Introduction of Deep Reinforcement Learning (RL)

Hung-yi Lee

Supervised Learning → RL

Human label

Cat

Human label

"3-3"?
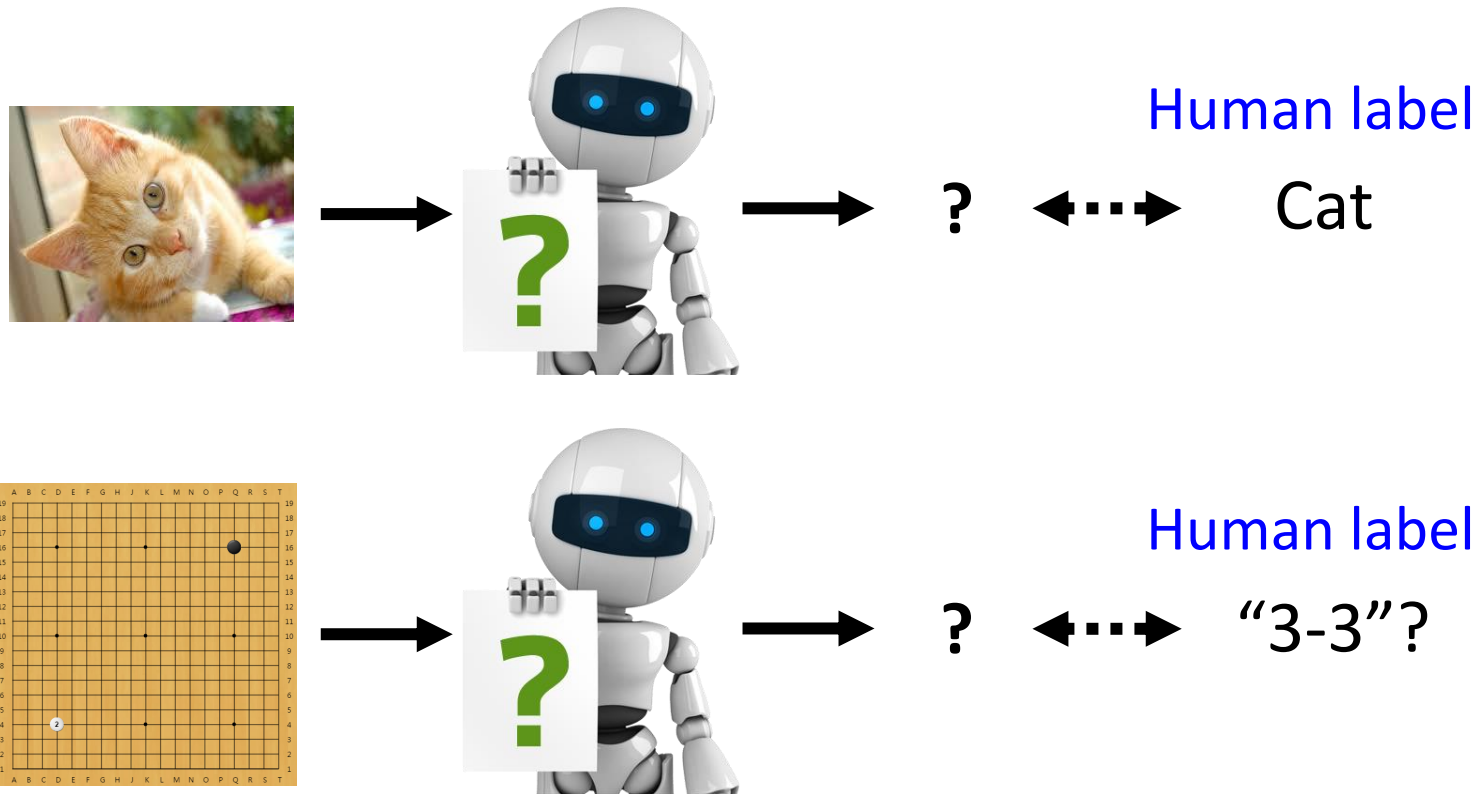
It is challenging to label data in some tasks.

