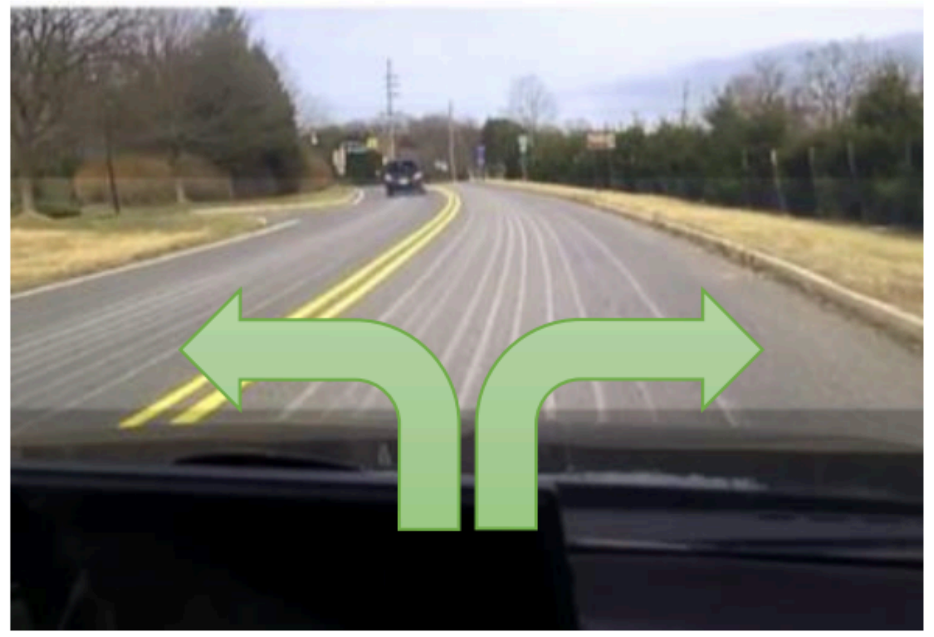
$\pi_\theta(a \mid s) = \mathbb{P}(a_t = a \mid s_t = s; \theta)$, where θ is some parameter.



$\mathbf{s}_t$



$\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$



$\mathbf{a}_t$

Objective

Suppose that since now we are living in the class of parametric policies:

$\pi_\theta(a \mid s) = \mathbb{P}(a_t = a \mid s_t = s; \theta)$, where θ is some parameter.

$$\theta^* = argmax_\theta \, J(\theta) = argmax_\theta \, \mathbb{E}_{p_\theta(\tau)}[\sum_{t=0}^{T} \gamma^t R_t] = argmax_\theta \, \mathbb{E}_{p_\theta(\tau)}[G(\tau)]$$

$$J(\theta) = \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \int p_\theta(\tau) G(\tau) d\tau$$

Objective

Suppose that since now we are living in the class of parametric policies:

$\pi_\theta(a \mid s) = \mathbb{P}(a_t = a \mid s_t = s; \theta)$, where θ is some parameter.

$$\theta^* = argmax_\theta\, J(\theta) = argmax_\theta\, \mathbb{E}_{p_\theta(\tau)}[\sum_{t=0}^{T} \gamma^t R_t] = argmax_\theta\, \mathbb{E}_{p_\theta(\tau)}[G(\tau)]$$

$$J(\theta) = \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \int p_\theta(\tau) G(\tau) d\tau$$

$$p_\theta(\tau) = p(s_0)\pi_\theta(a_0 \mid s_0)p(s_1 \mid s_0, a_0)\ldots$$

Objective's Gradient

$$p_\theta(\tau) = p(s_0)\pi_\theta(a_0 \mid s_0)p(s_1 \mid s_0, a_0)\ldots$$

$$\nabla J(\theta) = \nabla \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \int \nabla p_\theta(\tau) G(\tau) d\tau$$

Objective's Gradient

$$p_\theta(\tau) = p(s_0)\pi_\theta(a_0 \mid s_0)p(s_1 \mid s_0, a_0)\ldots$$

$$\nabla J(\theta) = \nabla \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \int \nabla p_\theta(\tau) G(\tau) d\tau$$

Log-derivative trick: $\nabla p_\theta(\tau) = p_\theta(\tau) \dfrac{\nabla p_\theta(\tau)}{p_\theta(\tau)} = p_\theta(\tau) \nabla \log p_\theta(\tau)$

