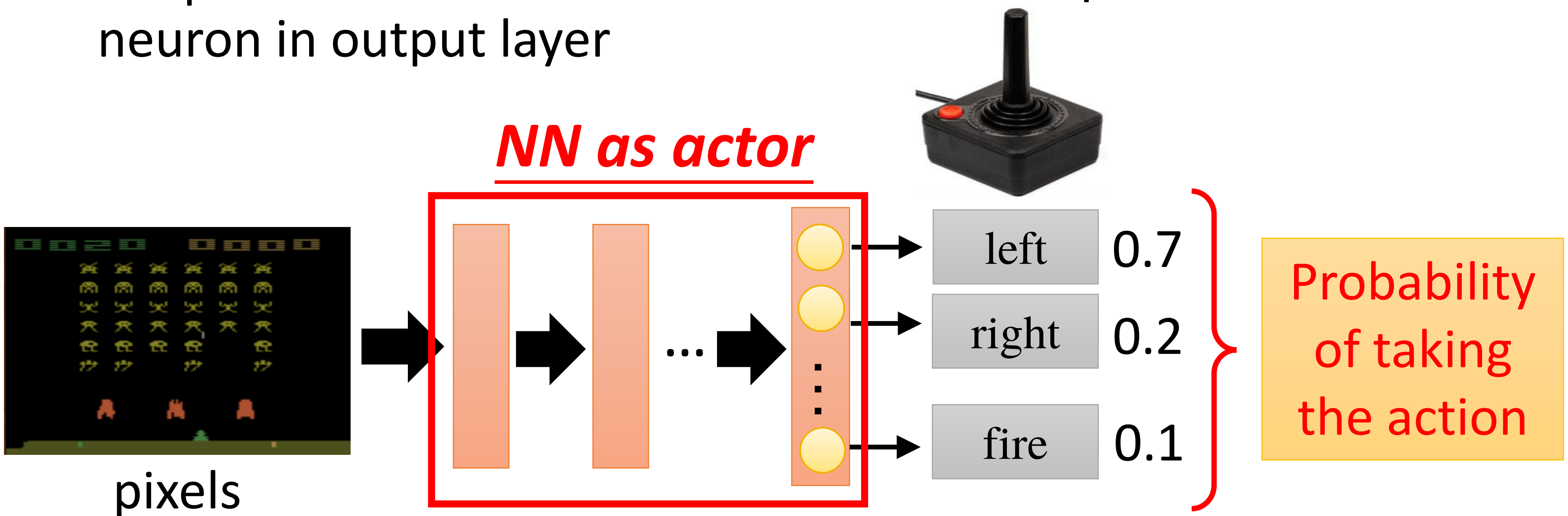
# Asynchronous Advantage Actor-Critic (A3C)

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, Koray Kavukcuoglu, “Asynchronous Methods for Deep Reinforcement Learning”, ICML, 2016

# Actor is a Neural network

- 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




Sample an action based on the probability or argmax

Actor can also have continuous action.

# Actor – Goodness of an Actor

- Given an actor $\pi(s)$ with network parameter $\theta^{\pi}$
- Use the actor $\pi(s)$ 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: $R = \sum_{t=1}^{T} r_t$

Even with the same actor, $R$ is different each time

Randomness in the actor and the game

We define $\bar{R}_{\theta^{\pi}}$ as the *expected total reward*

$\bar{R}_{\theta^{\pi}}$ evaluates the goodness of an actor $\pi(s)$

# Actor – Policy Gradient

$\theta^{\pi'} \leftarrow \theta^{\pi} + \eta \nabla \bar{R}_{\theta^\pi}$ Using $\theta^\pi$ to obtain $\{\tau^1, \tau^2, \cdots, \tau^N\}$

