CSCE 636 Neural Networks (Deep Learning)

Lecture 19: Deep Reinforcement Learning (continued)

Anxiao (Andrew) Jiang

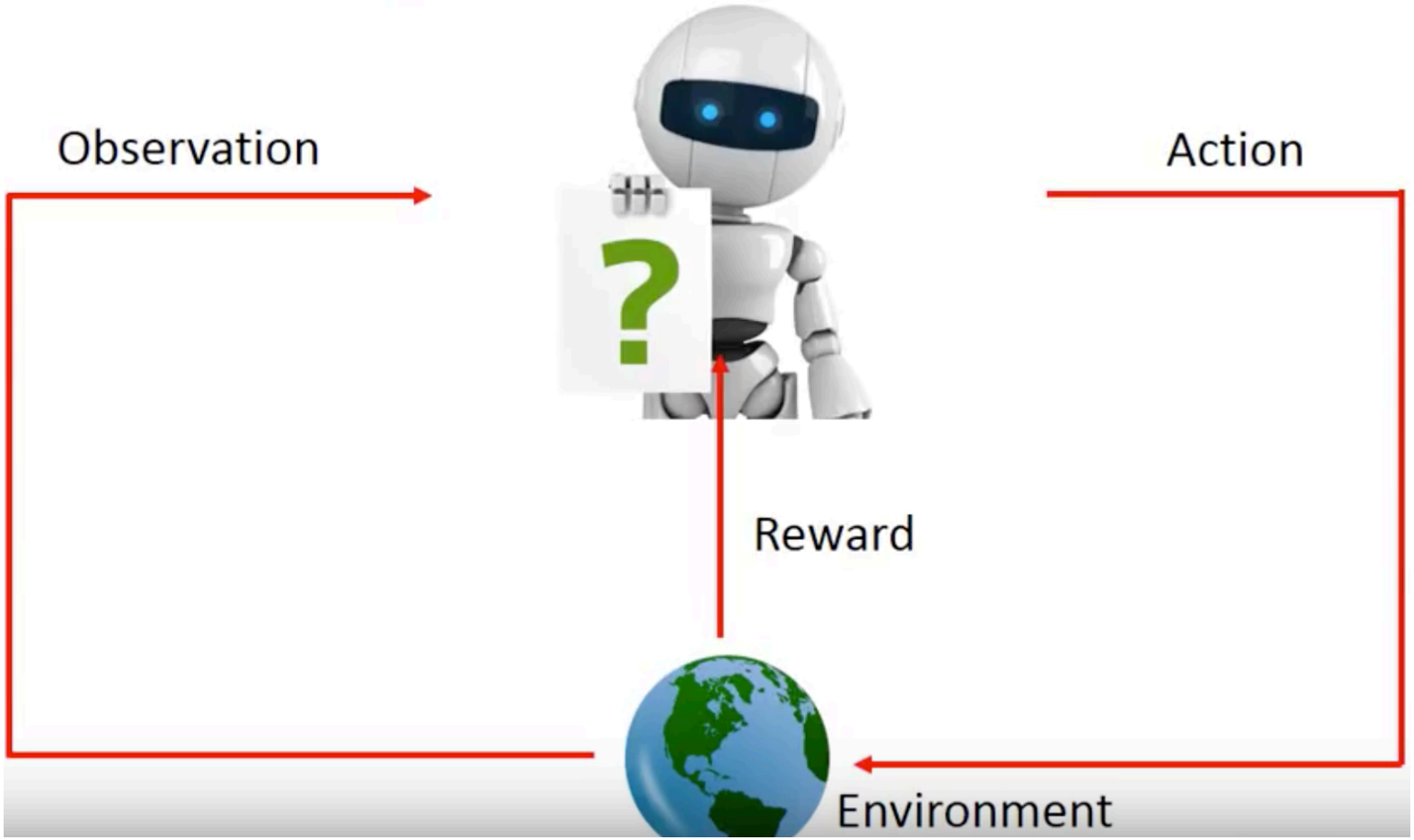
Based on the interesting lecture of Prof. Hung-yi Lee "Deep Reinforcement Learning"
https://www.youtube.com/watch?v=W8XF3ME8G2I&list=PLJV_el3uVTsPy9oCRY30oBPNLCo89yu49&index=33

Policy-based Approach

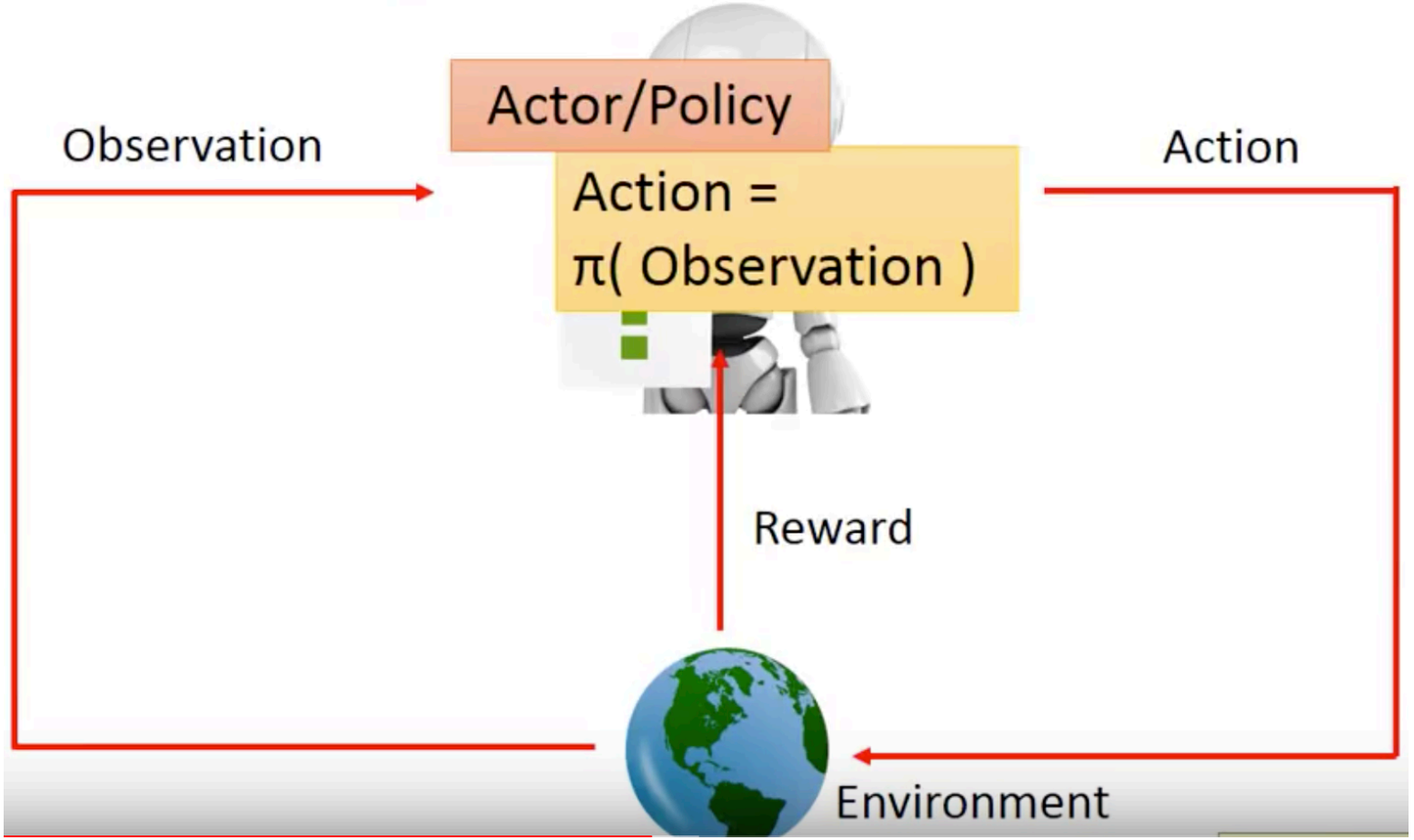
Learning an Actor

Note: Actor means “Agent”

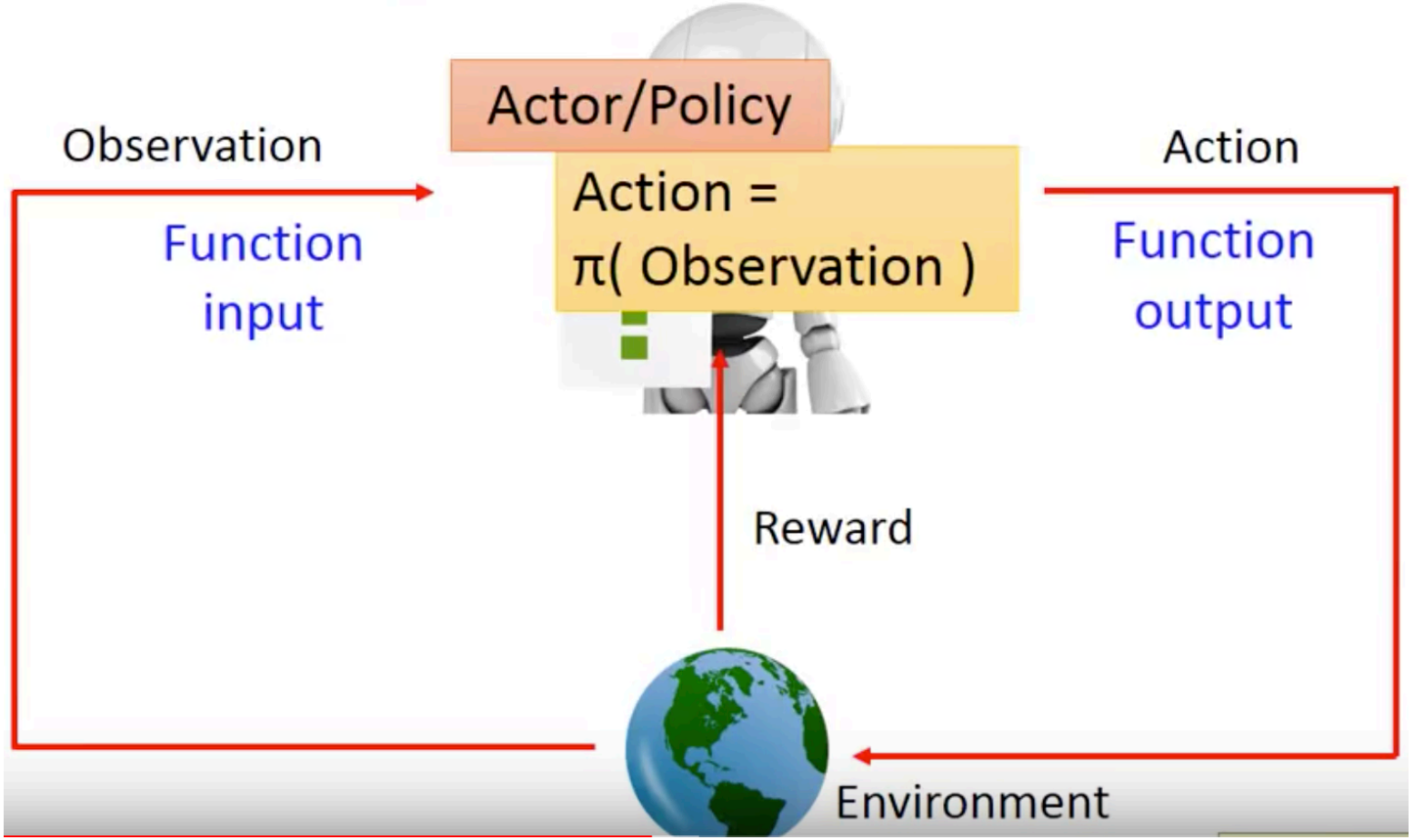
Machine Learning
≈ Looking for a Function