Objective's Gradient

$$p_\theta(\tau) = p(s_0)\pi_\theta(a_0 \,|\, s_0)p(s_1 \,|\, s_0, a_0)\ldots$$

$$\nabla J(\theta) = \nabla \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \int \nabla p_\theta(\tau) G(\tau) d\tau$$

Log-derivative trick: $\nabla p_\theta(\tau) = p_\theta(\tau)\dfrac{\nabla p_\theta(\tau)}{p_\theta(\tau)} = p_\theta(\tau) \nabla \log p_\theta(\tau) =$

$$= p_\theta(\tau) \sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \,|\, s_t)$$

Objective's Gradient

$$\nabla J(\theta) = \nabla \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \mathbb{E}_{p_\theta(\tau)}[\nabla \log p_\theta(\tau) G(\tau)]$$

REINFORCE (1992)

$$\nabla J(\theta) = \nabla \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \mathbb{E}_{p_\theta(\tau)}[\nabla \log p_\theta(\tau) G(\tau)]$$

1. Sample N trajectories from the environment using current policy π_{θ_k}

2. Estimate gradient using Monte-Carlo estimator:

$$\nabla J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_{i,t} \mid s_{i,t}) G(\tau_i) \Big]$$

3. Make gradient ascent step:

$$\theta_{k+1} = \theta_k + \alpha \nabla J(\theta_k)$$

REINFORCE (1992)

$$\nabla J(\theta) = \nabla \mathbb{E}_{p_\theta(\tau)}[G(\tau)] = \mathbb{E}_{p_\theta(\tau)}[\nabla \log p_\theta(\tau) G(\tau)]$$

1. Sample N trajectories from the environment using current policy π_{θ_k}

2. Estimate gradient using Monte-Carlo estimator:

$$\nabla J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_{i,t} \mid s_{i,t}) G(\tau_i) \Big]$$

3. Make gradient ascent step:

$$\theta_{k+1} = \theta_k + \alpha \nabla J(\theta_k)$$

No Replay Buffer

On-policy algorithm

Old samples can not be used for gradient update

Connection with Supervised Learning

$$\nabla J_{BC}(\theta) = \mathbb{E}_{\tau \sim D}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t)\Big],$$

Maximise log-likelihood to take the similar actions as an expert.