WARNING: Don't use RL when you have labeled data. ☺

Outline
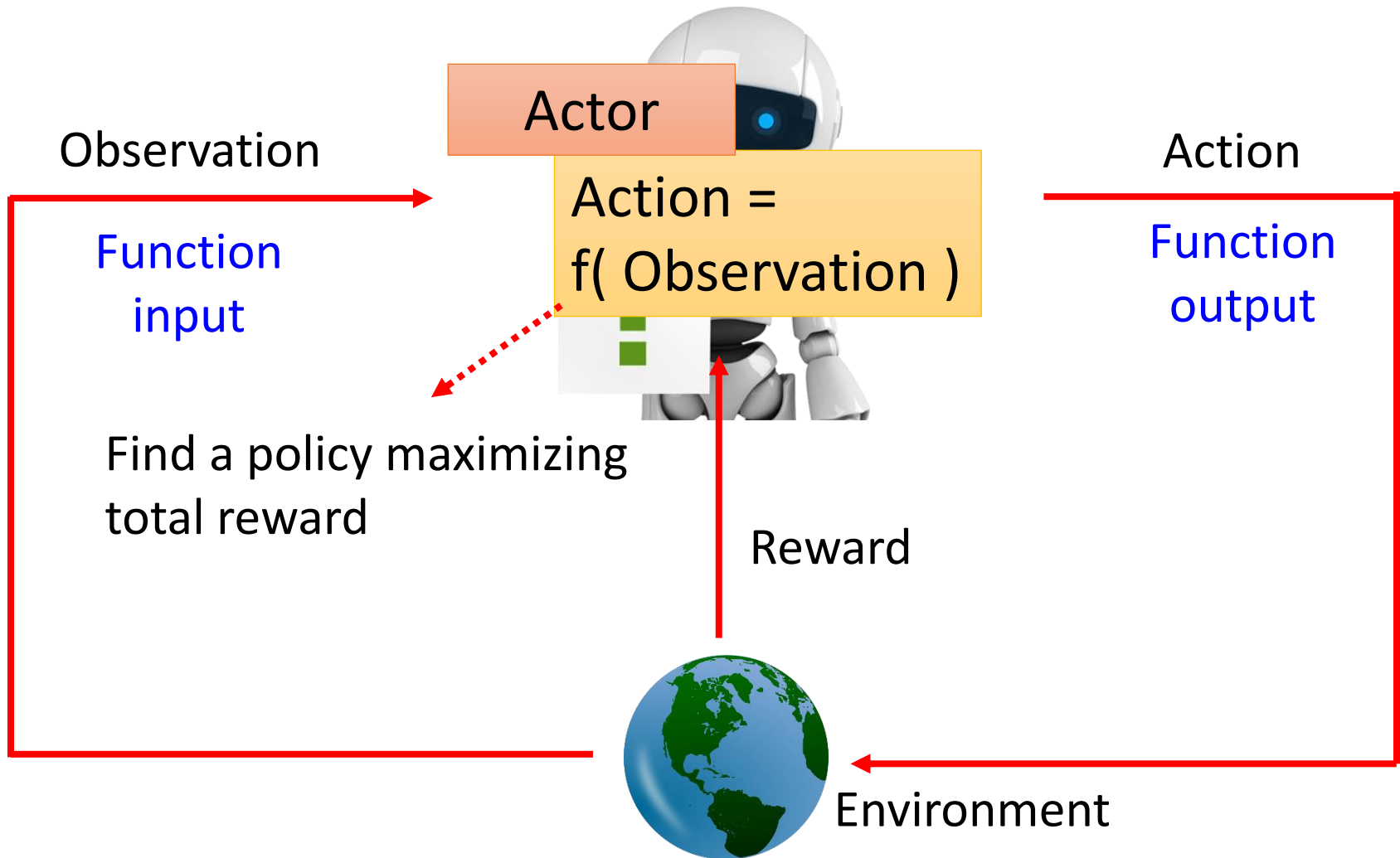
What is RL? (Three steps in ML)