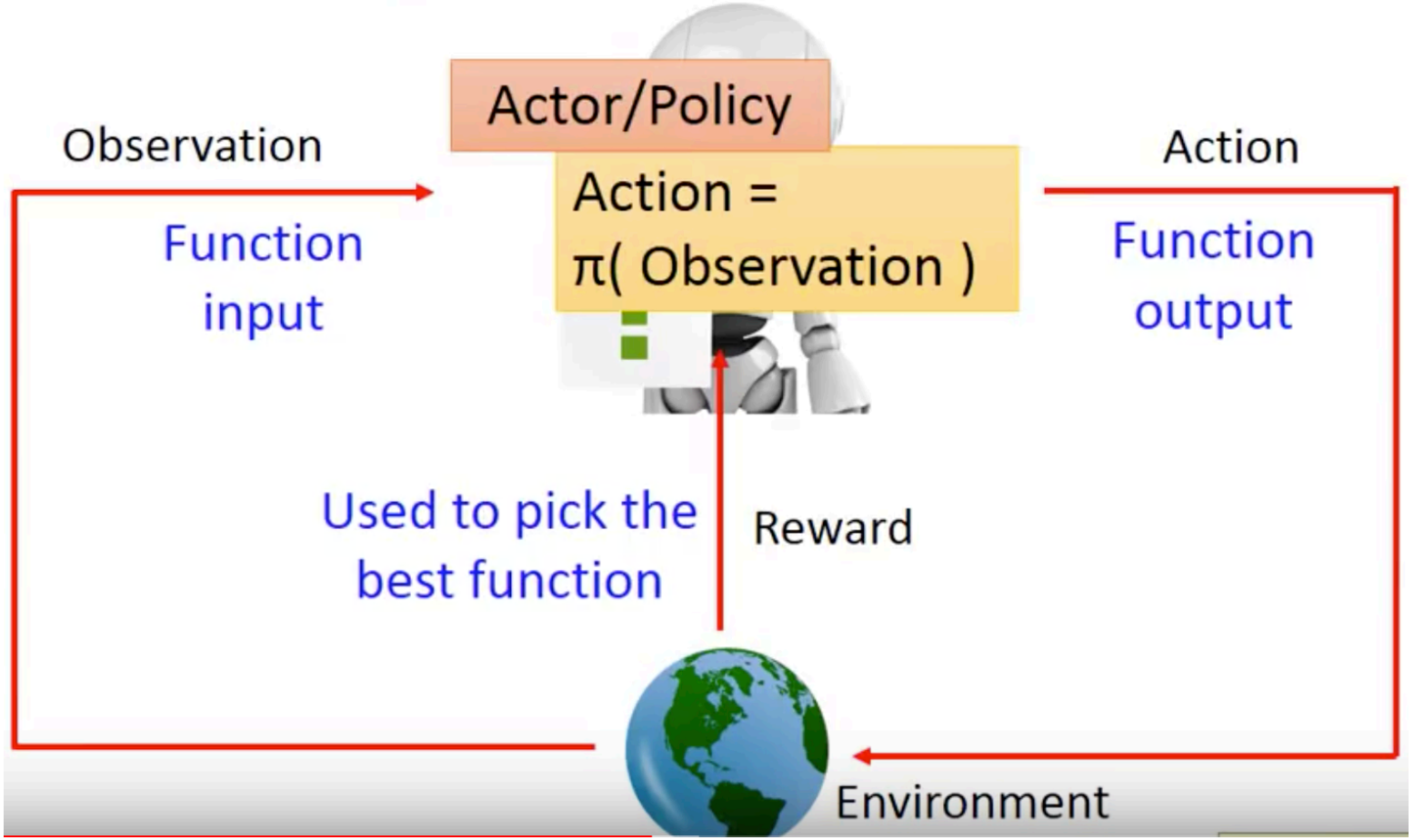
Machine Learning
≈ Looking for a Function



Machine Learning
≈ Looking for a Function



Machine Learning
≈ Looking for a Function



Neural network as Actor

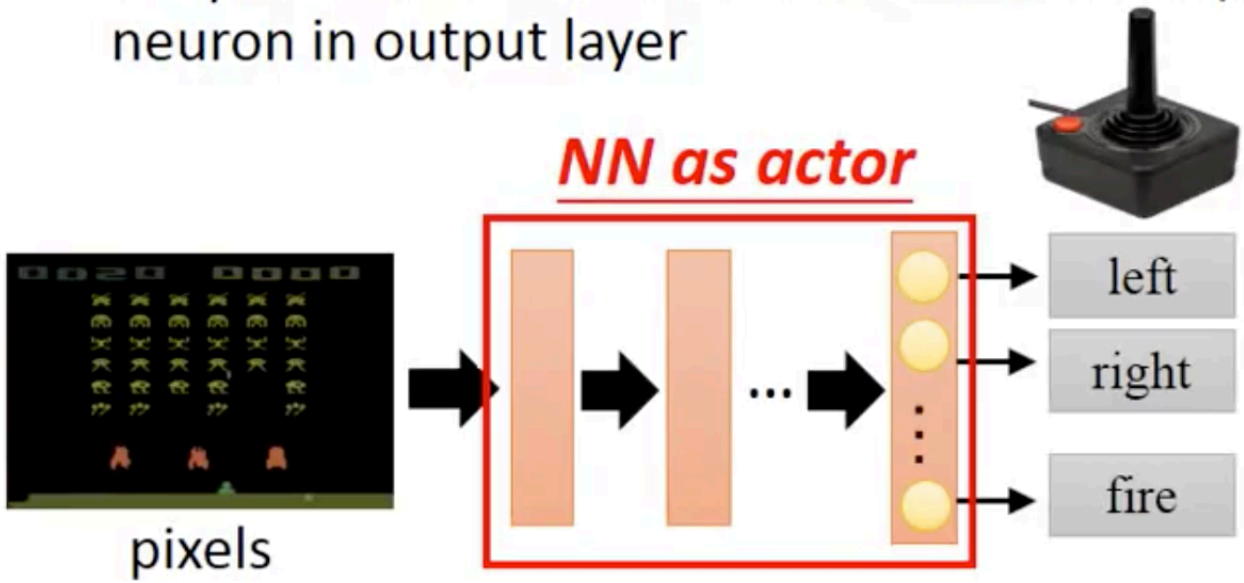
- Input of neural network: the observation of machine represented as a vector or a matrix

Neural network as Actor

- Input of neural network: the observation of machine represented as a vector or a matrix
- Output neural network : each action corresponds to a neuron in output layer

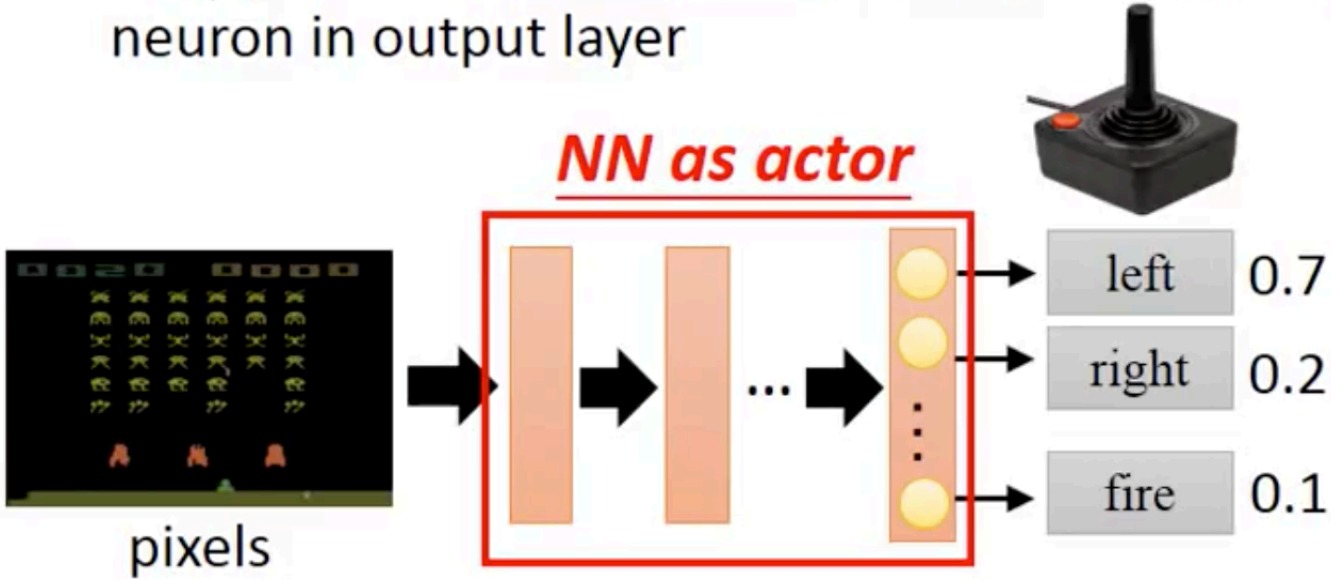
Neural network as Actor

- Input of neural network: the observation of machine represented as a vector or a matrix
- Output neural network : each action corresponds to a neuron in output layer



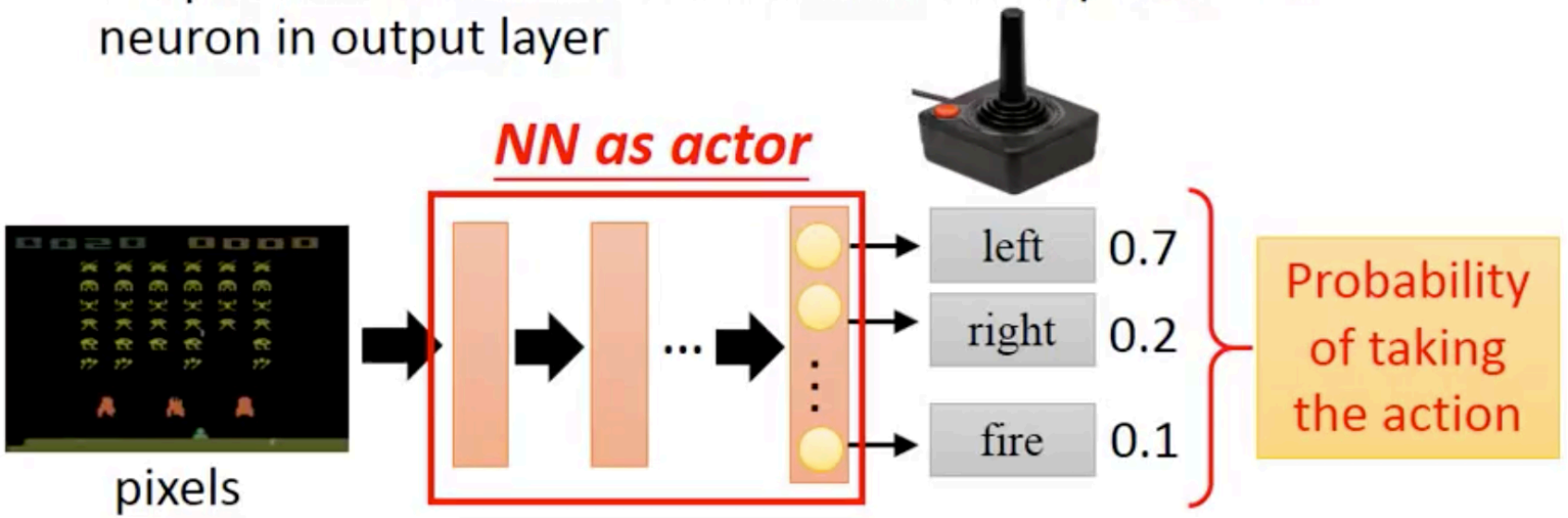
Neural network as Actor

- Input of neural network: the observation of machine represented as a vector or a matrix
- Output neural network : each action corresponds to a neuron in output layer