where D is a buffer contains samples collected by an expert

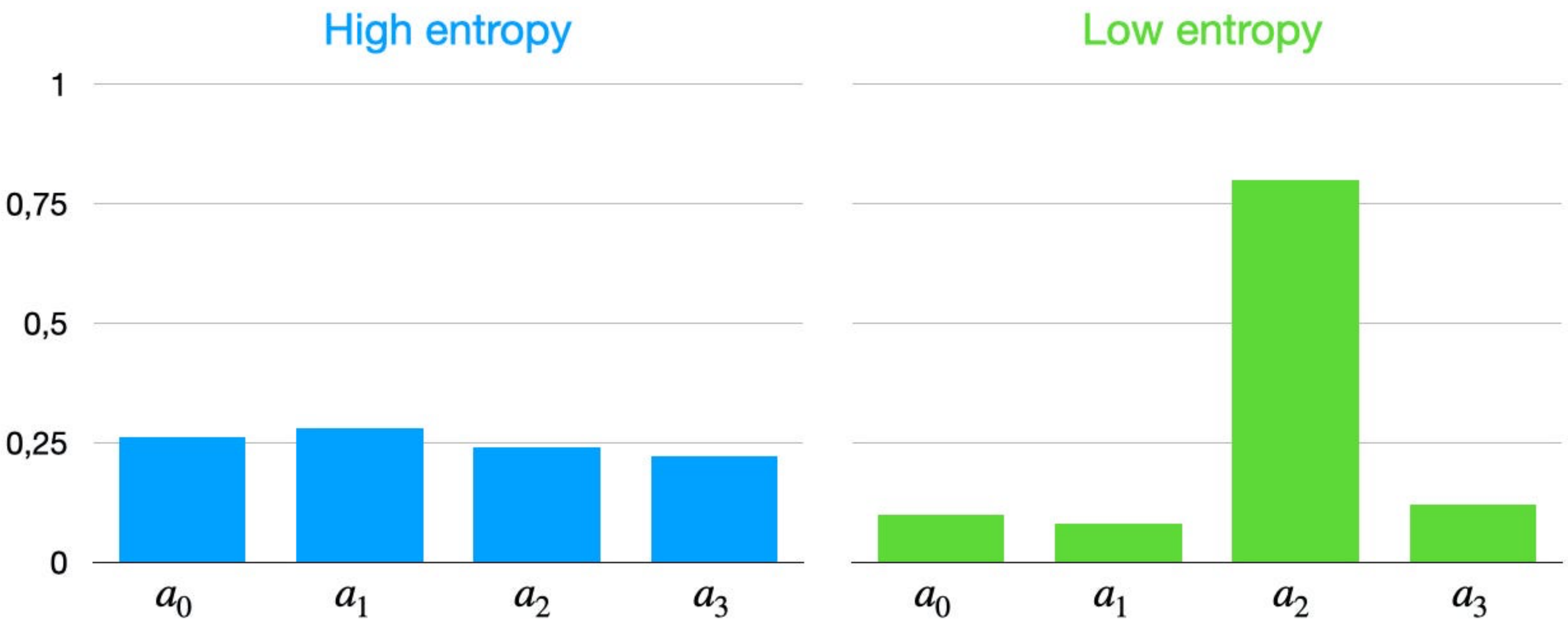
VS

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) G(\tau)\Big]$$

Learn actions which lead to higher returns

Entropy Regularisation

We would still like to sustain the exploration-exploitation trade-off.



Entropy Regularisation

We would still like to sustain the exploration-exploitation trade-off.

$$H(\pi_\theta(\,.\,|\,s)) = -\,\mathbb{E}_{\pi_\theta} \log \pi_\theta(\,.\,|\,s)$$ General case

$$H(\pi_\theta(\,.\,|\,s)) = -\sum_a \pi_\theta(a\,|\,s) \log \pi_\theta(a\,|\,s)$$ Discrete case

Entropy Regularisation

We would still like to sustain the exploration-exploitation trade-off.

$$H(\pi_\theta(\,.\,|\,s)) = -\,\mathbb{E}_{\pi_\theta} \log \pi_\theta(\,.\,|\,s)$$ General case

$$H(\pi_\theta(\,.\,|\,s)) = -\sum_a \pi_\theta(a\,|\,s) \log \pi_\theta(a\,|\,s)$$ Discrete case

Recall that uniform distribution has largest entropy while deterministic distribution has the lowest one.

We can add regularisation term $-\rho H(\pi_\theta(\,.\,|\,s))$ to our objective:

- To encourage an agent to increase curiosity

Policy-based RL

+ Optimise almost the true objective

+ Easy extended to high-dimensional or even continuous action spaced

+ Learn stochastic policies

+ No prior knowledge regarding the MDP dynamics

+ Easy to learn policy directly which seems more natural.

— Could get stuck in local optima

— Less sample efficient in comparison with value-based methods

— High variance ← Let's decrease it

Variance Reduction

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) G(\tau)\Big] = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \boxed{\sum_{k=0}^{T} \gamma^k R_k}\Big]$$

Variance Reduction

Current action $a_{i,t}$ influences only future rewards

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) G(\tau)\Big] = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \boxed{\sum_{k=0}^{T} \gamma^k R_k}\Big]$$

Variance Reduction

Current action $a_{i,t}$ influences only future rewards

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) G(\tau)\Big] = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \boxed{\sum_{k=0}^{T} \gamma^k R_k}\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \sum_{k=t}^{T} \gamma^k R_k\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \gamma^t \sum_{k=t}^{T} \gamma^{k-t} R_k\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \gamma^t G_t\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \gamma^t Q^{\pi_\theta}(s_t, a_t)\Big]$$

Variance Reduction

Current action $a_{i,t}$ influences only future rewards

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) G(\tau)\Big] = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \boxed{\sum_{k=0}^{T} \gamma^k R_k}\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \sum_{k=t}^{T} \gamma^k R_k\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \gamma^t \sum_{k=t}^{T} \gamma^{k-t} R_k\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \gamma^t G_t\Big]$$

$$= \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) \gamma^t Q^{\pi_\theta}(s_t, a_t)\Big] \longrightarrow \text{Let's ignore } \gamma^t$$

Variance Reduction: Baseline

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \,|\, s_t) Q_{\pi_\theta}(s_t, a_t)\Big]$$

Consider some baseline $b(s_t)$ and compute $\mathbb{E}_{p_\theta(\tau)}\big[b(s_t)\, \nabla \log \pi(a_t \,|\, s_t)\big]$:

Variance Reduction: Baseline

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) Q_{\pi_\theta}(s_t, a_t)\Big]$$