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

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

What if we replace $R(\tau^n)$ with $r_t^n$ ……

---

If in $\tau^n$ machine takes $a_t^n$ when seeing $s_t^n$

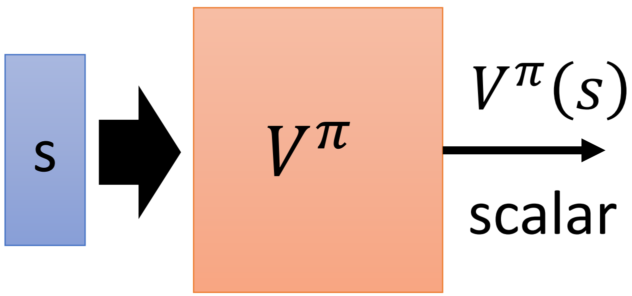
$R(\tau^n)$ is positive ⇒ Tuning $\theta$ to increase $p(a_t^n|s_t^n)$

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

It is very important to consider the cumulative reward $R(\tau^n)$ of the whole trajectory $\tau^n$ instead of immediate reward $r_t^n$

# Critic

- A critic does not determine the action.
- Given an actor π, it evaluates the how good the actor is
- State value function $V^{\pi}(s)$
  - When using actor $\pi$, the *cumulated* reward expects to be obtained after seeing observation (state) s