Neural network as Actor

- Input of neural network: the observation of machine represented as a vector or a matrix
- Output neural network : each action corresponds to a neuron in output layer

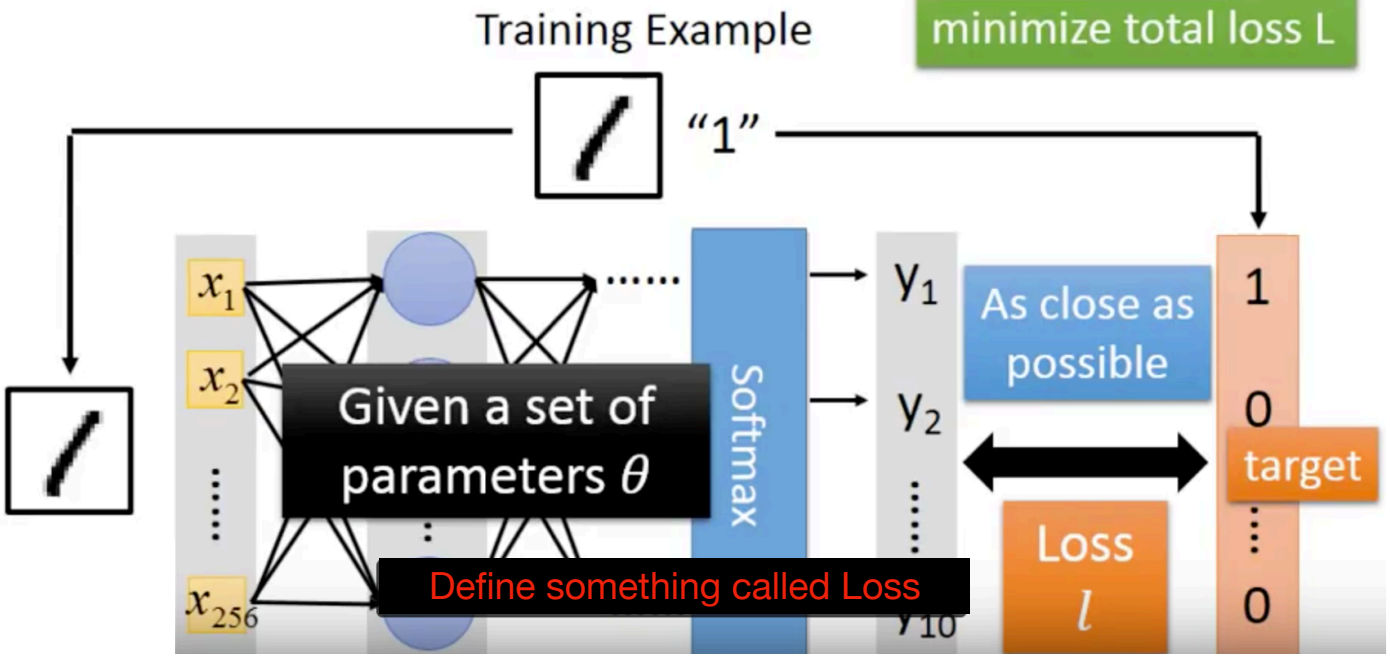


Goodness of Actor

Total Loss: $L = \sum_{n=1}^{N} l_n$

Find ***the network parameters*** θ^* that minimize total loss L

- Review: Supervised learning



Goodness of Actor

- Given an actor $\pi_\theta(s)$ with network parameter θ

Goodness of Actor

- Given an actor $\pi_\theta(s)$ with network parameter θ
- Use the actor $\pi_\theta(s)$ to play the video game
  - Start with observation s_1
  - Machine decides to take a_1
  - Machine obtains reward r_1
  - Machine sees observation s_2
  - Machine decides to take a_2
  - Machine obtains reward r_2
  - Machine sees observation s_3
  - .....
  - Machine decides to take a_T
  - Machine obtains reward r_T END

Total reward: $R_\theta = \sum_{t=1}^{T} r_t$

Goodness of Actor

- Given an actor $\pi_\theta(s)$ with network parameter θ
- Use the actor $\pi_\theta(s)$ to play the video game
  - Start with observation s_1
  - Machine decides to take a_1
  - Machine obtains reward r_1
  - Machine sees observation s_2
  - Machine decides to take a_2
  - Machine obtains reward r_2
  - Machine sees observation s_3
  - ……
  - Machine decides to take a_T
  - Machine obtains reward r_T

END

Total reward: $R_\theta = \sum_{t=1}^{T} r_t$

Even with the same actor, R_θ is different each time

Goodness of Actor

- Given an actor $\pi_\theta(s)$ with network parameter θ
- Use the actor $\pi_\theta(s)$ to play the video game
  - Start with observation s_1
  - Machine decides to take a_1
  - Machine obtains reward r_1
  - Machine sees observation s_2
  - Machine decides to take a_2
  - Machine obtains reward r_2
  - Machine sees observation s_3
  - ……
  - Machine decides to take a_T
  - Machine obtains reward r_T

END

Total reward: $R_\theta = \sum_{t=1}^{T} r_t$

Even with the same actor, R_θ is different each time

Randomness in the actor and the game

Goodness of Actor

- Given an actor $\pi_\theta(s)$ with network parameter θ
- Use the actor $\pi_\theta(s)$ to play the video game
  - Start with observation s_1
  - Machine decides to take a_1
  - Machine obtains reward r_1
  - Machine sees observation s_2
  - Machine decides to take a_2
  - Machine obtains reward r_2
  - Machine sees observation s_3
  - ……
  - Machine decides to take a_T
  - Machine obtains reward r_T

END

Total reward: $R_\theta = \sum_{t=1}^{T} r_t$

Even with the same actor, R_θ is different each time

Randomness in the actor and the game

We define $\bar{R}_\theta$ as the *expected value* of R_θ

Goodness of Actor

- An episode is considered as a trajectory τ

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$
  - If you use an actor to play the game, each τ has a probability to be sampled

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$
  - If you use an actor to play the game, each τ has a probability to be sampled
    - The probability depends on actor parameter θ: $P(\tau|\theta)$

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$
  - If you use an actor to play the game, each τ has a probability to be sampled
    - The probability depends on actor parameter θ: $P(\tau|\theta)$

$\bar{R}_\theta$

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$
  - If you use an actor to play the game, each τ has a probability to be sampled
    - The probability depends on actor parameter θ: $P(\tau|\theta)$

$$\bar{R}_\theta = \sum_\tau R(\tau) P(\tau|\theta)$$

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$
  - If you use an actor to play the game, each τ has a probability to be sampled
    - The probability depends on actor parameter θ: $P(\tau|\theta)$

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta)$$

Sum over all possible trajectory

Use π_θ to play the game N times, obtain $\{\tau^1, \tau^2, \cdots, \tau^N\}$

Sampling τ from $P(\tau|\theta)$ N times

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$
  - If you use an actor to play the game, each τ has a probability to be sampled
    - The probability depends on actor parameter θ: $P(\tau|\theta)$

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)$$

Use π_θ to play the game N times, obtain $\{\tau^1, \tau^2, \cdots, \tau^N\}$

Goodness of Actor

- An episode is considered as a trajectory τ
  - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$
  - $R(\tau) = \sum_{n=1}^{N} r_n$
  - If you use an actor to play the game, each τ has a probability to be sampled
    - The probability depends on actor parameter θ: $P(\tau|\theta)$

$$\bar{R}_\theta = \sum_\tau R(\tau) P(\tau|\theta) \approx \frac{1}{N} \sum_{n=1}^{N} R(\tau^n)$$

Use π_θ to play the game N times, obtain $\{\tau^1, \tau^2, \cdots, \tau^N\}$

Gradient Ascent

- Problem statement

$$\theta^* = arg\max_{\theta} \bar{R}_\theta \qquad \bar{R}_\theta = \sum_{\tau} R(\tau)P(\tau|\theta)$$

- Gradient ascent
  - Start with θ^0
  - $\theta^1 \leftarrow \theta^0 + \eta\nabla\bar{R}_{\theta^0}$
  - $\theta^2 \leftarrow \theta^1 + \eta\nabla\bar{R}_{\theta^1}$
  - ……

Gradient Ascent

- Problem statement

$$\theta^* = arg \max_{\theta} \bar{R}_\theta \qquad \bar{R}_\theta = \sum_{\tau} R(\tau) P(\tau|\theta)$$

- Gradient ascent
  - Start with θ^0
  - $\theta^1 \leftarrow \theta^0 + \eta \nabla \bar{R}_{\theta^0}$
  - $\theta^2 \leftarrow \theta^1 + \eta \nabla \bar{R}_{\theta^1}$
  - ……

$$\theta = \{w_1, w_2, \cdots, b_1, \cdots\}$$

$$\nabla \bar{R}_\theta$$

Gradient Ascent

- Problem statement

$$\theta^* = arg\max_{\theta} \bar{R}_\theta \qquad \bar{R}_\theta = \sum_{\tau} R(\tau)P(\tau|\theta)$$

- Gradient ascent
  - Start with θ^0
  - $\theta^1 \leftarrow \theta^0 + \eta\nabla\bar{R}_{\theta^0}$
  - $\theta^2 \leftarrow \theta^1 + \eta\nabla\bar{R}_{\theta^1}$
  - ……

$$\theta = \{w_1, w_2, \cdots, b_1, \cdots\}$$

$$\nabla\bar{R}_\theta = \begin{bmatrix} \partial\bar{R}_\theta/\partial w_1 \\ \partial\bar{R}_\theta/\partial w_2 \\ \vdots \\ \partial\bar{R}_\theta/\partial b_1 \\ \vdots \end{bmatrix}$$

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta)$$

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta)$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta) = \sum_\tau R(\tau)P(\tau|\theta)\frac{\nabla P(\tau|\theta)}{P(\tau|\theta)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta) = \sum_\tau R(\tau)P(\tau|\theta)\frac{\nabla P(\tau|\theta)}{P(\tau|\theta)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \sum_\tau R(\tau)P(\tau|\theta)\nabla log P(\tau|\theta)$$

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta) = \sum_\tau R(\tau)P(\tau|\theta)\frac{\nabla P(\tau|\theta)}{P(\tau|\theta)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \sum_\tau R(\tau)P(\tau|\theta)\nabla log P(\tau|\theta)$$

$$\frac{dlog(f(x))}{dx} = \frac{1}{f(x)}\frac{df(x)}{dx}$$

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta) = \sum_\tau R(\tau)P(\tau|\theta)\frac{\nabla P(\tau|\theta)}{P(\tau|\theta)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \sum_\tau R(\tau)P(\tau|\theta)\nabla log P(\tau|\theta)$$

$$\frac{dlog(f(x))}{dx} = \frac{1}{f(x)}\frac{df(x)}{dx}$$

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta) = \sum_\tau R(\tau)P(\tau|\theta)\frac{\nabla P(\tau|\theta)}{P(\tau|\theta)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \sum_\tau R(\tau)P(\tau|\theta)\nabla log P(\tau|\theta)$$

$$\frac{dlog(f(x))}{dx} = \frac{1}{f(x)}\frac{df(x)}{dx}$$

$$\approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n) \bigtriangledown \log P(\tau^n|\theta)$$

Use π_θ to play the game N times, obtain $\{\tau^1, \tau^2, \cdots, \tau^N\}$

Gradient Ascent

$\nabla log P(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

$P(\tau|\theta) =$

Gradient Ascent

$\nabla log P(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

$$P(\tau|\theta) =$$

$$p(s_1)p(a_1|s_1,\theta)p(r_1,s_2|s_1,a_1)p(a_2|s_2,\theta)p(r_2,s_3|s_2,a_2)\cdots$$

