

Reinforcement Learning

HSE, winter - spring 2025

Lecture 4: Policy-based RL



Sergei Laktionov
slaktionov@hse.ru
[LinkedIn](#)

Recap: Value-based Methods

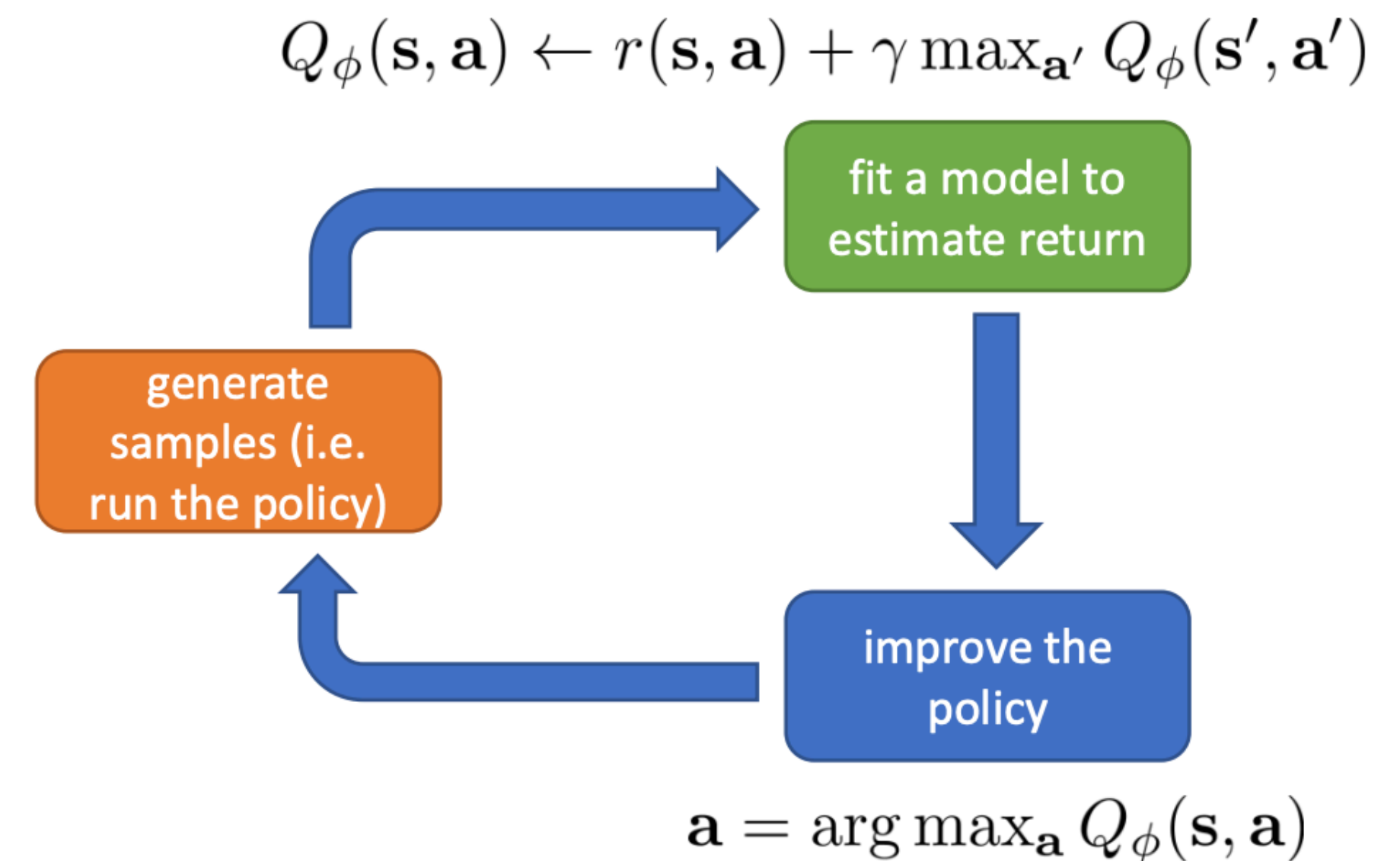
1. Approximate action-value function with a neural network: $Q(s, a) \approx Q_{\theta}(s, a)$
2. Take action which maximises $Q_{\theta}(s, a)$