$V^{\pi}(s)$ is large

$V^{\pi}(s)$ is smaller

# Critic

$V^{以前的阿光}(大馬步飛) = \text{bad}$

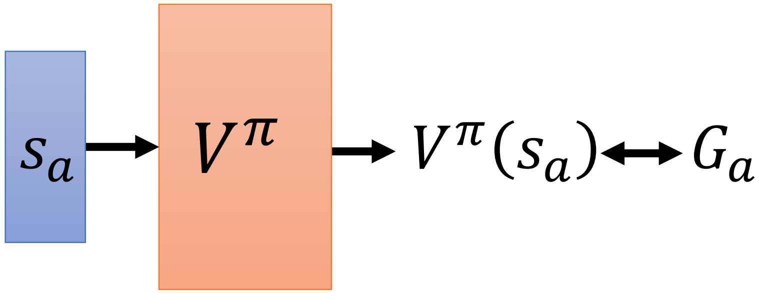
$V^{變強的阿光}(大馬步飛) = \text{good}$


# How to estimate $V^\pi(s)$

- Monte-Carlo based approach
  - The critic watches $\pi$ playing the game

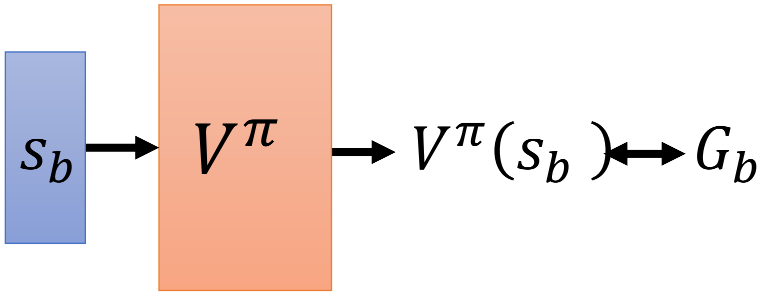
After seeing $s_a$,

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



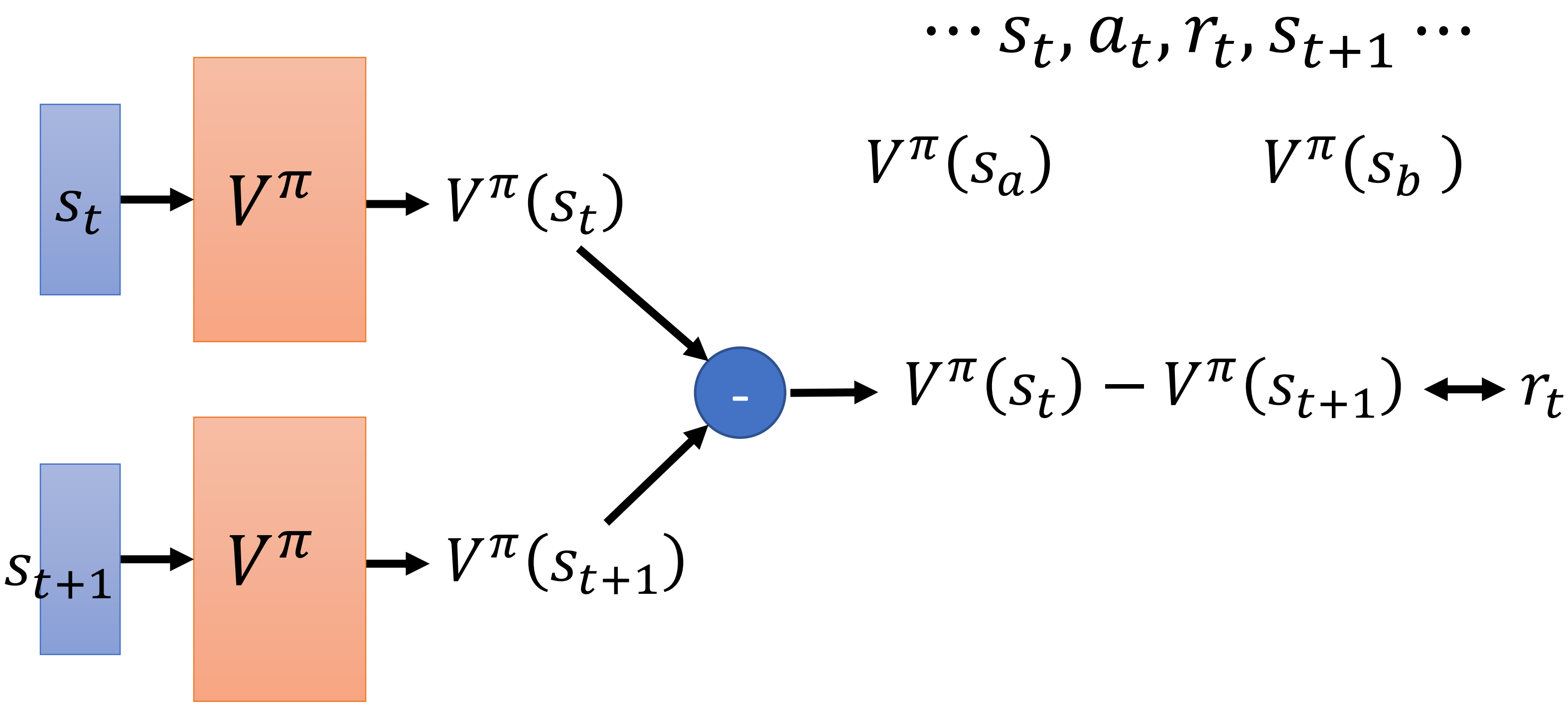

After seeing $s_b$,

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



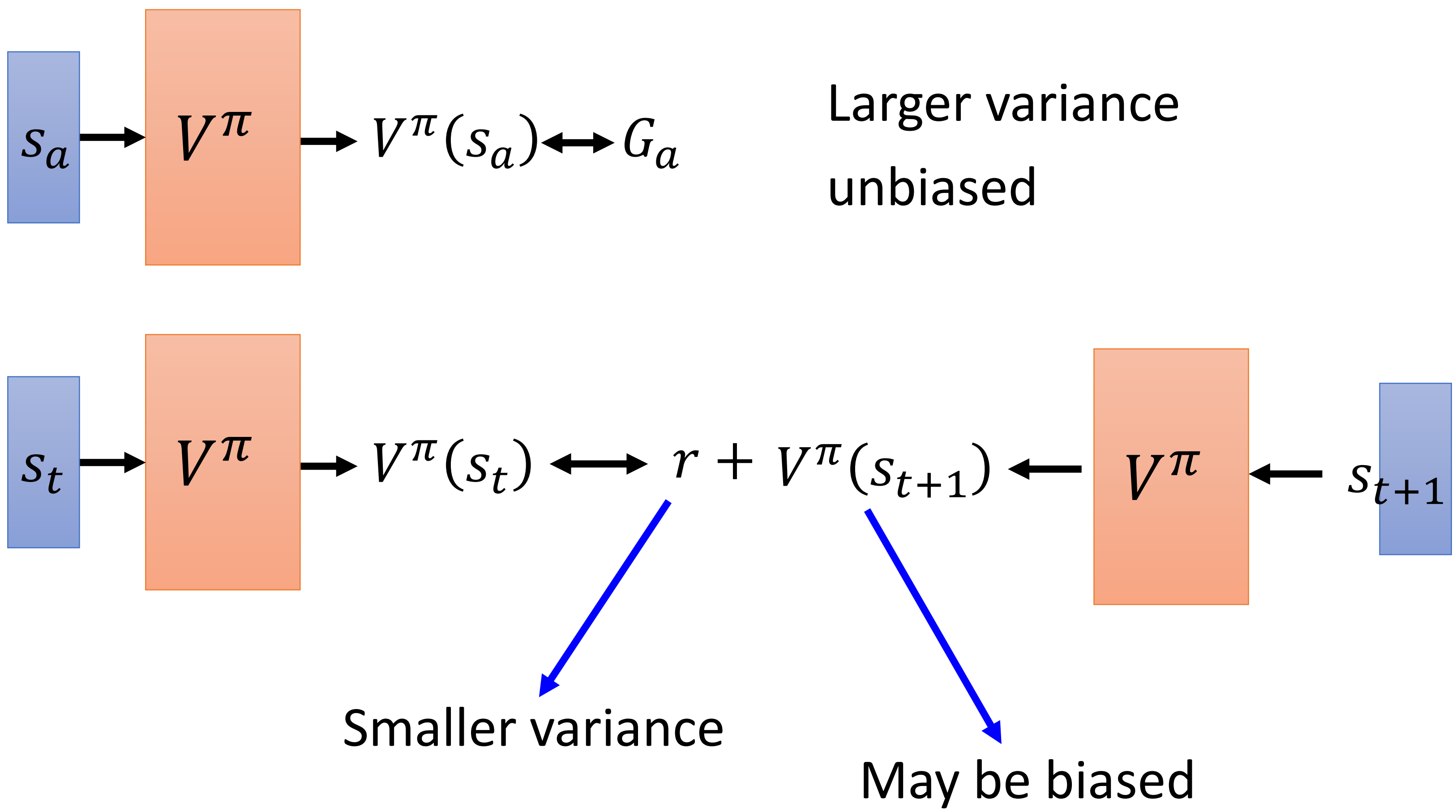
# How to estimate $V^{\pi}(s)$

- Temporal-difference approach




Some applications have very long episodes, so that delaying all learning until an episode's end is too slow.

# MC v.s. TD


$s_a$
$V^\pi$
$V^\pi(s_a) \leftrightarrow G_a$
Larger variance
unbiased
$s_t$
$V^\pi$
$V^\pi(s_t) \leftrightarrow r + V^\pi(s_{t+1})$
$V^\pi$
$s_{t+1}$
Smaller variance
May be biased

# MC v.s. TD

[Sutton, v2, Example 6.4]

- The critic has the following 8 episodes
  - $s_a, r = 0, s_b, r = 0$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 0$, END

$$V^\pi(s_b) = 3/4$$

$$V^\pi(s_a) =?$$ 0? 3/4?