Gradient Ascent

$\nabla log P(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

$$P(\tau|\theta) =$$

$$p(s_1)p(a_1|s_1,\theta)p(r_1,s_2|s_1,a_1)p(a_2|s_2,\theta)p(r_2,s_3|s_2,a_2)\cdots$$

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

Gradient Ascent

$\nabla log P(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

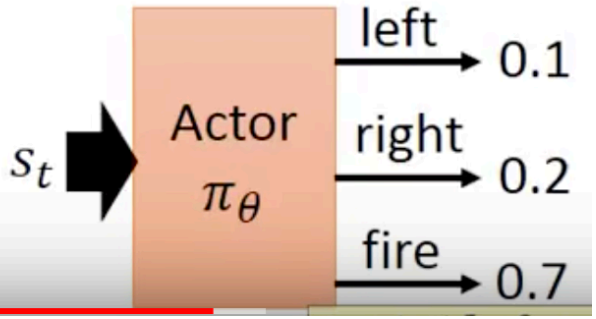
$$P(\tau|\theta) =$$

$$p(s_1)p(a_1|s_1,\theta)p(r_1,s_2|s_1,a_1)p(a_2|s_2,\theta)p(r_2,s_3|s_2,a_2)\cdots$$

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

not related to your actor

Control by your actor π_θ



Gradient Ascent

$\nabla log P(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

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

Gradient Ascent

$\nabla logP(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

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

$$logP(\tau|\theta)$$

$$= logp(s_1) + \sum_{t=1}^{T} logp(a_t|s_t,\theta) + logp(r_t, s_{t+1}|s_t, a_t)$$

Gradient Ascent

$\nabla logP(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

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

$$logP(\tau|\theta)$$

$$= logp(s_1) + \sum_{t=1}^{T} logp(a_t|s_t,\theta) + logp(r_t, s_{t+1}|s_t, a_t)$$

Ignore the terms not related to θ

Gradient Ascent

$\nabla logP(\tau|\theta) = ?$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_T, a_T, r_T\}$

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

$$logP(\tau|\theta)$$

$$= logp(s_1) + \sum_{t=1}^{T} logp(a_t|s_t,\theta) + logp(r_t, s_{t+1}|s_t, a_t)$$

$$\nabla logP(\tau|\theta) = \nabla \sum_{t=1}^{T} logp(a_t|s_t,\theta)$$

Ignore the terms not related to θ

Gradient Ascent

$$\nabla logP(\tau|\theta) = \sum_{t=1}^{T} \nabla logp(a_t|s_t,\theta)$$

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} R(\tau^n) \nabla logP(\tau^n|\theta) = \frac{1}{N} \sum_{n=1}^{N} R(\tau^n) \sum_{t=1}^{T_n} \nabla logp(a_t^n|s_t^n,\theta)$$

$$= \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla logp(a_t^n|s_t^n,\theta)$$

Gradient Ascent

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla log P(\tau|\theta) = \sum_{t=1}^{T} \nabla log p(a_t|s_t,\theta)$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\nabla log P(\tau^n|\theta) = \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\sum_{t=1}^{T_n} \nabla log p(a_t^n|s_t^n,\theta)$$

$$= \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n} R(\tau^n)\nabla log p(a_t^n|s_t^n,\theta)$$

If in τ^n machine takes a_t^n when seeing s_t^n in

Gradient Ascent

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla log P(\tau|\theta) = \sum_{t=1}^{T} \nabla log p(a_t|s_t,\theta)$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\nabla log P(\tau^n|\theta) = \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\sum_{t=1}^{T_n} \nabla log p(a_t^n|s_t^n,\theta)$$

$$= \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n} R(\tau^n)\nabla log p(a_t^n|s_t^n,\theta)$$

If in τ^n machine takes a_t^n when seeing s_t^n in

$R(\tau^n)$ is positive ⇒ Tuning θ to increase $p(a_t^n|s_t^n)$

Gradient Ascent

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla log P(\tau|\theta) = \sum_{t=1}^{T} \nabla log p(a_t|s_t,\theta)$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\nabla log P(\tau^n|\theta) = \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\sum_{t=1}^{T_n} \nabla log p(a_t^n|s_t^n,\theta)$$

$$= \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n} R(\tau^n)\nabla log p(a_t^n|s_t^n,\theta)$$

If in τ^n machine takes a_t^n when seeing s_t^n in

$R(\tau^n)$ is positive ⇒ Tuning θ to increase $p(a_t^n|s_t^n)$

$R(\tau^n)$ is negative ⇒ Tuning θ to decrease $p(a_t^n|s_t^n)$

The following lecture is based on the interesting lecture of Prof. Hung-yi Lee “Deep Reinforcement Learning”
https://www.youtube.com/watch?v=W8XF3ME8G2I&list=PLJV_el3uVTsPy9oCRY30oBPNLCo89yu49&index=33

Policy Gradient

Policy Gradient

Given actor parameter θ

Policy Gradient

Given actor parameter θ

τ^1: (s_1^1, a_1^1)

(s_2^1, a_2^1)