Consider some baseline $b(s_t)$ and compute $\mathbb{E}_{p_\theta(\tau)}\left[b(s_t) \nabla \log \pi(a_t \mid s_t)\right]$:

$$\mathbb{E}_{p_\theta(\tau)}\left[b(s_t) \nabla \log \pi(a_t \mid s_t)\right] = \int p_\theta(\tau) b(s_t) \nabla \log p_\theta(\tau) d\tau =$$

$$= \int b(s_t) \nabla p_\theta(\tau) d\tau = b(s_t) \nabla \mathbb{E}_{p_\theta(\tau)}[1] = b(s_t) \nabla [1] = 0$$

Variance Reduction: Baseline

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) Q_{\pi_\theta}(s_t, a_t)\Big]$$

Consider some baseline $b(s_t)$ and compute $\mathbb{E}_{p_\theta(\tau)}\big[b(s_t)\nabla \log \pi(a_t \mid s_t)\big]$:

$$\mathbb{E}_{p_\theta(\tau)}\big[b(s_t)\nabla \log \pi(a_t \mid s_t)\big] = \int p_\theta(\tau) b(s_t) \nabla \log p_\theta(\tau) d\tau =$$

$$= \int b(s_t) \nabla p_\theta(\tau) d\tau = b(s_t) \nabla \mathbb{E}_{p_\theta(\tau)}[1] = b(s_t) \nabla [1] = 0$$

$$Var(X + Y) = Var(X) + Var(Y) + 2cov(X, Y)$$

Variance Reduction: Baseline

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t)[Q^{\pi_\theta}(s_t, a_t) - b(s_t)]\Big]$$

Variance Reduction: Baseline

$$\nabla J_{PG}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t)[Q^{\pi_\theta}(s_t, a_t) - b(s_t)]\Big]$$

Typically we take $V^{\pi_\theta}(s)$ as a baseline so $Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s) = A^{\pi_\theta}(s,a)$ is an advantage function, a relative measure of action's utility.

Actor-Critic

$$\nabla J_{AC}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) A^{\pi_\theta}(s_t, a_t)\Big]$$

We could approximate $A^{\pi_\theta}(s, a)$ with a neural network. However it's not obvious which target we should choose as there are no Bellman equations for A.

Actor-Critic

$$\nabla J_{AC}(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \mid s_t) A^{\pi_\theta}(s_t, a_t)\Big]$$

1. Approximate V^{π_θ} with with another neural network V^{ϕ}

2. For the transition (s, a, r, s'):

$$A^{\pi_\theta}(s, a) = Q^{\pi_\theta}(s, a) - V^{\pi_\theta}(s) = \mathbb{E}_{s' \sim p(.|s,a)}[r + \gamma V(s')] \approx r + \gamma V^{\phi}(s') - V^{\phi}(s)$$

Advantage Actor-Critic

- Generate trajectories $\{\tau_i\}_{i=1}^{N}$ following π_θ
- Policy improvement:

  Estimate gradient and make gradient ascent step:

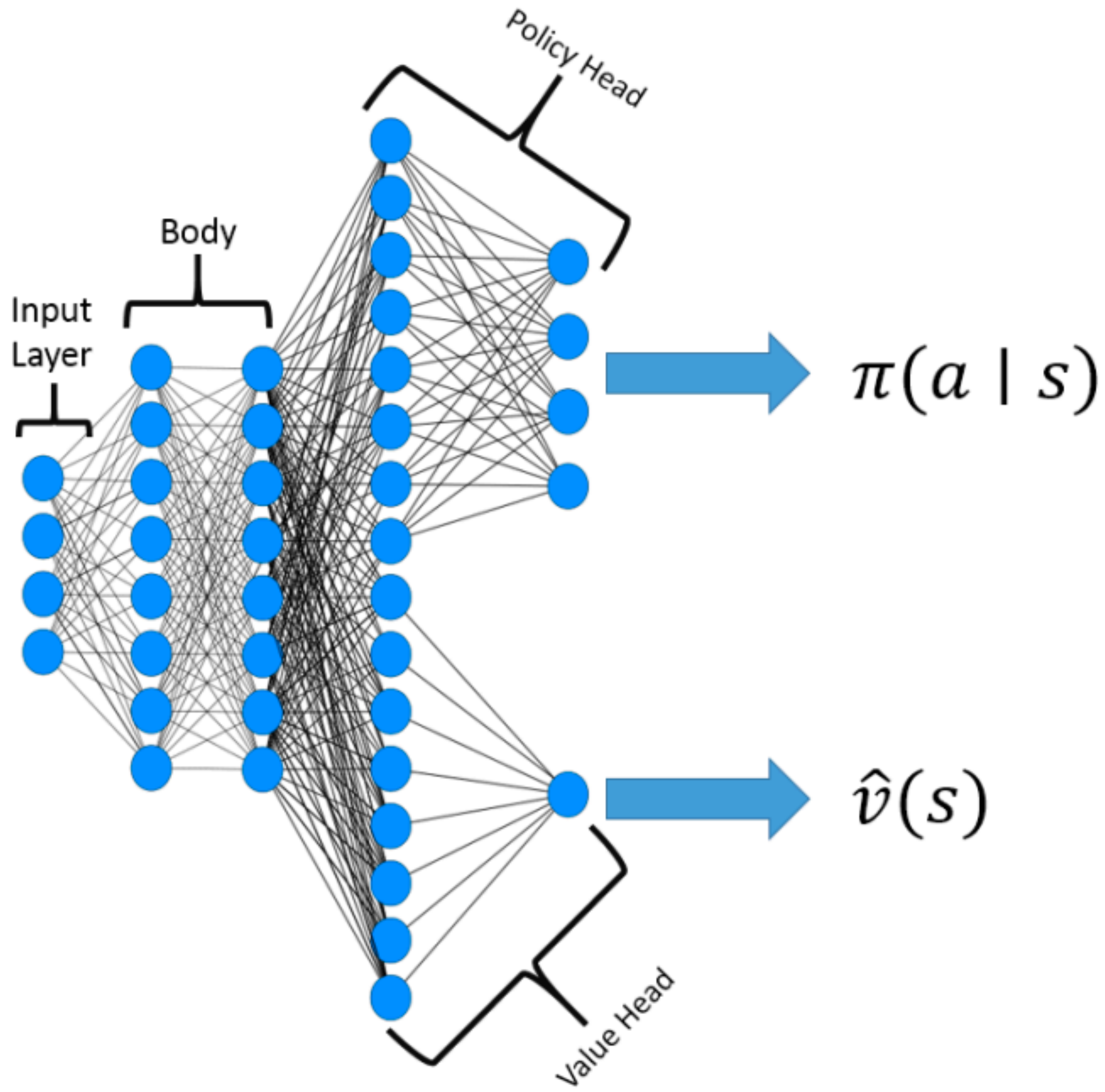
$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\Big[\sum_{t=0}^{T}\nabla \log \pi_\theta(a_{i,t}\,|\,s_{i,t})A^{\phi}(s_{i,t}, a_{i,t})\Big]$$

- Policy evaluation:

  Estimate gradient and make gradient descent step:

$$\nabla_\phi L(\phi) \approx \frac{1}{N}\sum_{i=1}^{N}\Big[\sum_{t=0}^{T}\nabla_\phi(r_{i,t} + \gamma \boxed{V_{\phi^-}}(s_{i,t+1}) - V_\phi(s_{i,t}))^2\Big]$$