Policy Gradient

Actor-Critic

Reward Shaping

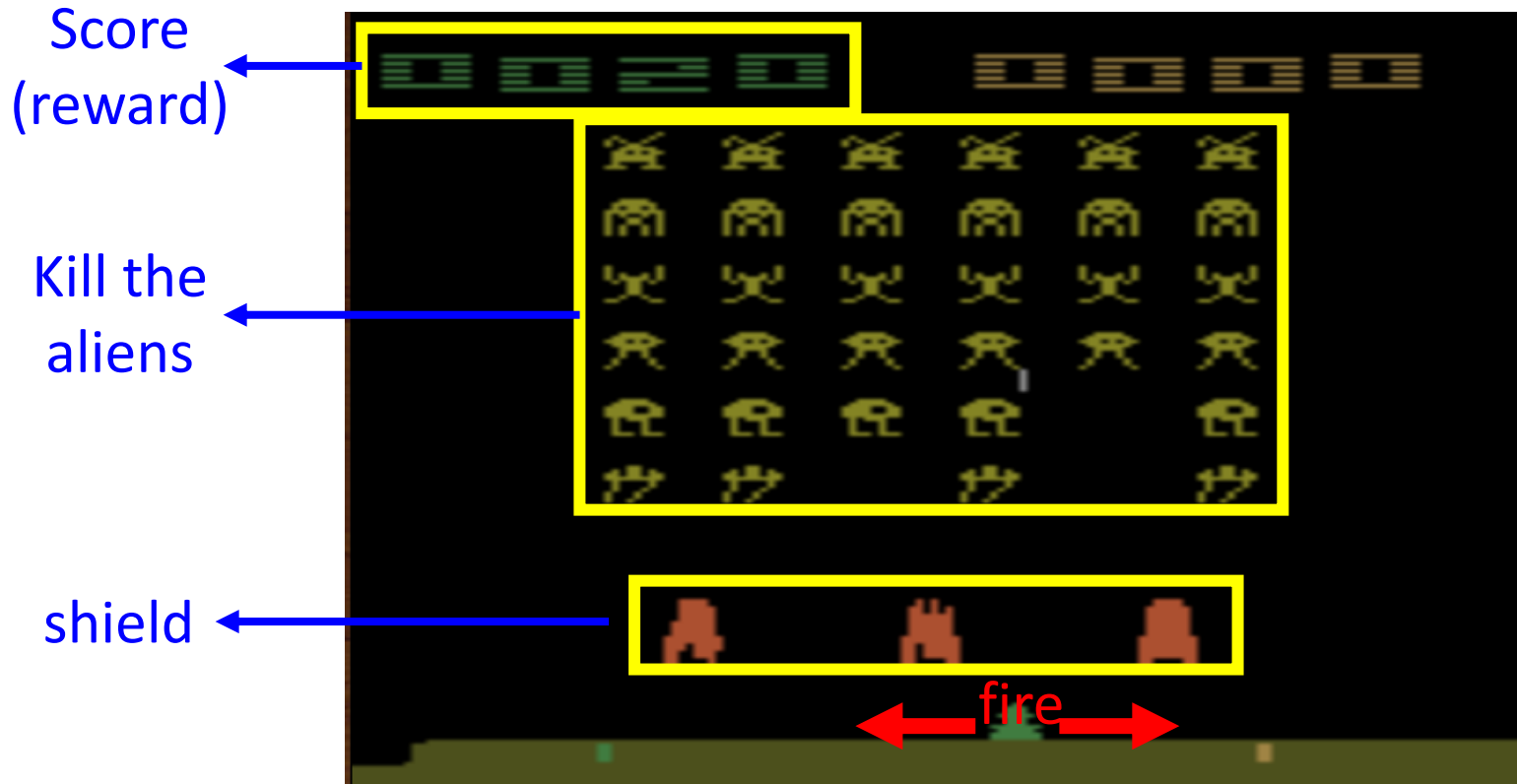No Reward: Learning from Demonstration

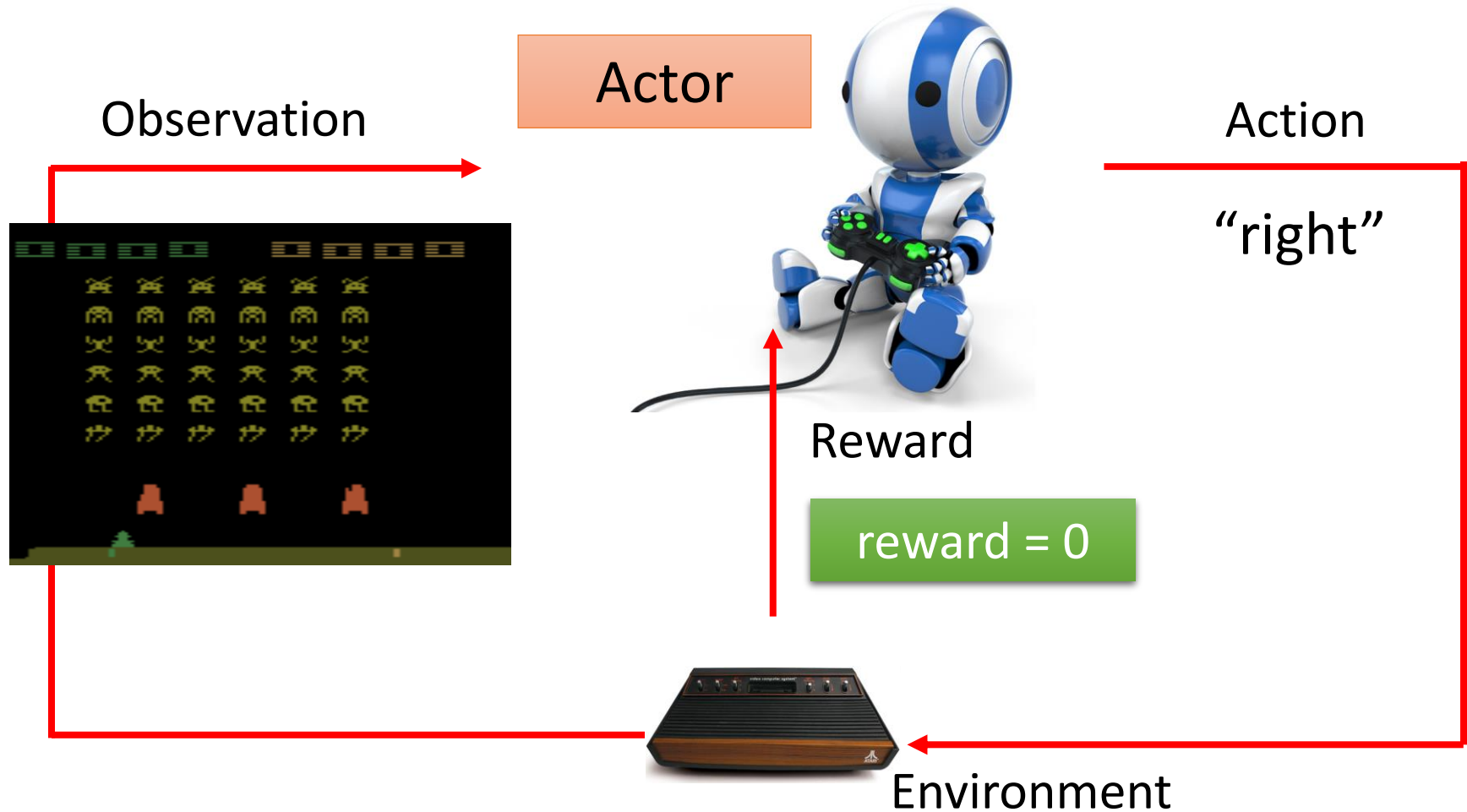# Machine Learning
≈ Looking for a Function

Observation

Actor

Action

Action =
f(Observation)

Function
input

Function
output

Find a policy maximizing
total reward

Reward

Environment

Example: Playing Video Game

- Space invader

Termination: all the aliens are killed, or your spaceship is destroyed.

Score (reward)

Kill the aliens

shield

fire

Example: Playing Video Game

Observation

Actor

Action

"right"

Reward

reward = 0

Environment

Example: Playing Video Game

Observation

Actor

Action

"fire"

Reward

reward = 5
if killing an alien.