⋮

Policy Gradient

Given actor parameter θ

τ^1: (s_1^1, a_1^1) $R(\tau^1)$

(s_2^1, a_2^1) $R(\tau^1)$

⋮ ⋮

Policy Gradient

Given actor parameter θ

τ^1: (s_1^1, a_1^1) $R(\tau^1)$

(s_2^1, a_2^1) $R(\tau^1)$

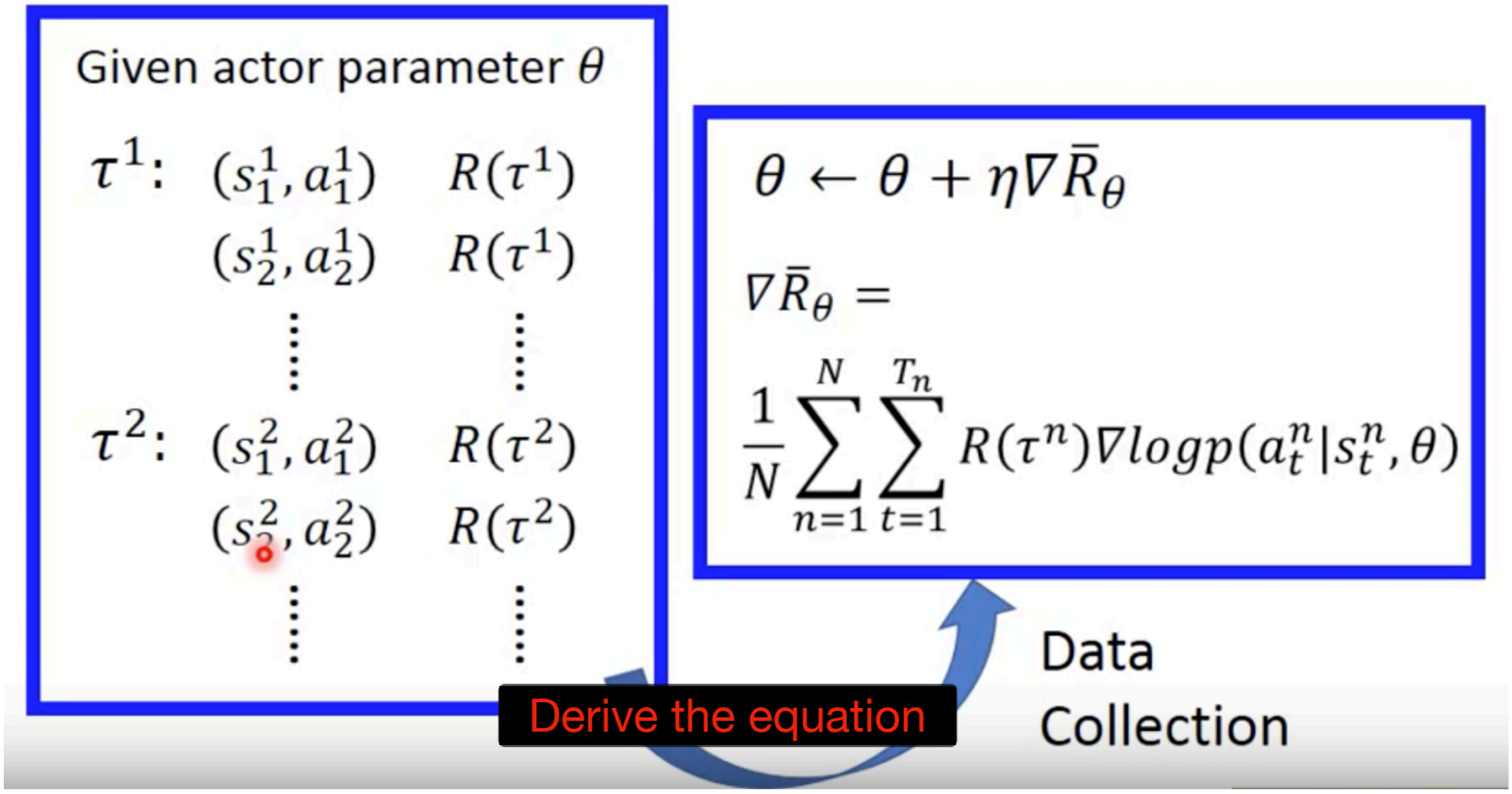
$\vdots$ $\vdots$

τ^2: (s_1^2, a_1^2) $R(\tau^2)$

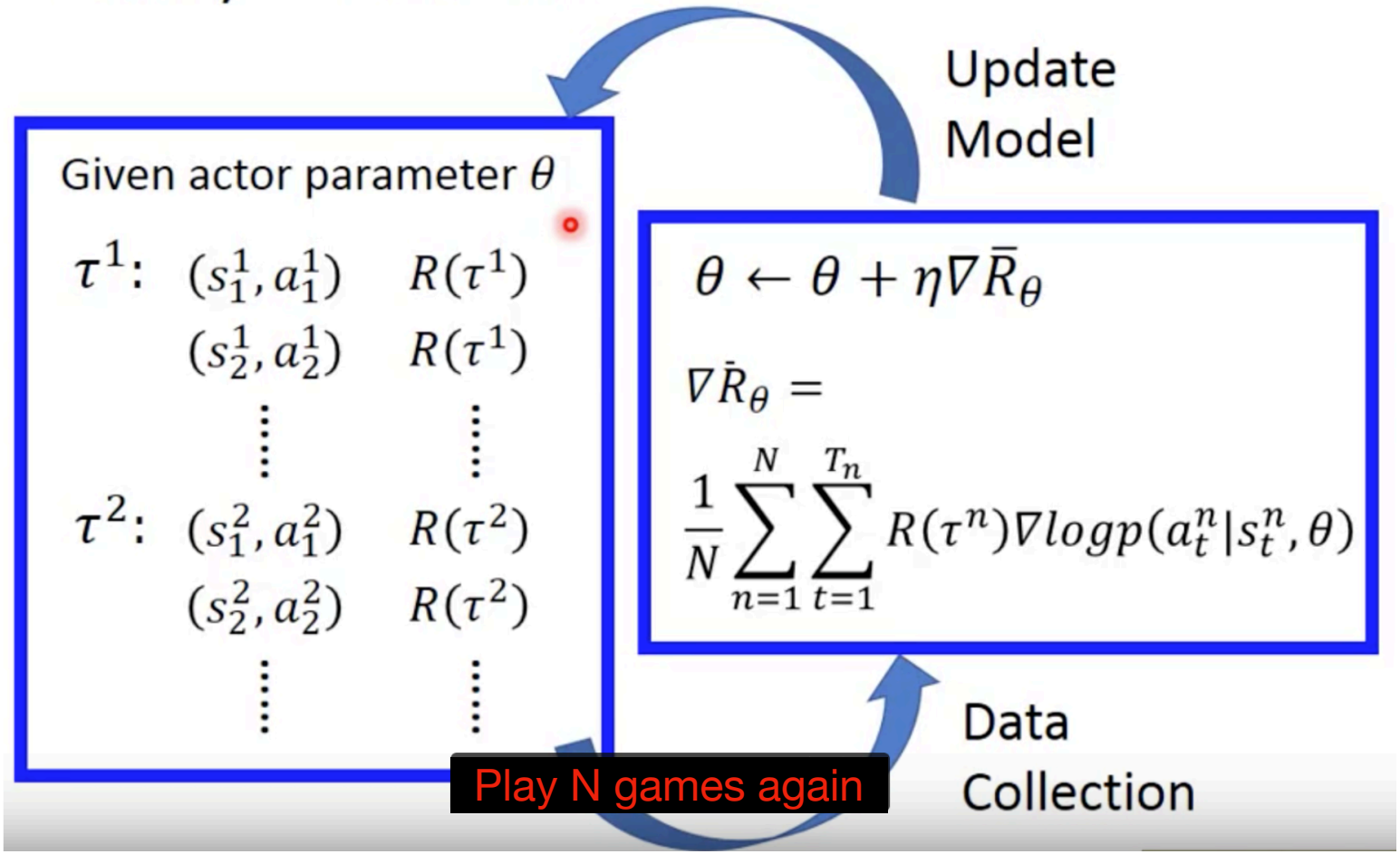
(s_2^2, a_2^2) $R(\tau^2)$

$\vdots$ $\vdots$

Policy Gradient

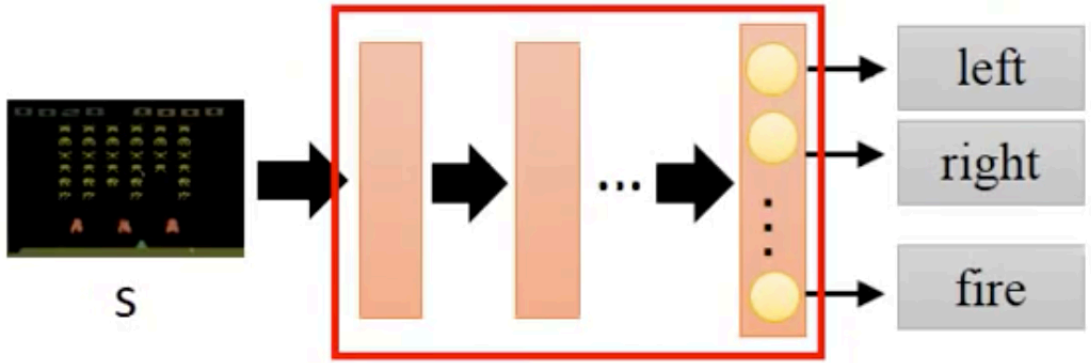


Policy Gradient



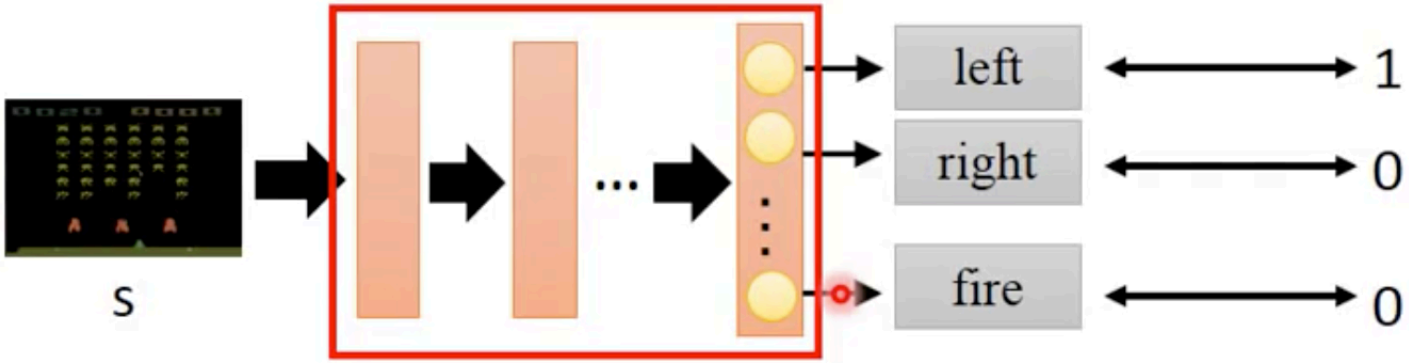
Policy Gradient

Considered as
Classification Problem



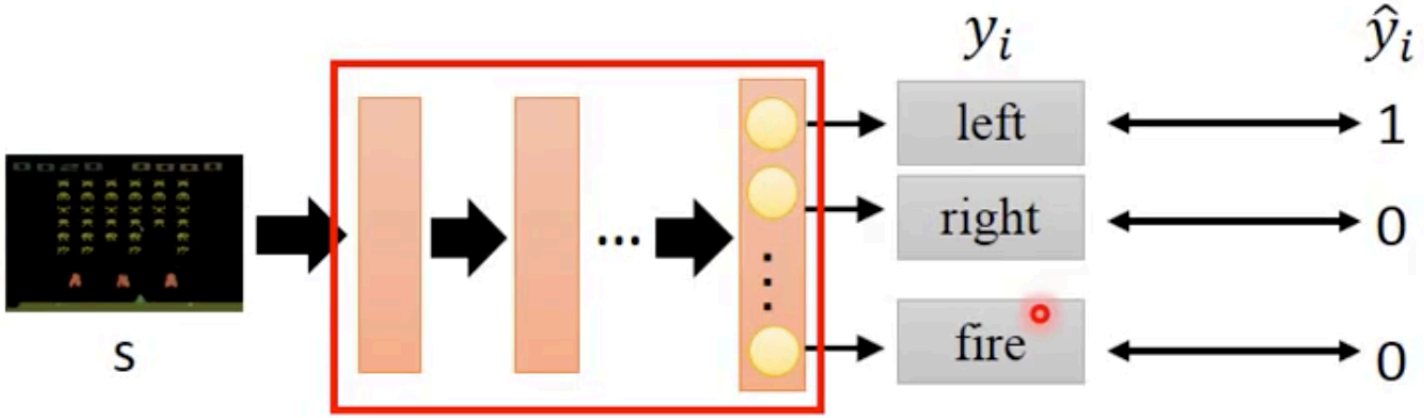
Policy Gradient

Considered as
Classification Problem



Policy Gradient

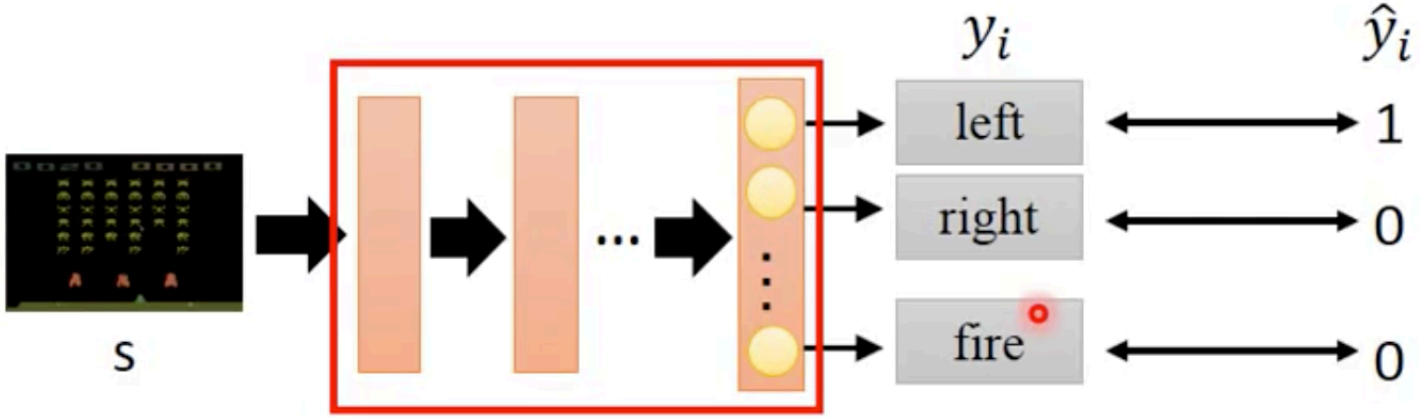
Considered as
Classification Problem



Policy Gradient

Considered as Classification Problem

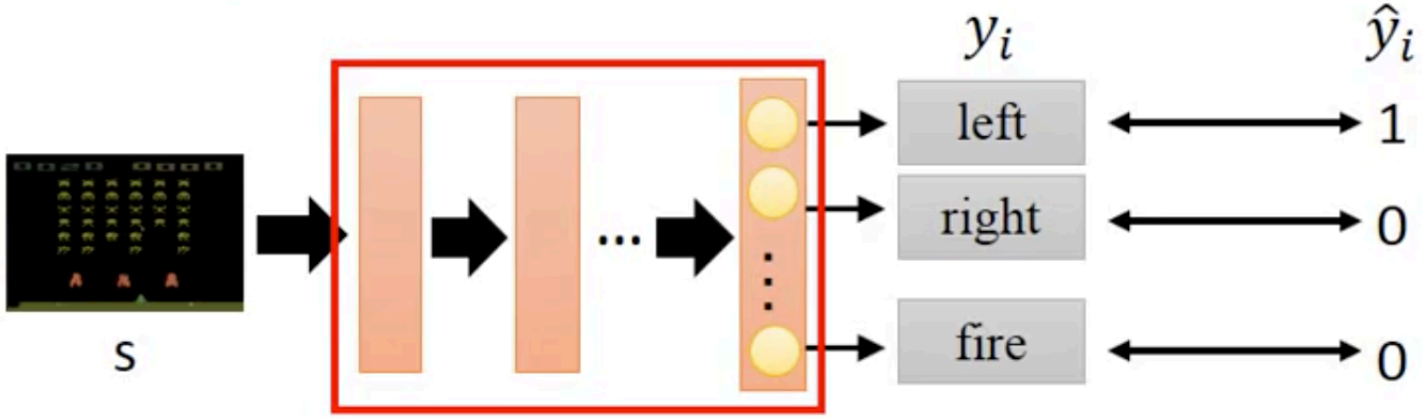
Minimize: $-\sum_{i=1}^{3} \hat{y}_i log y_i$