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



Recap: Value-based Methods

1. Estimate 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.”

—Jason Statham

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

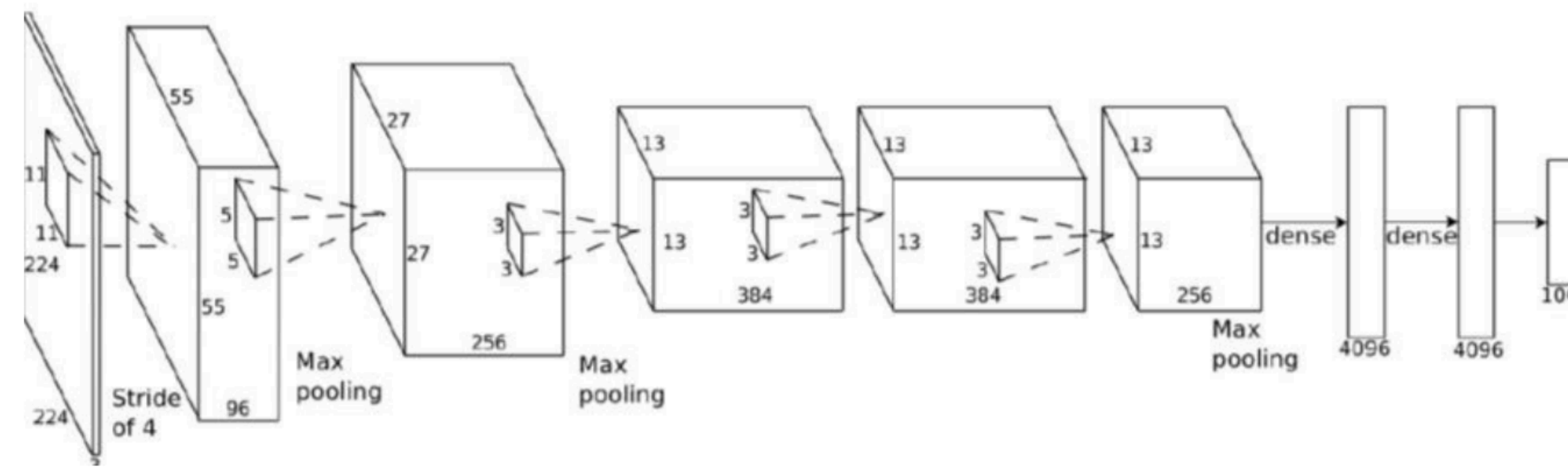
Parametric Policy

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

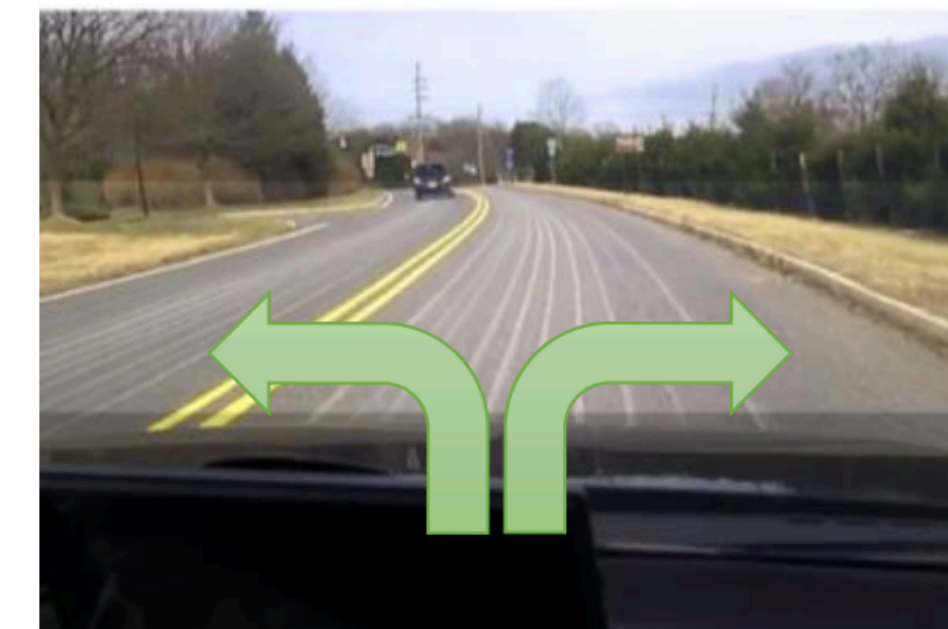
$\pi_{\theta}(a | s) = \mathbb{P}(a_t = a | s_t = s; \theta)$, where θ is some parameter.