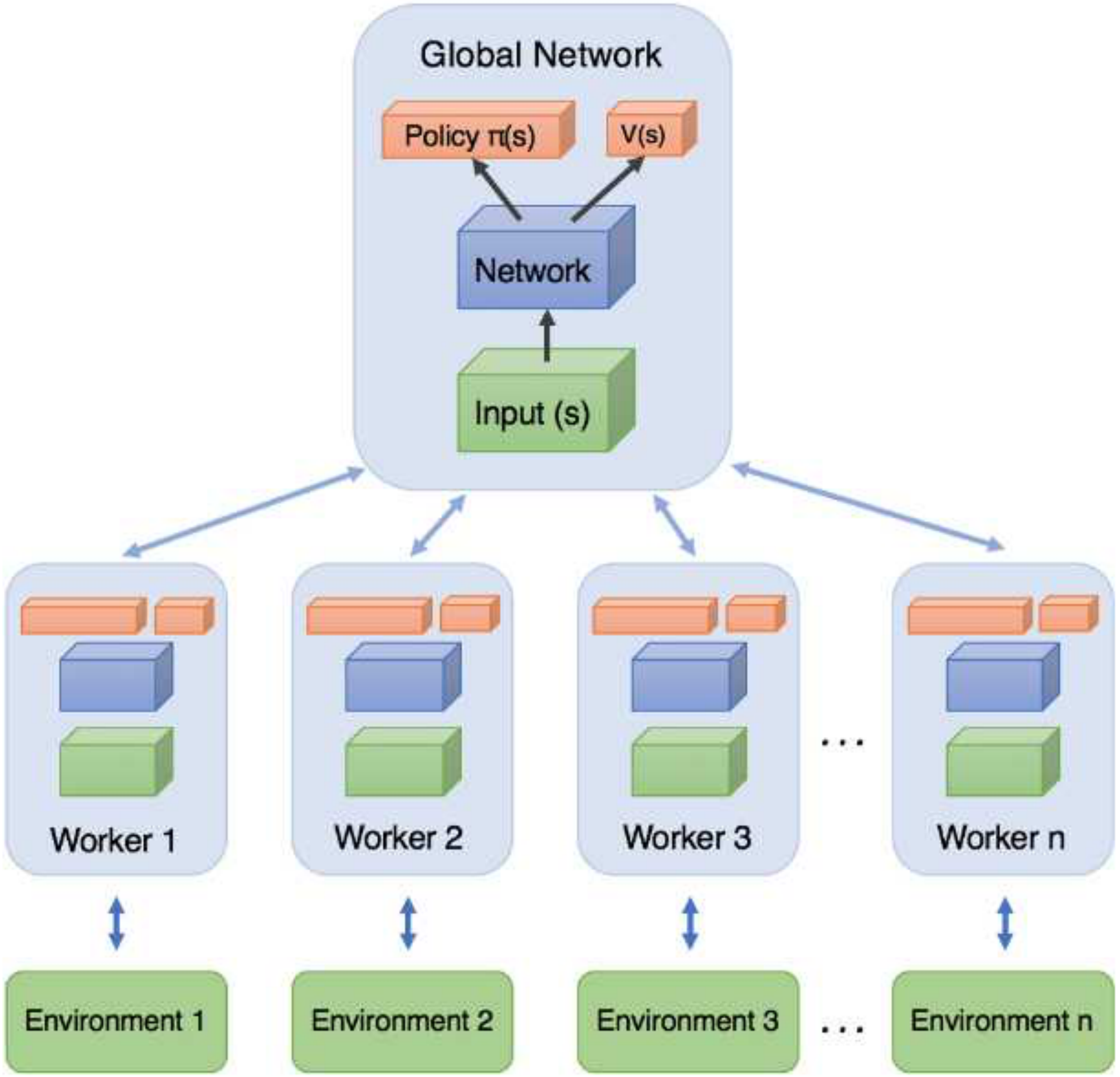
Not target network, just frozen parameters from the previous step



Source

Asynchronous Advantage Actor-Critic (A3C)