Policy Gradient

Considered as Classification Problem

Minimize: $-\sum_{i=1}^{3} \hat{y}_i log y_i$

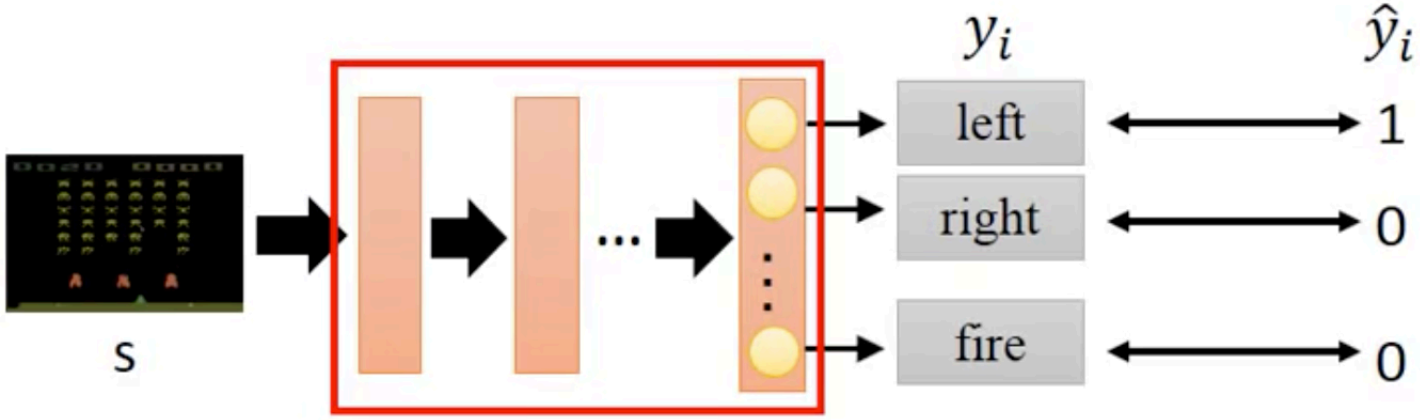


Maximize: $logy_i =$

Policy Gradient

Considered as Classification Problem

Minimize: $-\sum_{i=1}^{3} \hat{y}_i log y_i$

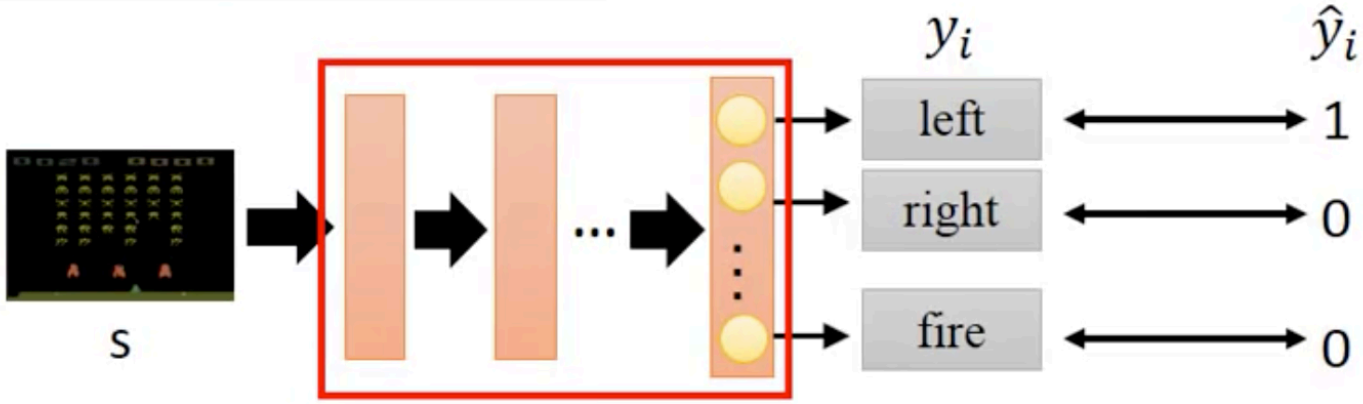


Maximize: $logy_i = logP("left"|s)$

Policy Gradient

Considered as Classification Problem

Minimize: $-\sum_{i=1}^{3} \hat{y}_i log y_i$



Maximize: $logy_i = logP("left"|s)$

$$\theta \leftarrow \theta + \eta \nabla logP("left"|s)$$

Policy Gradient

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta =$$

$$\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \blacksquare \nabla log p(a_t^n | s_t^n, \theta)$$

Given actor parameter θ

τ^1:	(s_1^1, a_1^1)	$R(\tau^1)$
	(s_2^1, a_2^1)	$R(\tau^1)$
	⋮	⋮
τ^2:	(s_1^2, a_1^2)	$R(\tau^2)$
	(s_2^2, a_2^2)	$R(\tau^2)$
	⋮	⋮

Policy Gradient

Given actor parameter θ

τ^1: (s_1^1, a_1^1) $R(\tau^1)$

(s_2^1, a_2^1) $R(\tau^1)$

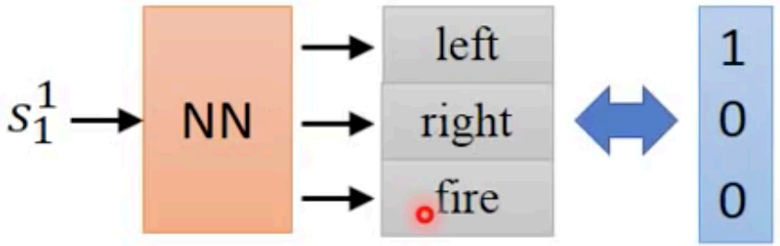
τ^2: (s_1^2, a_1^2) $R(\tau^2)$

(s_2^2, a_2^2) $R(\tau^2)$

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \blacksquare \nabla log p(a_t^n | s_t^n, \theta)$$

$a_1^1 = left$



Policy Gradient

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \blacksquare \nabla log p(a_t^n | s_t^n, \theta)$$

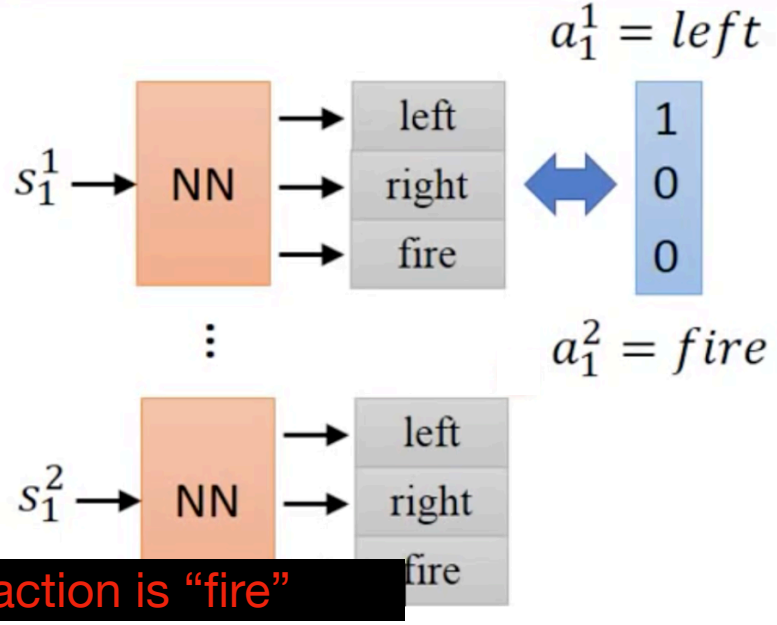
Given actor parameter θ

τ^1: (s_1^1, a_1^1) $R(\tau^1)$

(s_2^1, a_2^1) $R(\tau^1)$

τ^2: (s_1^2, a_1^2) $R(\tau^2)$

(s_2^2, a_2^2) $R(\tau^2)$



If our action is “fire”

Policy Gradient

Given actor parameter θ

τ^1: (s_1^1, a_1^1) $R(\tau^1)$

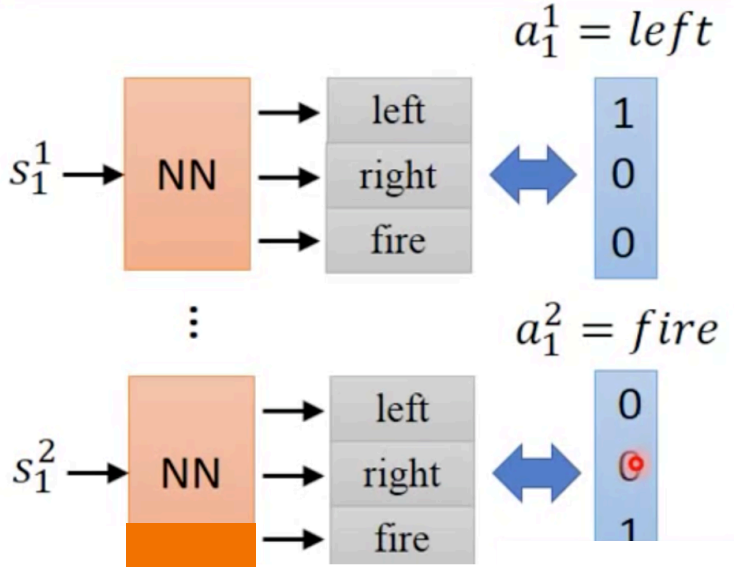
(s_2^1, a_2^1) $R(\tau^1)$

τ^2: (s_1^2, a_1^2) $R(\tau^2)$

(s_2^2, a_2^2) $R(\tau^2)$

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \blacksquare \nabla log p(a_t^n | s_t^n, \theta)$$



Policy Gradient

Given actor parameter θ

$$\tau^1: \quad (s_1^1, a_1^1) \quad R(\tau^1)$$
$$(s_2^1, a_2^1) \quad R(\tau^1)$$
$$\vdots \qquad \vdots$$
$$\tau^2: \quad (s_1^2, a_1^2) \quad R(\tau^2)$$
$$(s_2^2, a_2^2) \quad R(\tau^2)$$
$$\vdots \qquad \vdots$$

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta =$$

$$\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Policy Gradient

Given actor parameter θ

τ^1: (s_1^1, a_1^1) $R(\tau^1)$

(s_2^1, a_2^1) $R(\tau^1)$

⋮ ⋮

τ^2: (s_1^2, a_1^2) $R(\tau^2)$

(s_2^2, a_2^2) $R(\tau^2)$

⋮ ⋮

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta =$$

$$\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Each training data is weighted by $R(\tau^n)$

Policy Gradient

Given actor parameter θ

τ^1: (s_1^1, a_1^1) $R(\tau^1)$ **2**

(s_2^1, a_2^1) $R(\tau^1)$ **2**

τ^2: (s_1^2, a_1^2) $R(\tau^2)$ **1**

(s_2^2, a_2^2) $R(\tau^2)$ **1**

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Each training data is weighted by $R(\tau^n)$

Policy Gradient

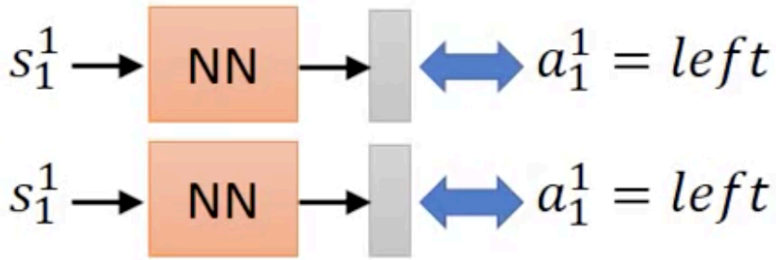
$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Given actor parameter θ

τ^1:	(s_1^1, a_1^1)	$R(\tau^1)$	2
	(s_2^1, a_2^1)	$R(\tau^1)$	2
	⋮	⋮	
τ^2:	(s_1^2, a_1^2)	$R(\tau^2)$	1
	(s_2^2, a_2^2)	$R(\tau^2)$	1
	⋮	⋮	

Each training data is weighted by $R(\tau^n)$



Policy Gradient

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Given actor parameter θ

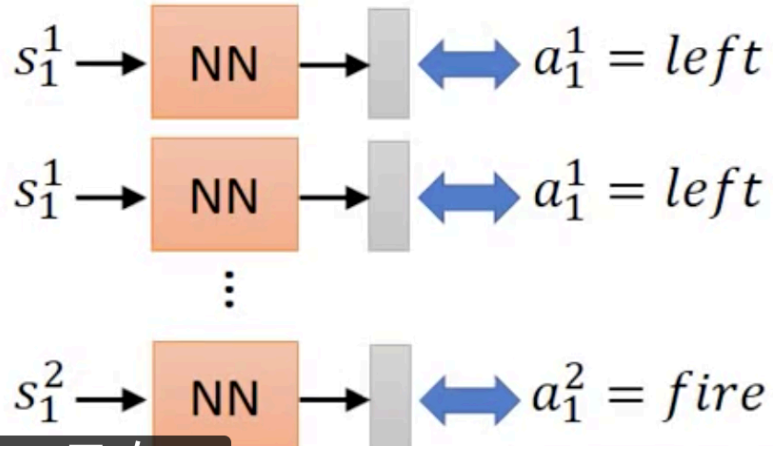
τ^1: (s_1^1, a_1^1) $R(\tau^1)$ 2

(s_2^1, a_2^1) $R(\tau^1)$ 2

τ^2: (s_1^2, a_1^2) $R(\tau^2)$ 1

(s_2^2, a_2^2) $R(\tau^2)$ 1

Each training data is weighted by $R(\tau^n)$



1. Use the current neural network (actor) to play the game, to get data from many episodes.

2. As a result, we get many triplets (observation, action, reward).

3. Turn the reinforcement learning problem into a classification problem. Train the network.

4. Use the new network to collect more data. Use the new data to train the new network.

5. Repeat the above steps, until the network's performance converges.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} (R(\tau^n) - b) \nabla log p(a_t^n | s_t^n, \theta)$$

Add a Baseline

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_{\theta} \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Add a Baseline

It is possible that $R(\tau^n)$ is always positive.