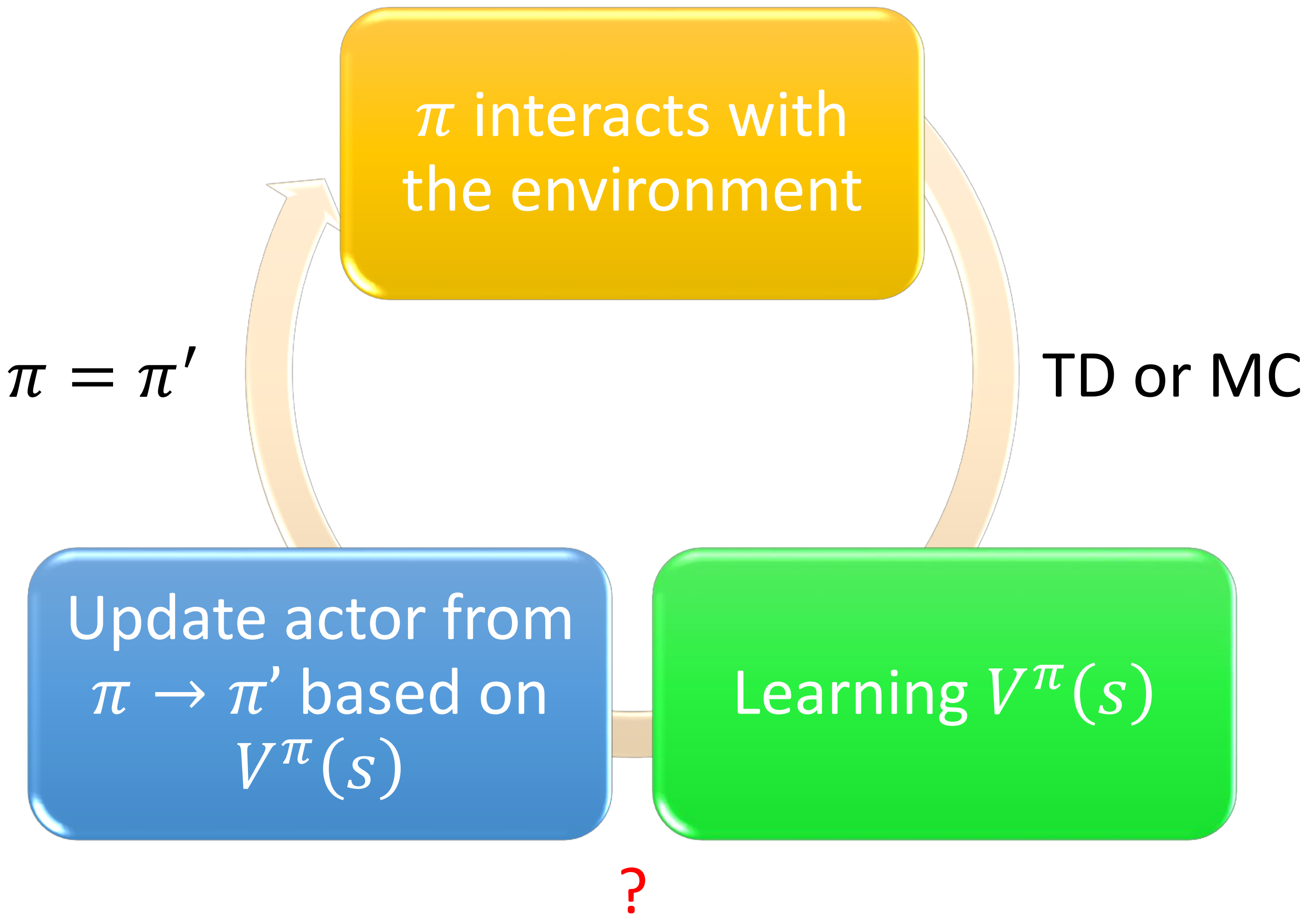
Monte-Carlo: $V^\pi(s_a) = 0$

Temporal-difference:

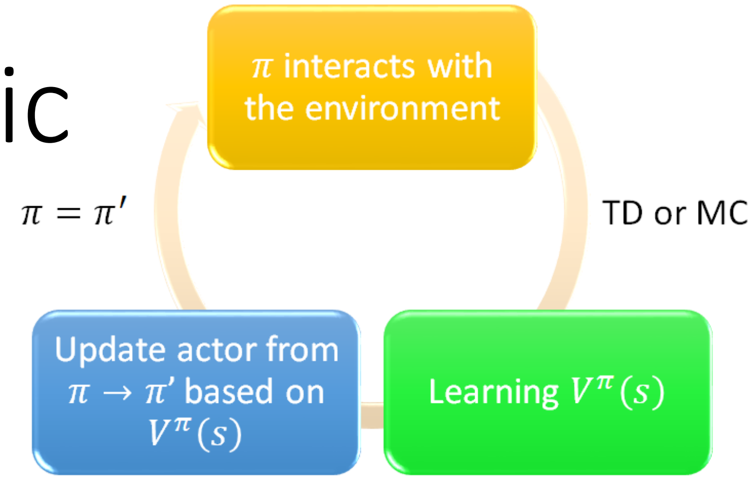
$$\underbrace{V^\pi(s_a)}_{3/4} + \underbrace{r}_{0} = \underbrace{V^\pi(s_b)}_{3/4}$$

(The actions are ignored here.)

# Actor-Critic



# Advantage Actor-Critic




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

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

Evaluated by critic

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

Baseline is added

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

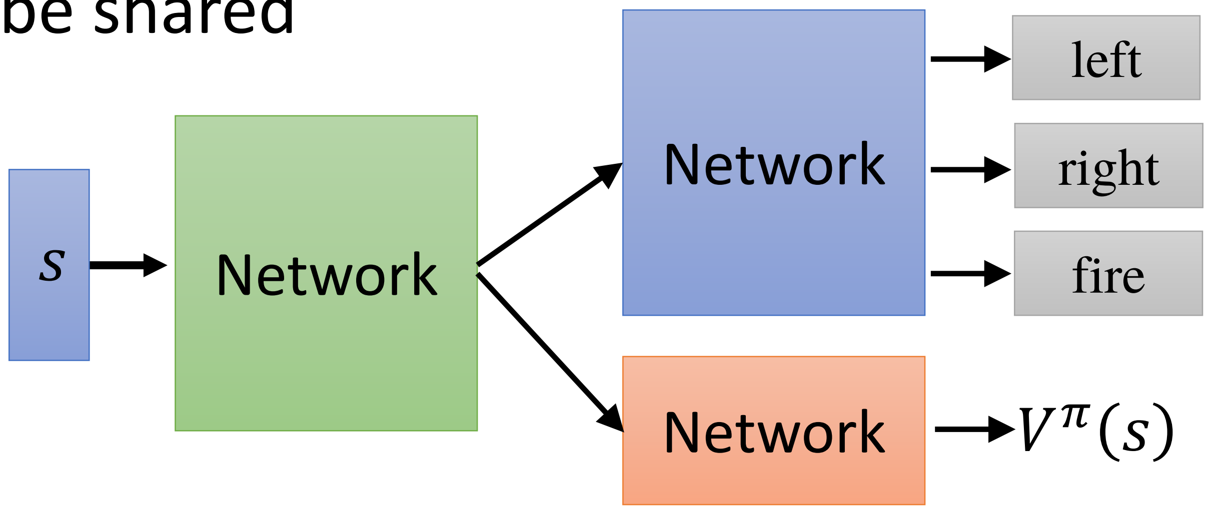
Expected reward $r_t^n$ we obtain if we use actor $\pi$

Positive advantage function $\Leftrightarrow$ Increasing the prob. of action $a_t^n$

Negative advantage function $\Leftrightarrow$ decreasing the prob. of action $a_t^n$