Environment

Example: Learning to play Go

Observation

Action

AlphaGo

Reward

Next Move

Environment
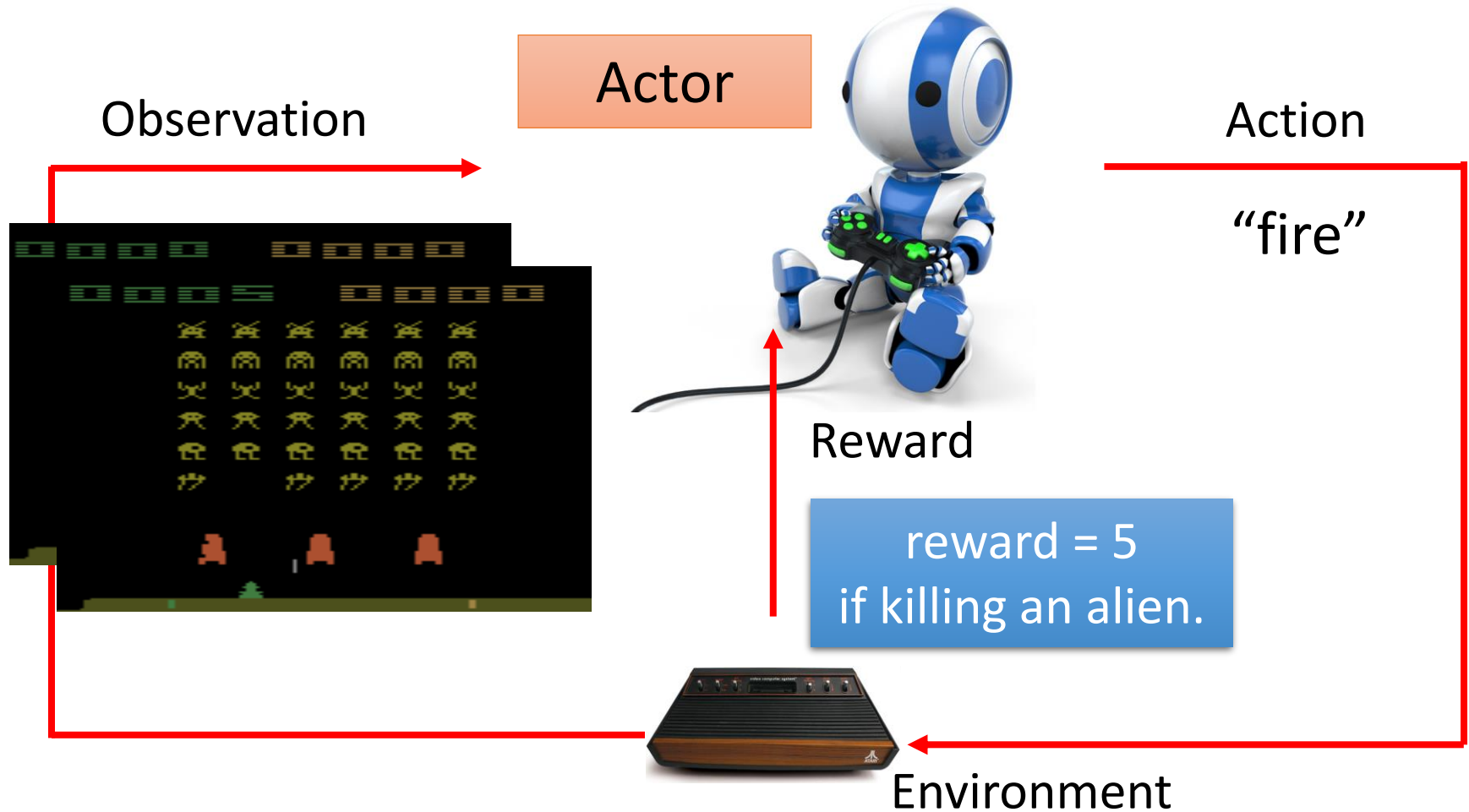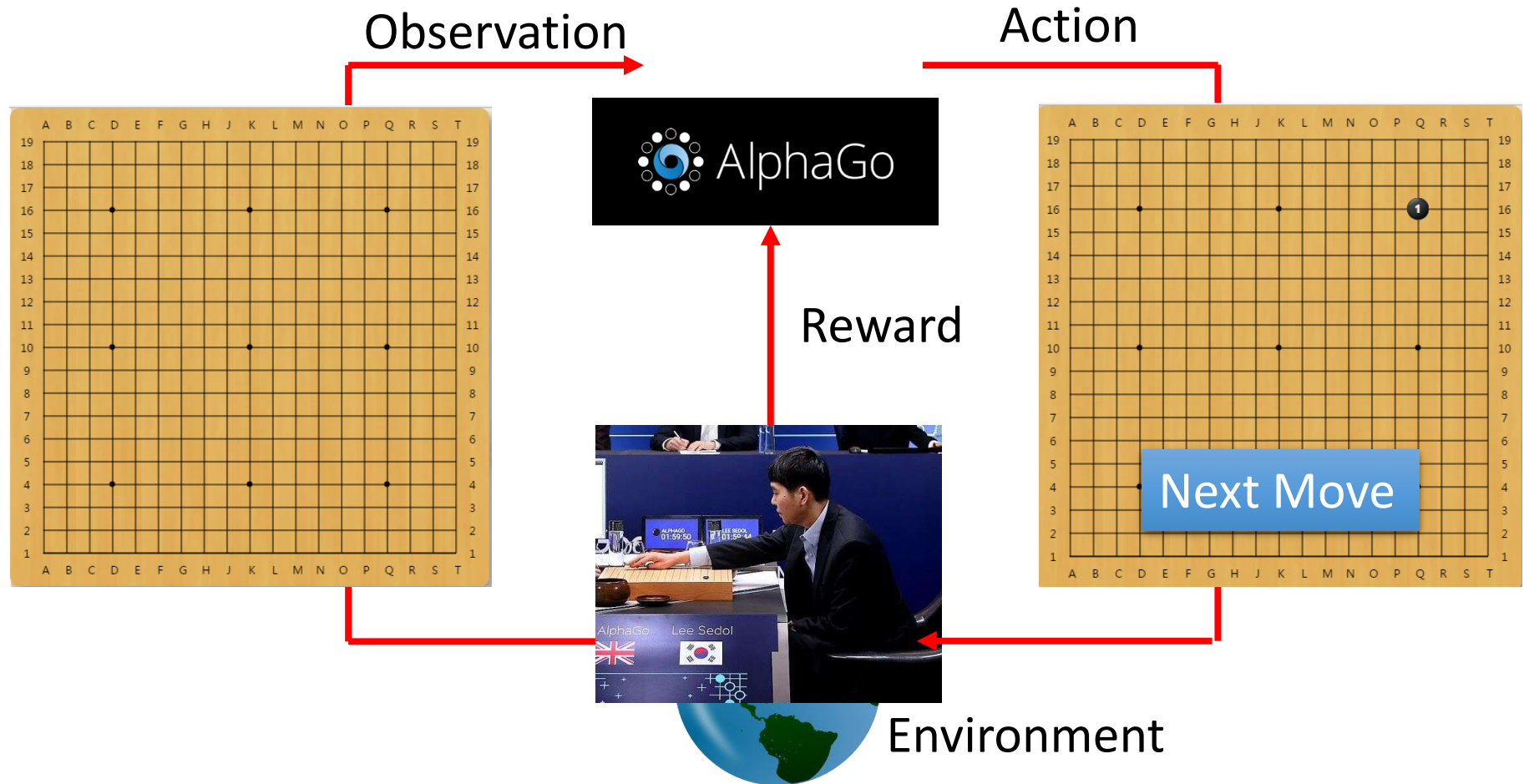
Example: Learning to play Go