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

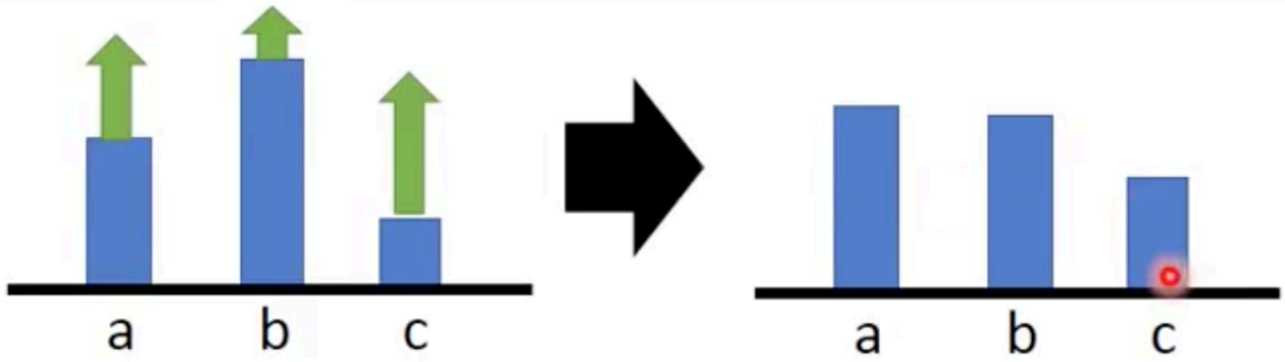
Add a Baseline

It is possible that $R(\tau^n)$ is always positive.

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Ideal case



All three probabilities increase

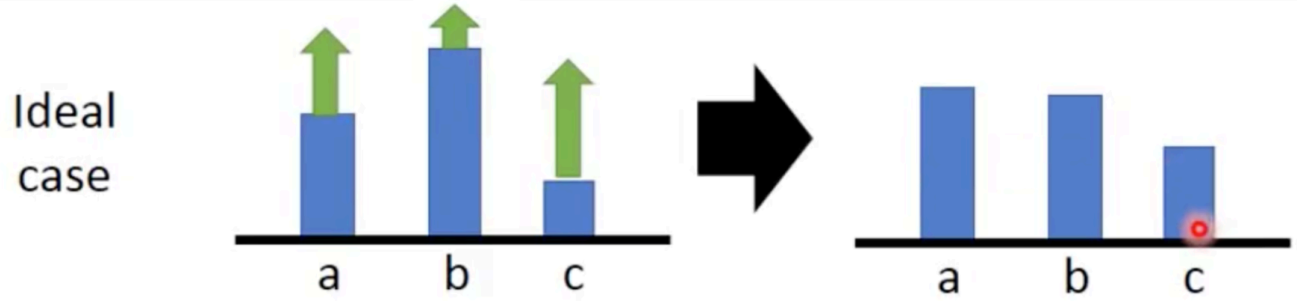
Since they form a probability distribution, we need to normalize them (to sum to 1).

Add a Baseline

It is possible that $R(\tau^n)$ is always positive.

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$



All three probabilities increase

Since they form a probability distribution, we need to normalize them (to sum to 1).

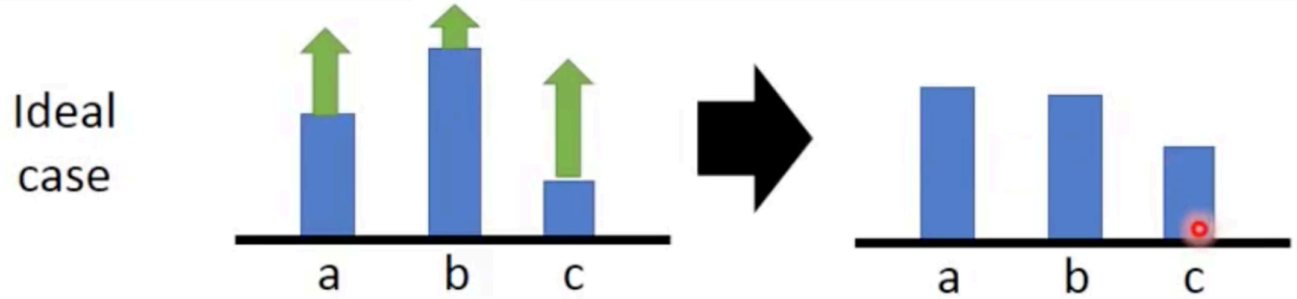
But in reality, we only sample (limited) trajectories. It is possible that an action of a small probability did not get sampled, and after normalization, its probability becomes even smaller (and thus even harder to be sampled). So we would like to avoid normalization (and actually reduce its impact at least). For that, we would like the reward to be sometimes positive and sometimes negative (more balanced).

Add a Baseline

It is possible that $R(\tau^n)$ is always positive.

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$



All three probabilities increase

Since they form a probability distribution, we need to normalize them (to sum to 1).

Solution: reduce the reward by a bias value:

$$R(\tau^n) - b$$

So that the reward $R(\tau^n) - b$ is sometimes positive, sometimes negative.

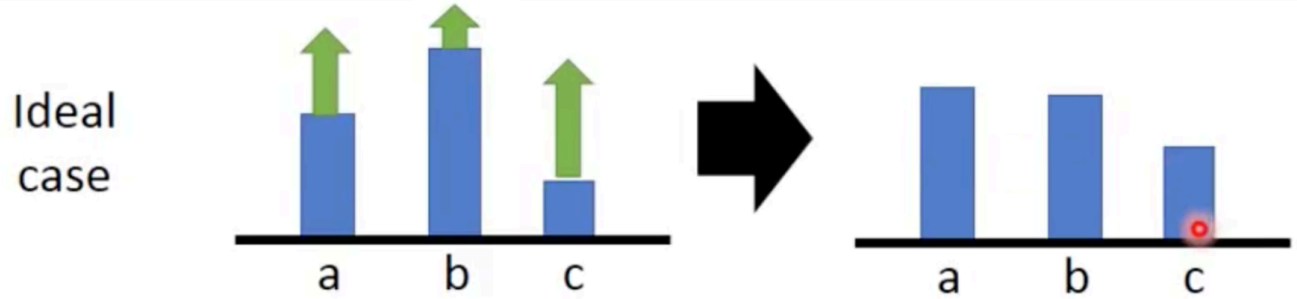
(You decide how to select the bias b.)

Add a Baseline

It is possible that $R(\tau^n)$ is always positive.

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$



All three probabilities increase

Since they form a probability distribution, we need to normalize them (to sum to 1).

When $R(\tau^n) - b$ is positive:

the coresponding action probabity is increased

When $R(\tau^n) - b$ is negative:

the coresponding action probabity is decreased

So that the reward $R(\tau^n) - b$ is sometimes positive, sometimes negative.

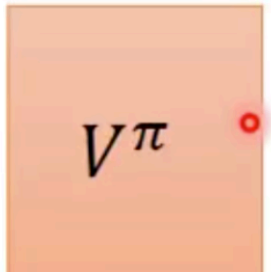
Critic

- A critic does not determine the action.
- Given an actor, it evaluates the how good the actor is

An actor can be found from a critic.

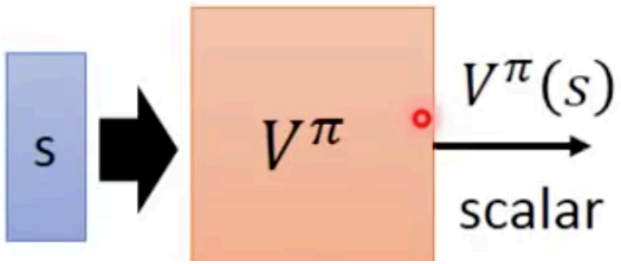
Three kinds of Critics

- A critic is a function depending on the actor π it is evaluated
  - The function is represented by a neural network
- State value function $V^{\pi}(s)$
  - When using actor π, the *cumulated* reward expects to be obtained after seeing observation (state) s



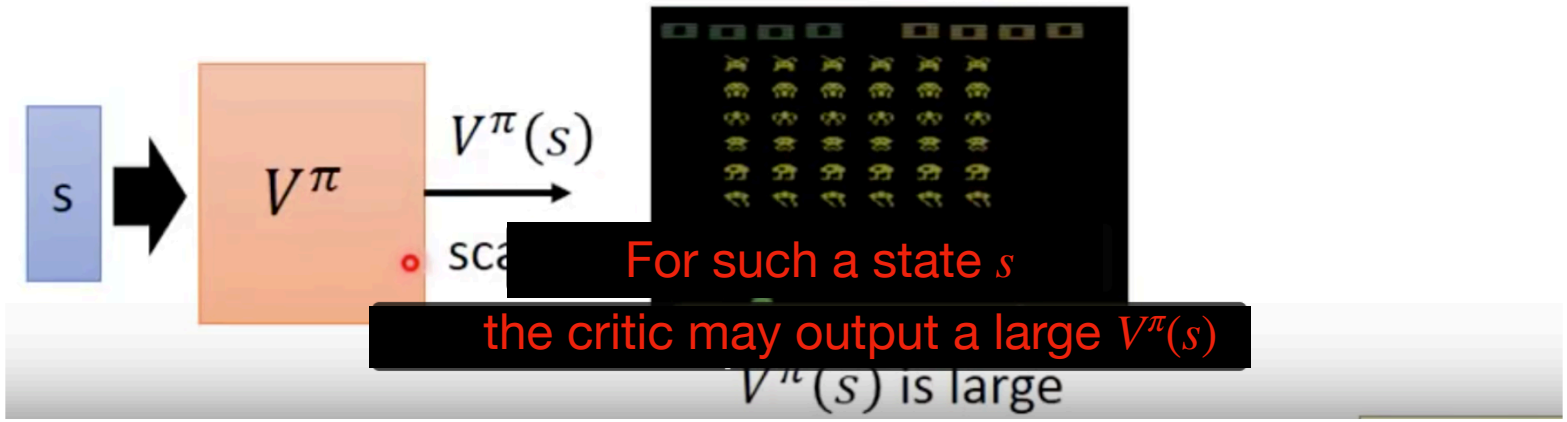
Three kinds of Critics

- A critic is a function depending on the actor π it is evaluated
  - The function is represented by a neural network
- State value function $V^{\pi}(s)$
  - When using actor π, the *cumulated* reward expects to be obtained after seeing observation (state) s