# Advantage Actor-Critic

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

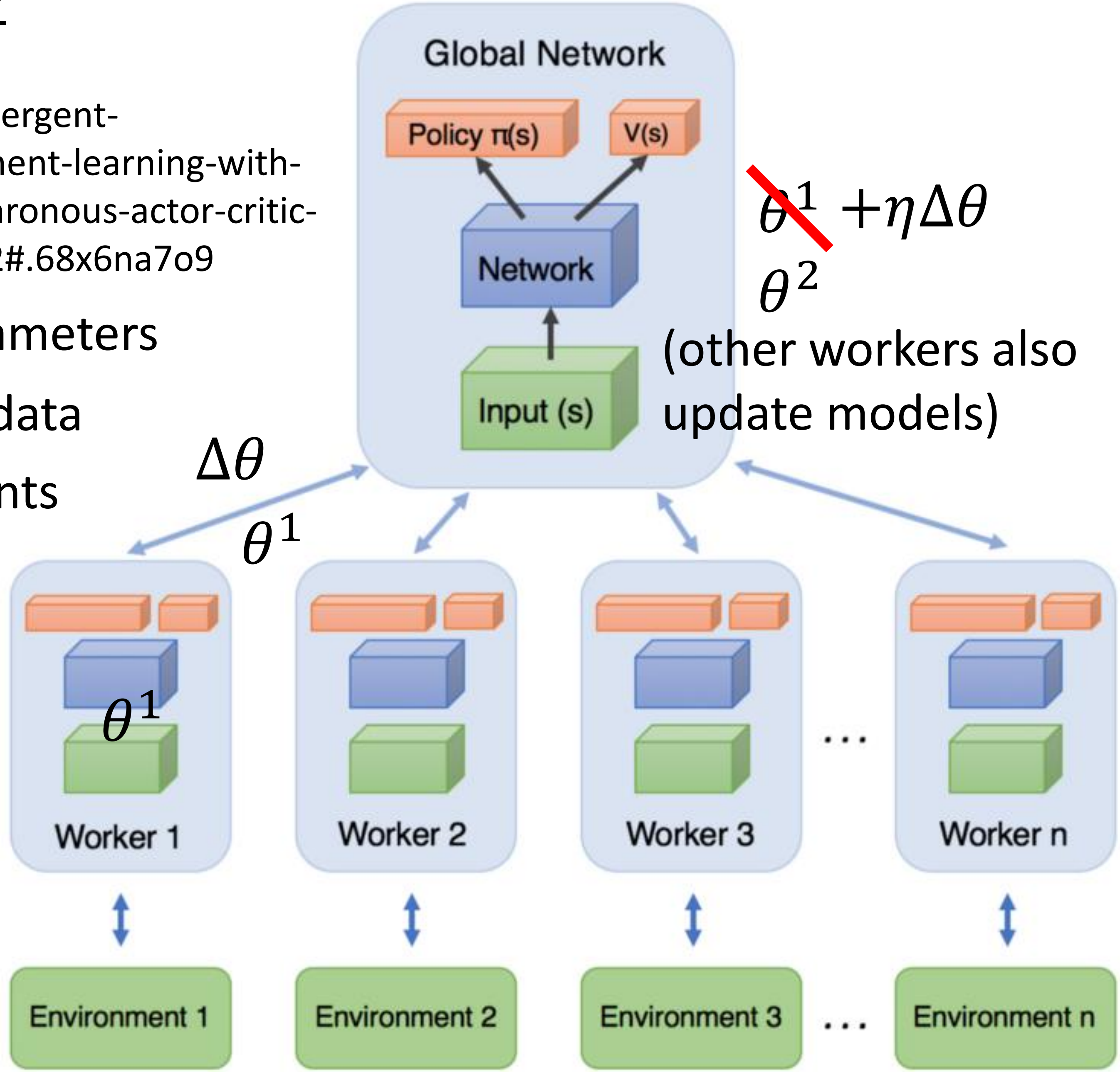



  - Use output entropy as regularization for $\pi(s)$
    - Larger entropy is preferred → exploration

# ***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
2. Sampling some data
3. Compute gradients
4. Update global models




$\cancel{\theta^1} + \eta\Delta\theta$

$\theta^2$

(other workers also update models)

$\Delta\theta$

$\theta^1$

$\theta^1$

$\Delta\theta$