s_t

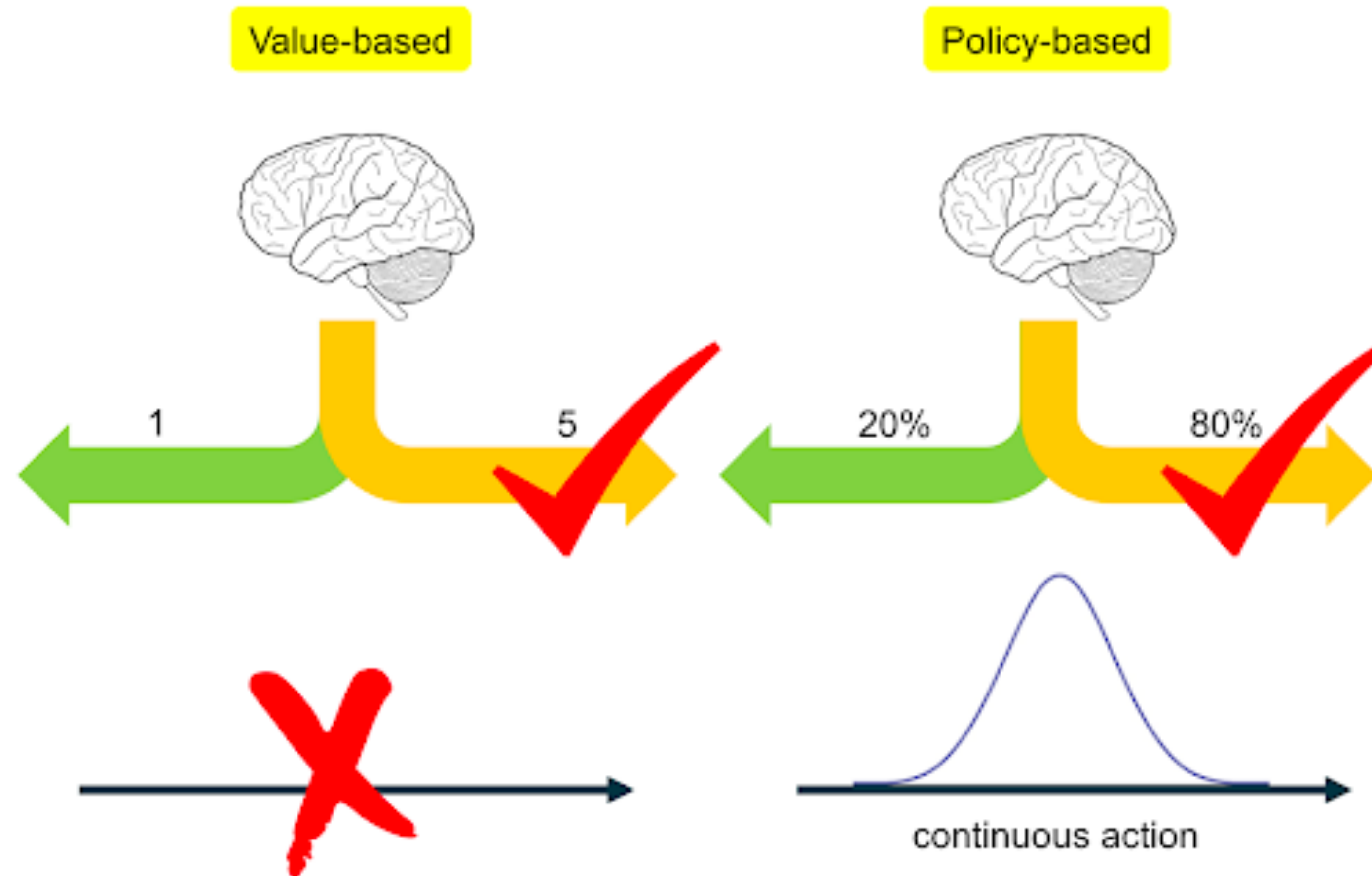


$\pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t)$



\mathbf{a}_t

Value-based vs Policy-based



Objective

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

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

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

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

Objective

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

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

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

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

$$p_{\theta}(\tau) = p(s_0) \pi_{\theta}(a_0 | s_0) p(s_1 | s_0, a_0) \dots$$

Objective's Gradient

$$p_{\theta}(\tau) = p(s_0)\pi_{\theta}(a_0 | s_0)p(s_1 | s_0, a_0)\dots$$

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

Objective's Gradient

$$p_{\theta}(\tau) = p(s_0)\pi_{\theta}(a_0 | s_0)p(s_1 | s_0, a_0)\dots$$

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

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

Objective's Gradient

$$p_{\theta}(\tau) = p(s_0)\pi_{\theta}(a_0 | s_0)p(s_1 | s_0, a_0)\dots$$

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

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

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

Objective's Gradient

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

REINFORCE (1992)

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

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

2. Estimate gradient using Monte-Carlo estimator:

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

3. Make gradient ascent step:

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

REINFORCE (1992)

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

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

2. Estimate gradient using Monte-Carlo estimator:

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

3. Make gradient ascent step:

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


On-policy algorithm



No Replay Buffer

Old samples can not be used
for gradient update

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

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

Variance Reduction

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

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

Variance Reduction

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

$$\begin{aligned}\nabla J_{PG}(\theta) &= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) G(\tau) \right] = \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \boxed{\sum_{k=0}^T \gamma^k R_k} \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \sum_{k=t}^T \gamma^k R_k \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \gamma^t \sum_{k=t}^T \gamma^{k-t} R_k \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \gamma^t G_t \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \gamma^t Q^{\pi_{\theta}}(s_t, a_t) \right]\end{aligned}$$