Find an actor maximizing expected reward.

Observation

Action

AlphaGo

Reward

reward = 0 in most cases

If win, reward = 1

If loss, reward = -1

Environment
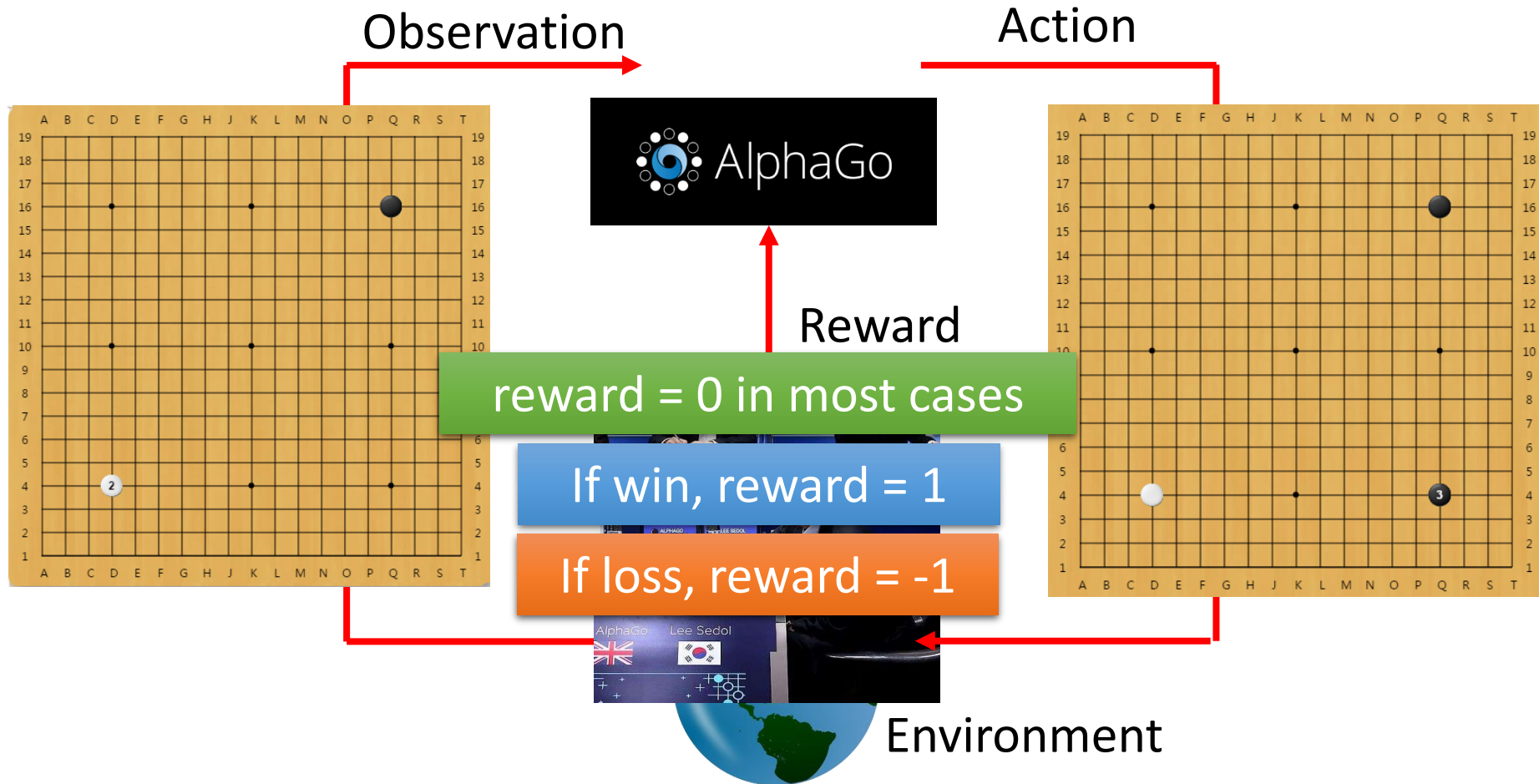
Machine Learning is so simple ……

Step 1: function with unknown ➜ Step 2: define loss from training data ➜ Step 3: optimization

Step 1: Function with Unknown

Policy Network (Actor)