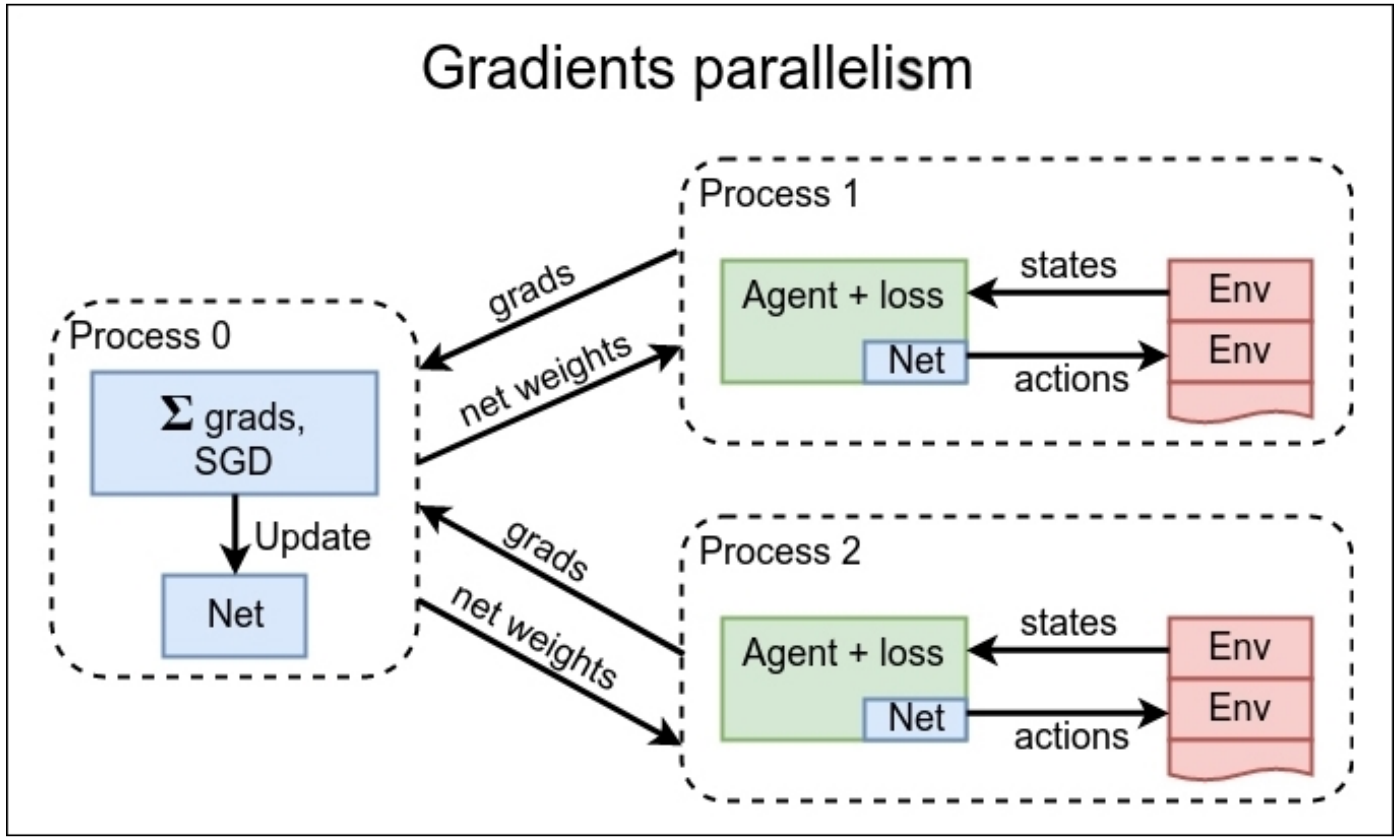
- N-step advantage estimation
- LSTM network
- No experience replay
- Entropy regularisation