A Closer Look at PG

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

$$\begin{aligned}\nabla J_{PG}(\theta) &= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) G(\tau) \right] = \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \boxed{\sum_{k=0}^T \gamma^k R_k} \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \sum_{k=t}^T \gamma^k R_k \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \gamma^t \sum_{k=t}^T \gamma^{k-t} R_k \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \gamma^t G_t \right] \\&= \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t=0}^T \nabla \log \pi_{\theta}(a_t | s_t) \gamma^t Q^{\pi_{\theta}}(s_t, a_t) \right] \quad \longrightarrow \quad \text{Let's ignore } \gamma^t\end{aligned}$$

A Closer Look at PG

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

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

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

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

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

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

<https://arxiv.org/abs/1906.07073>



Let's ignore γ^t

Connection to Supervised Learning

$$\mathbb{E}_{\tau \sim D} \left[\sum_{t=0}^T \log \pi_{\theta}(a_t | s_t) \right]$$

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

Connection to Supervised Learning

$$\mathbb{E}_{\tau \sim D} \left[\sum_{t=0}^T \log \pi_{\theta}(a_t | s_t) \right]$$

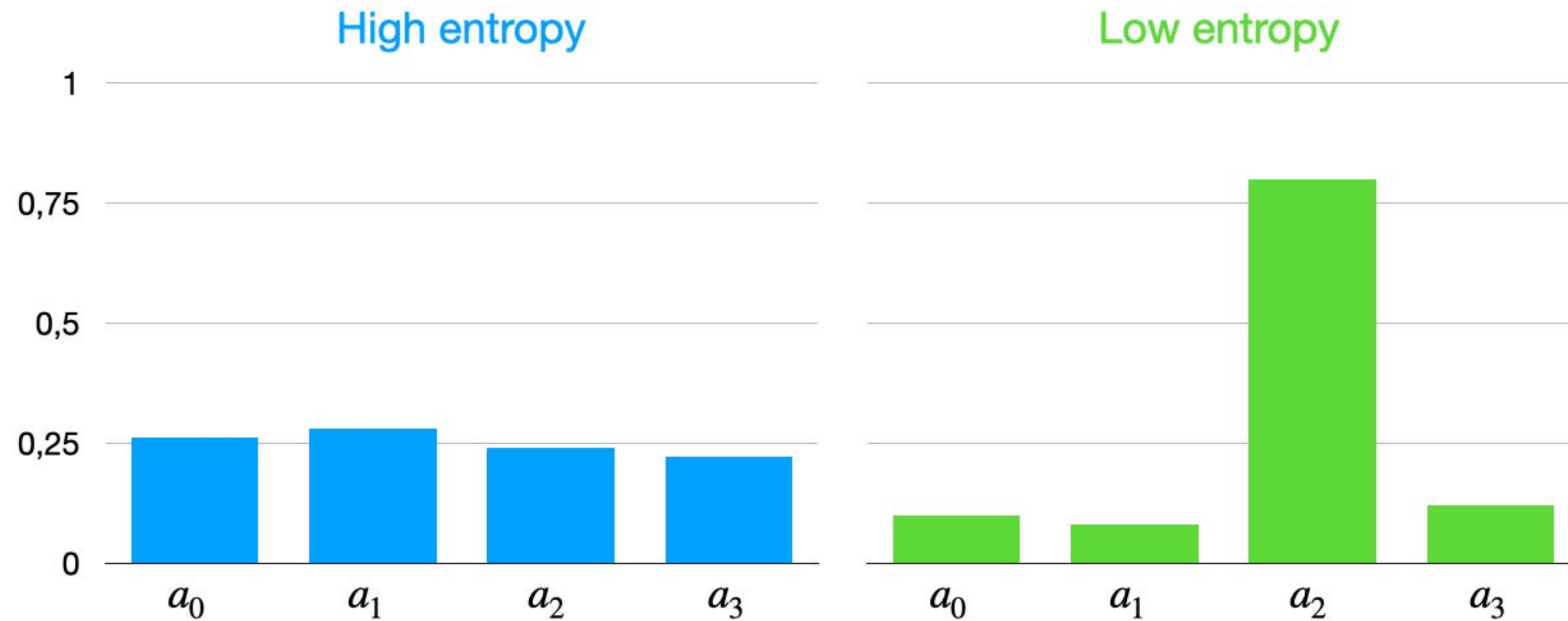
Maximise log-likelihood to take the similar actions as an expert.

$$\mathbb{E}_{p_{\theta_{old}}(\tau)} \left[\sum_{t=0}^T \log \pi_{\theta}(a_t | s_t) Q^{\theta_{old}}(s_t, a_t) \right]$$

Maximise reward-weighted log-likelihood
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.

$$H(\pi_{\theta}(\cdot | s)) = - \mathbb{E}_{\pi_{\theta}} \log \pi_{\theta}(\cdot | s) \quad \text{General case}$$

$$H(\pi_{\theta}(\cdot | s)) = - \sum_a \pi_{\theta}(a | s) \log \pi_{\theta}(a | s) \quad \text{Discrete case}$$