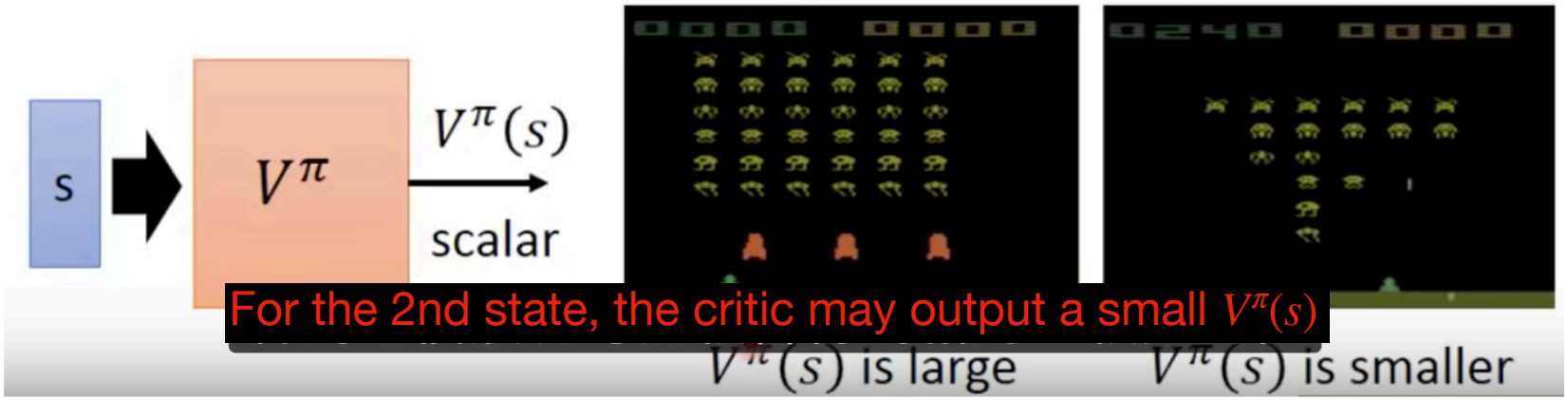
Three kinds of Critics

- A critic is a function depending on the actor π it is evaluated
  - The function is represented by a neural network
- State value function $V^{\pi}(s)$
  - When using actor π, the *cumulated* reward expects to be obtained after seeing observation (state) s



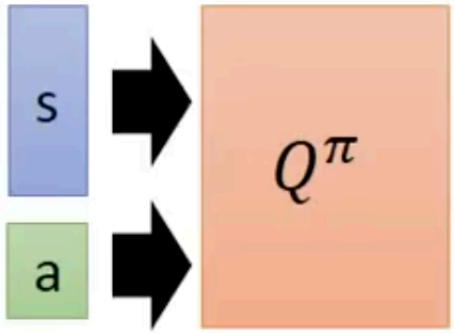
Three kinds of Critics

- A critic is a function depending on the actor π it is evaluated
  - The function is represented by a neural network
- State value function $V^{\pi}(s)$
  - When using actor π, the *cumulated* reward expects to be obtained after seeing observation (state) s



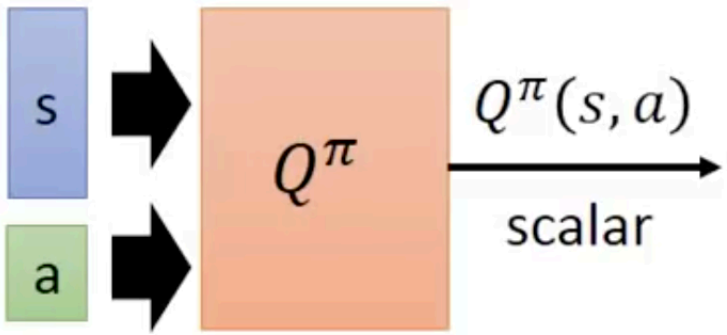
Three kinds of Critics

- State-action value function $Q^{\pi}(s, a)$
  - When using actor π, the *cumulated* reward expects to be obtained after seeing observation s and taking a



Three kinds of Critics

- State-action value function $Q^{\pi}(s, a)$
  - When using actor π, the *cumulated* reward expects to be obtained after seeing observation s and taking a

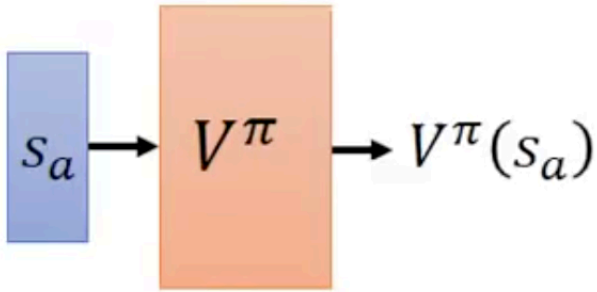


How to estimate $V^{\pi}(s)$

- Monte-Carlo based approach
  - The critic watches π playing the game

After seeing s_a,

Until the end of the episode, the cumulated reward is G_a

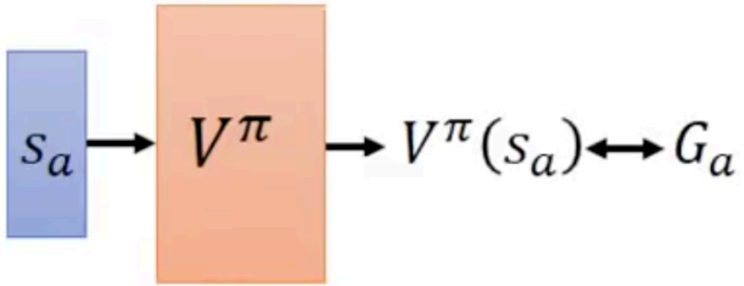


How to estimate $V^{\pi}(s)$

- Monte-Carlo based approach
  - The critic watches π playing the game

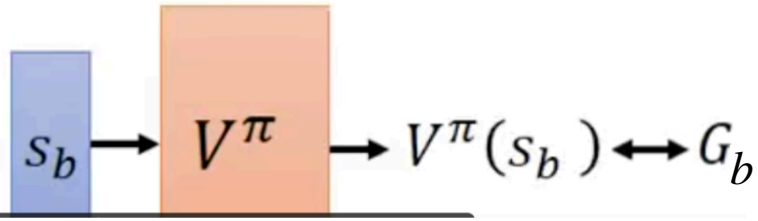
After seeing s_a,

Until the end of the episode, the cumulated reward is G_a



After seeing s_b,

Until the end of the episode, the cumulated reward is G_b

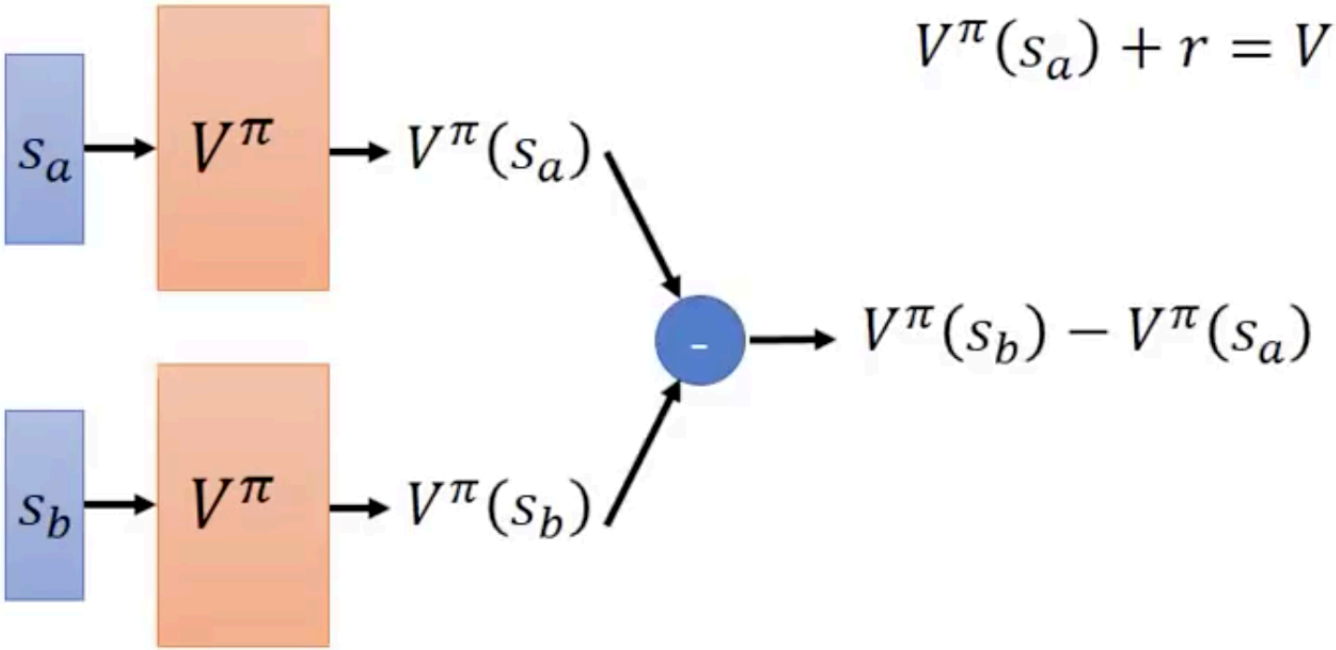


How to estimate $V^{\pi}(s)$

- Temporal-difference approach

$$\cdots s_a, a, r, s_b \cdots$$

$$V^{\pi}(s_a) + r = V^{\pi}(s_b)$$



Actor-Critic

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla log p(a_t^n | s_t^n, \theta)$$

Evaluated by critic

Advantage Function: $r_t^n - \left(V^{\pi_\theta}(s_t^n) - V^{\pi_\theta}(s_{t+1}^n)\right)$

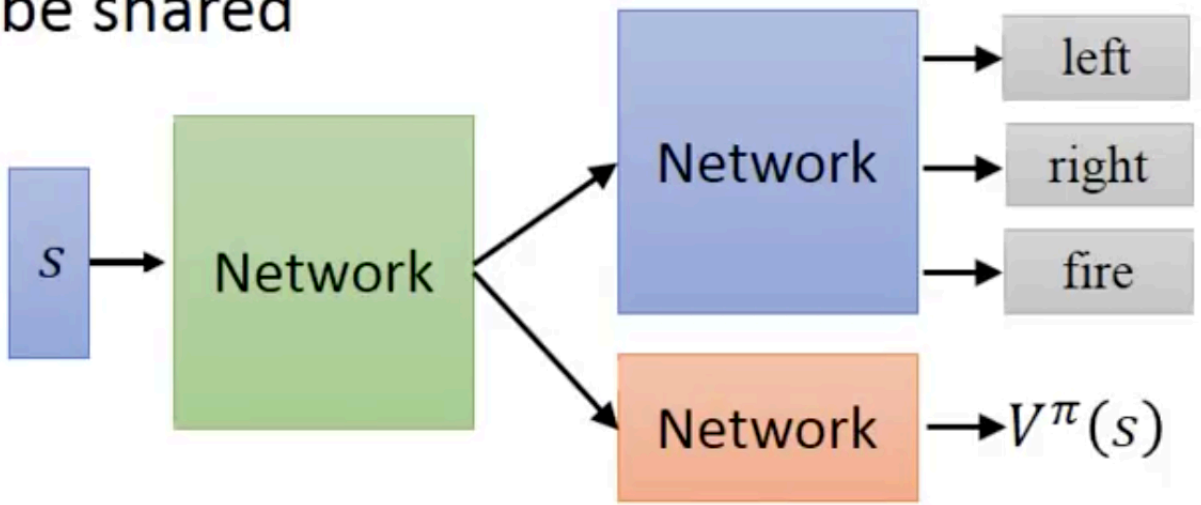
Baseline is added

The reward r_t^n we truly obtain when taking action a_t^n

Expected reward r_t^n we obtain if we use actor π_θ

Actor-Critic

- Tips
  - The parameters of actor $\pi(s)$ and critic $V^{\pi}(s)$ can be shared



  - Use output entropy as regularization for $\pi(s)$

Asynchronous

Source of image:
https://medium.com/emergent-future/simple-reinforcement-learning-with-tensorflow-part-8-asynchronous-actor-critic-agents-a3c-c88f72a5e9f2#.68x6na7o9

1. Copy global parameters