Source

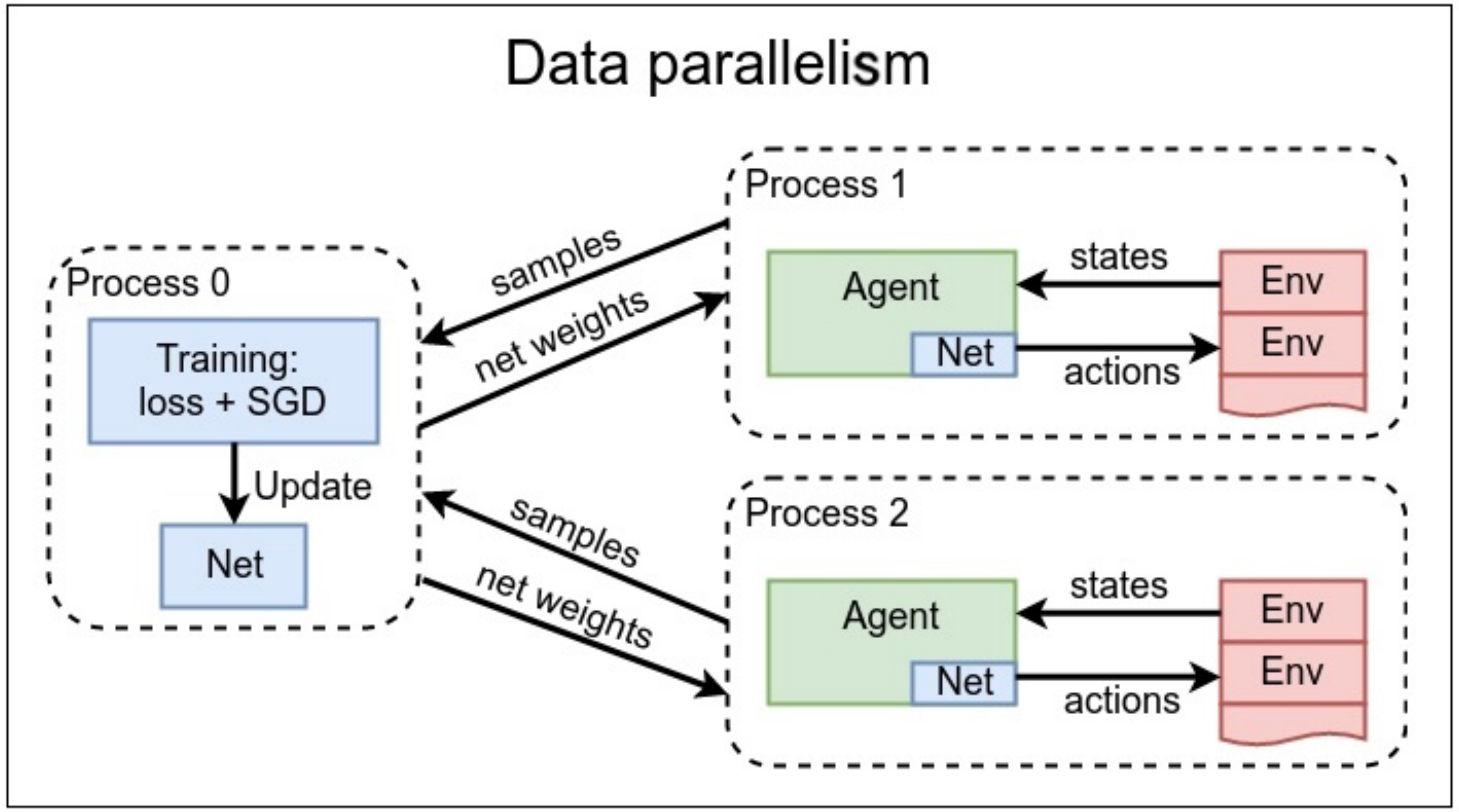
Asynchronous vs Parallel

A3C



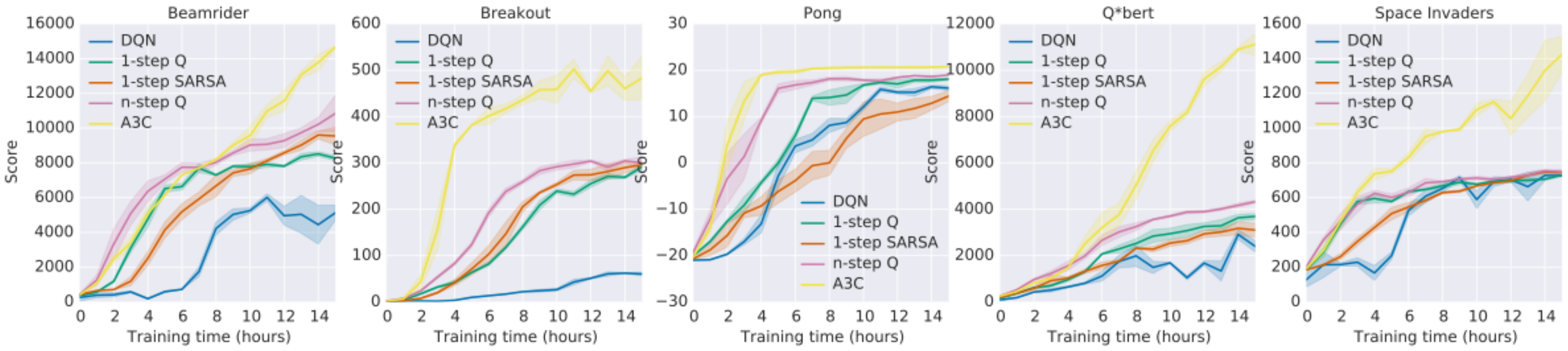
Source

A2C



Source

Comparison



Method	Training Time	Mean	Median
DQN	8 days on GPU	121.9%	47.5%
Gorila	4 days, 100 machines	215.2%	71.3%
D-DQN	8 days on GPU	332.9%	110.9%
Dueling D-DQN	8 days on GPU	343.8%	117.1%
Prioritized DQN	8 days on GPU	463.6%	127.6%
A3C, FF	1 day on CPU	344.1%	68.2%
A3C, FF	4 days on CPU	496.8%	116.6%
A3C, LSTM	4 days on CPU	623.0%	112.6%

Table 1. Mean and median human-normalized scores on 57 Atari games using the human starts evaluation metric. Supplementary Table SS3 shows the raw scores for all games.

Source

Background

1. Practical RL course by YSDA, week 6
2. Reinforcement Learning Textbook (in Russian): 5
3. Sutton & Barto, Chapter 13
4. DeepMind course, Lecture 9

Thank you for your attention!