Entropy Regularisation

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

$$H(\pi_{\theta}(\cdot | s)) = - \mathbb{E}_{\pi_{\theta}} \log \pi_{\theta}(\cdot | s) \quad \text{General case}$$

$$H(\pi_{\theta}(\cdot | s)) = - \sum_a \pi_{\theta}(a | s) \log \pi_{\theta}(a | s) \quad \text{Discrete case}$$

We can add regularisation term $\rho H(\pi_{\theta}(\cdot | s))$ to our objective:

- To encourage agent's curiosity

Variance Reduction: Baseline

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

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

Variance Reduction: Baseline

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

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

$$\begin{aligned} \mathbb{E}_{p_{\theta}(\tau)} [b(s_t) \nabla \log \pi(a_t | s_t)] &= \int p_{\theta}(\tau) b(s_t) \nabla \log p_{\theta}(\tau) d\tau = \\ &= \int b(s_t) \nabla p_{\theta}(\tau) d\tau = b(s_t) \nabla \mathbb{E}_{p_{\theta}(\tau)}[1] = b(s_t) \nabla [1] = 0 \end{aligned}$$

Variance Reduction: Baseline

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

Variance Reduction: Baseline

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

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

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

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

Actor-Critic

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

1. Approximate $V^{\pi_{\theta}}$ with another neural network V^{ϕ}
2. For the transition (s, a, r, s') :

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

Advantage Actor-Critic

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

- Policy improvement:

Estimate gradient and make gradient ascent step:

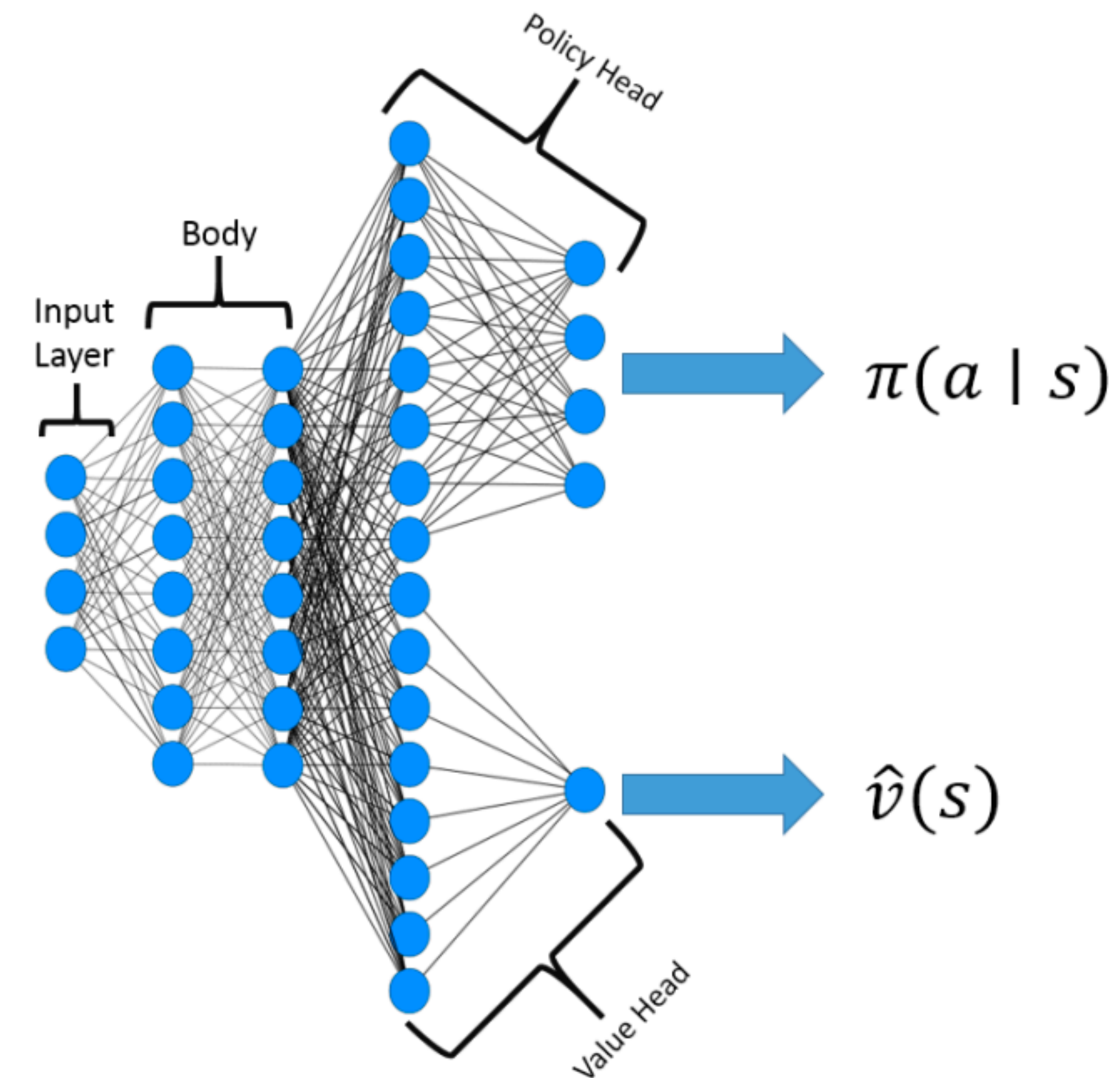
$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^N \left[\sum_{t=0}^T \nabla \log \pi_\theta(a_{i,t} | s_{i,t}) A^\phi(s_{i,t}, a_{i,t}) \right]$$

- Policy evaluation:

Estimate gradient and make gradient descent step:

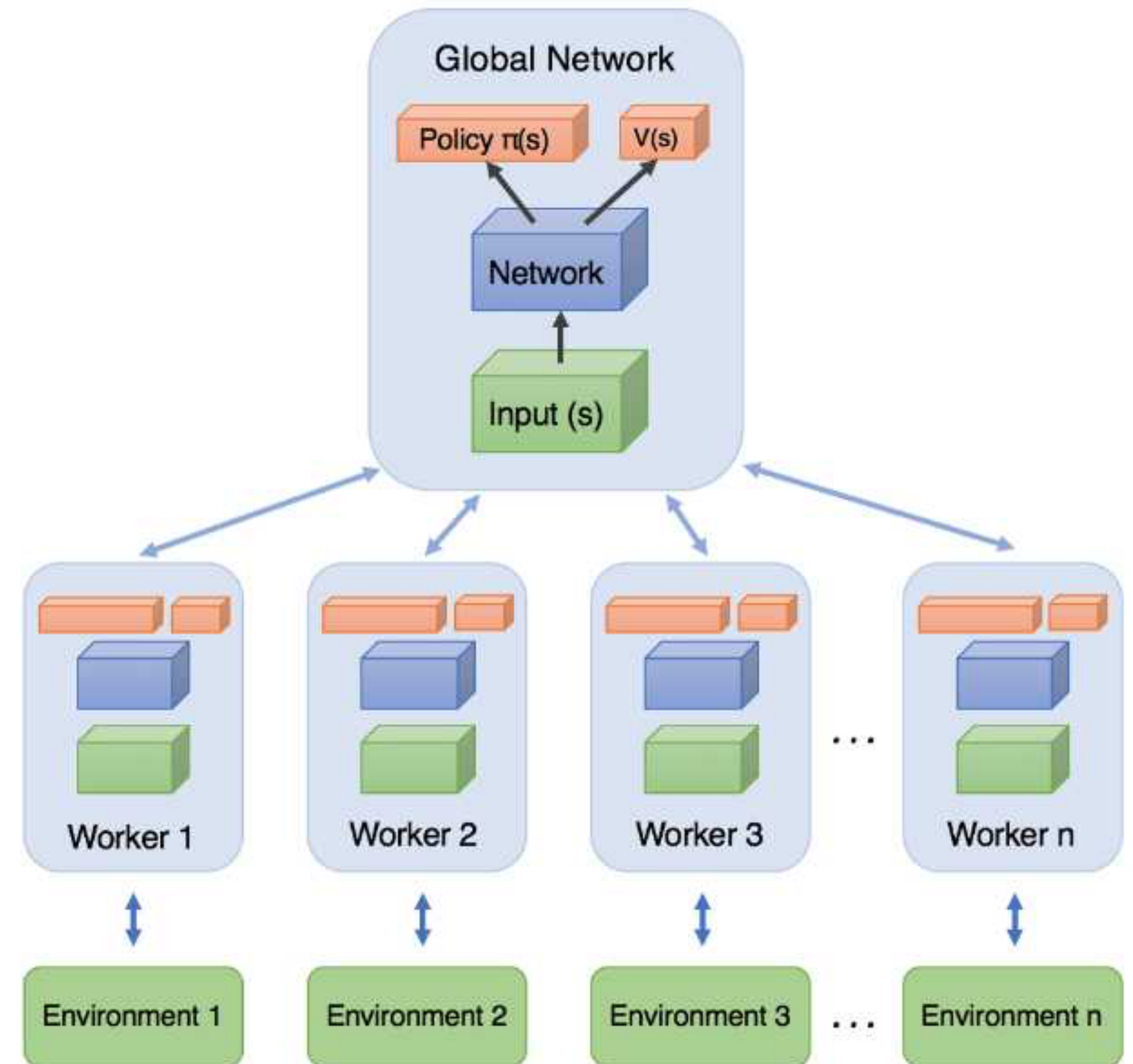
$$\nabla_\phi L(\phi) \approx \frac{1}{N} \sum_{i=1}^N \left[\sum_{t=0}^T \nabla_\phi (r_{i,t} + \gamma \boxed{V_{\phi^-}(s_{i,t+1})} - V_\phi(s_{i,t}))^2 \right]$$

Not a target network, just a frozen network from the previous step



Asynchronous Advantage Actor-Critic (A3C)

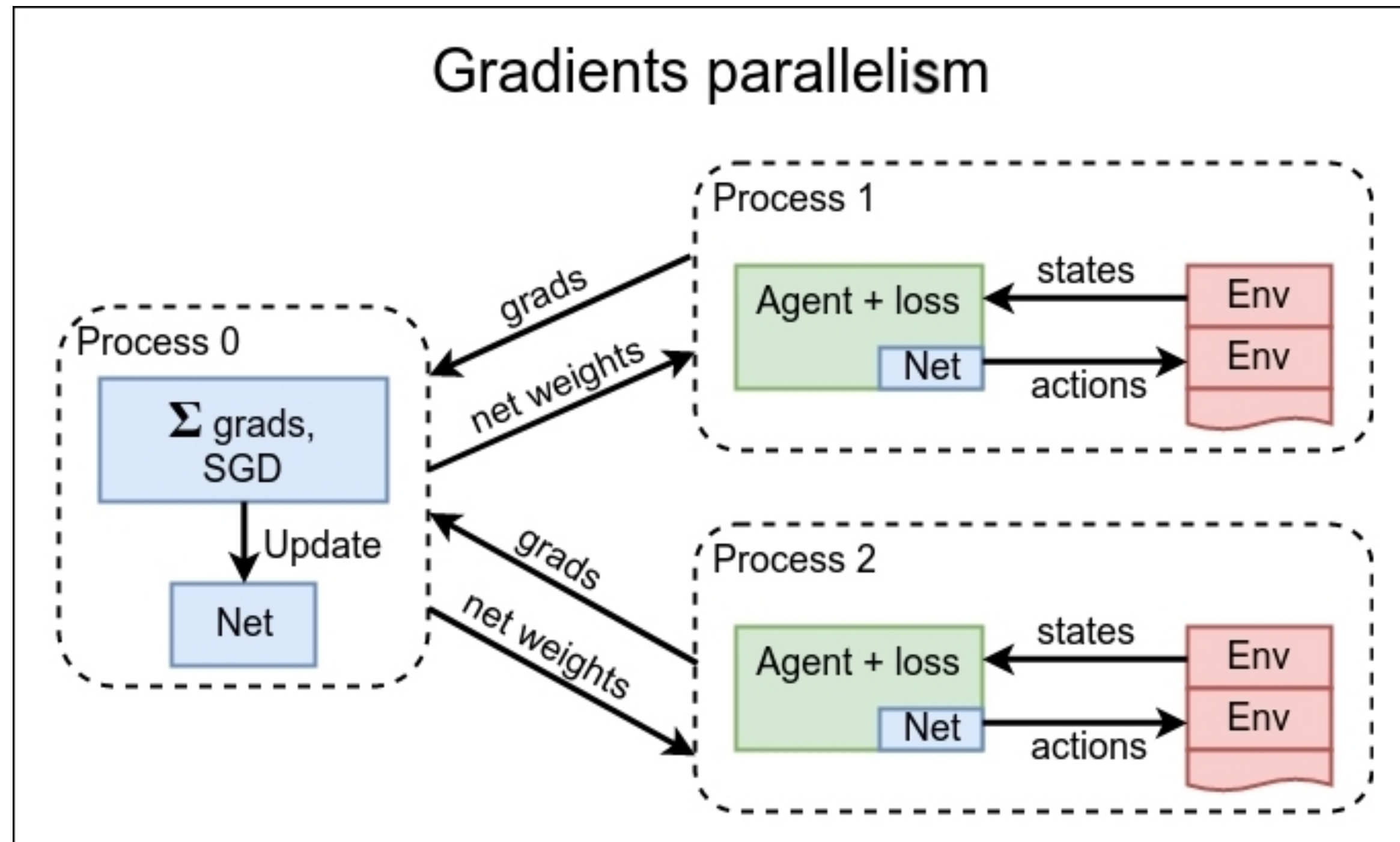
- N-step advantage estimation
- LSTM network
- No experience replay
- Entropy regularisation



Source

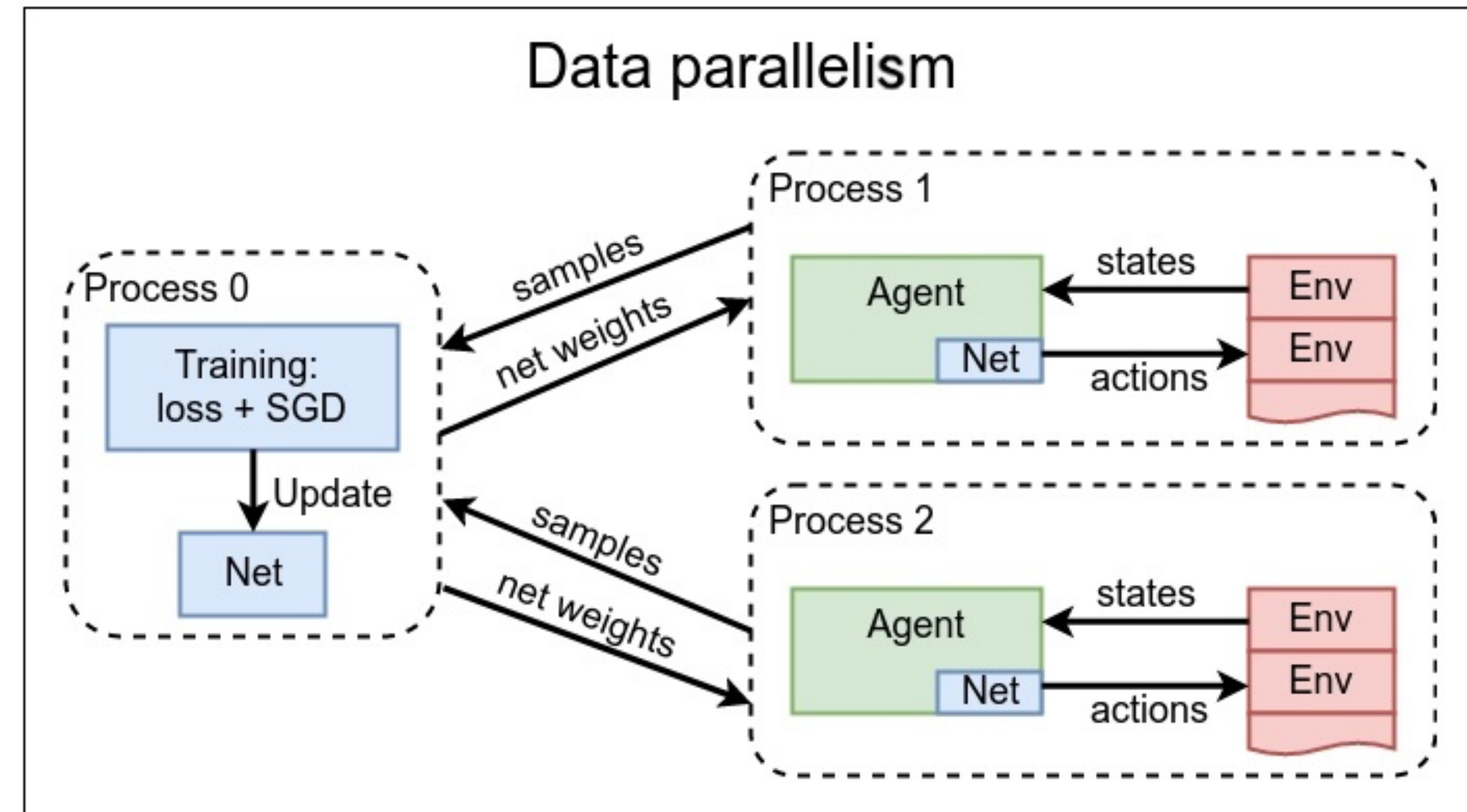
Asynchronous vs Parallel

A3C



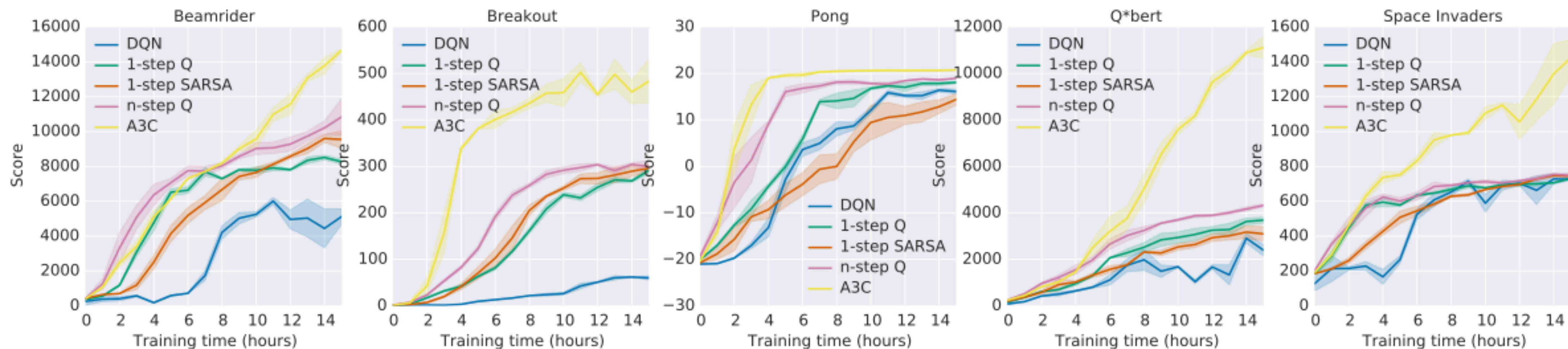
[Source](#)

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 S3 shows the raw scores for all games.

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!