Sample based on scores

left	0.7
right	0.2
fire	0.1

Score of an action

pixels
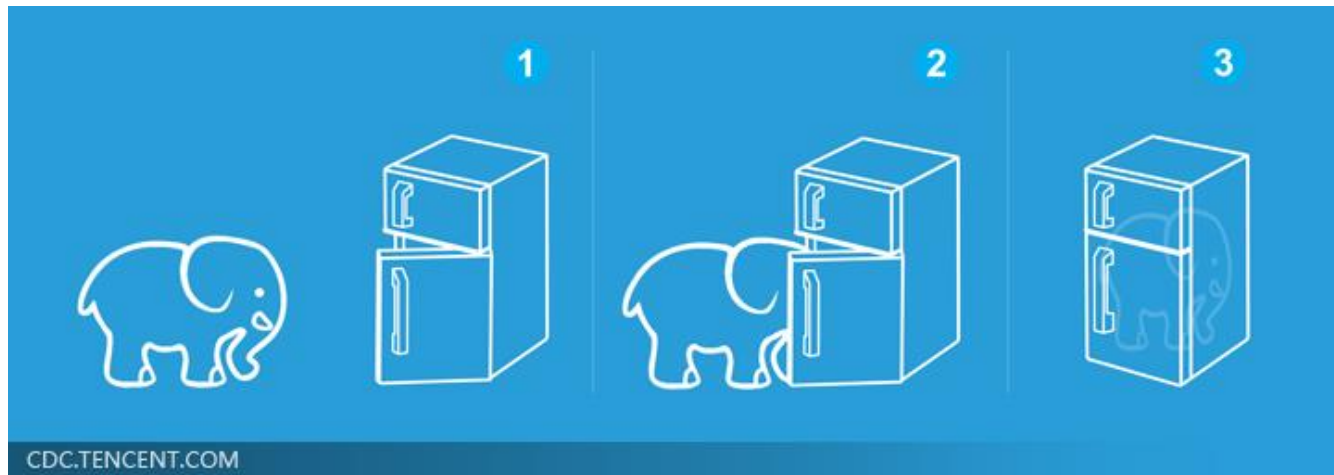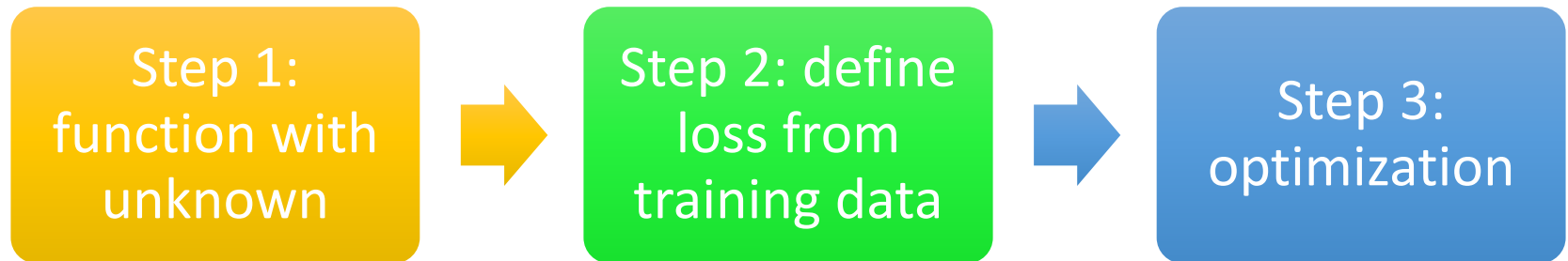
Considered as Classification!
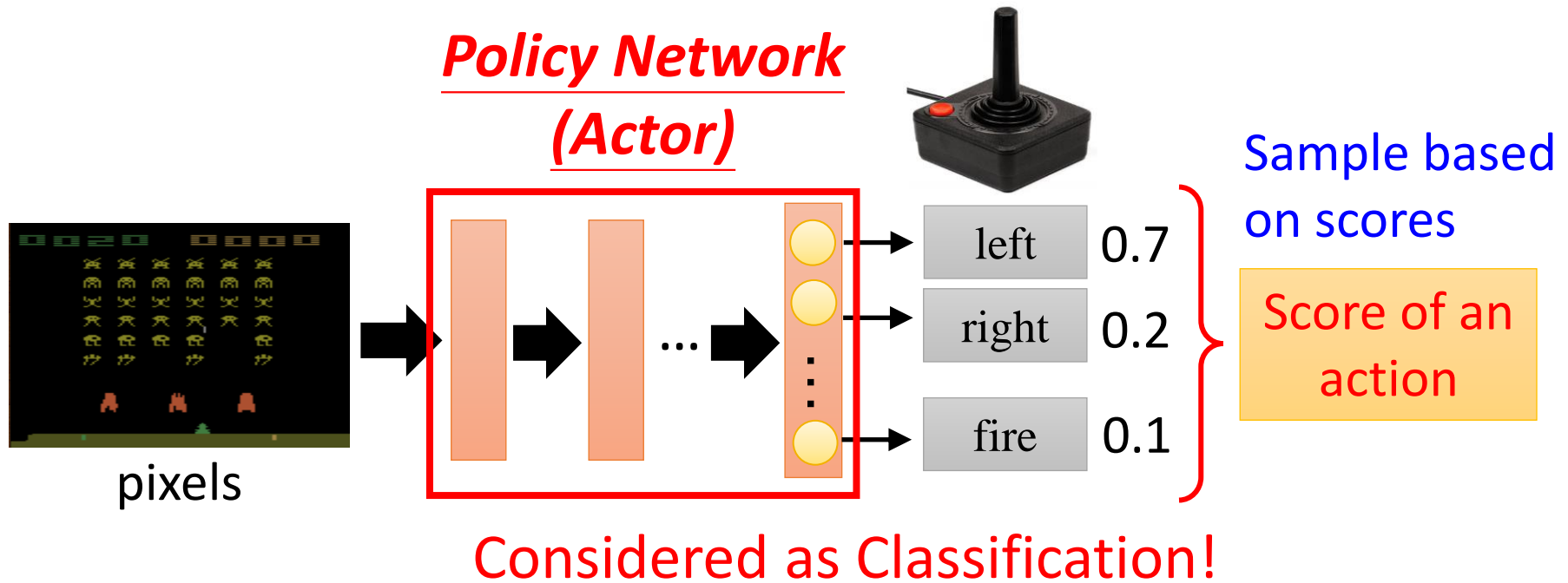
- Input of neural network: the observation of machine represented as a vector or a matrix

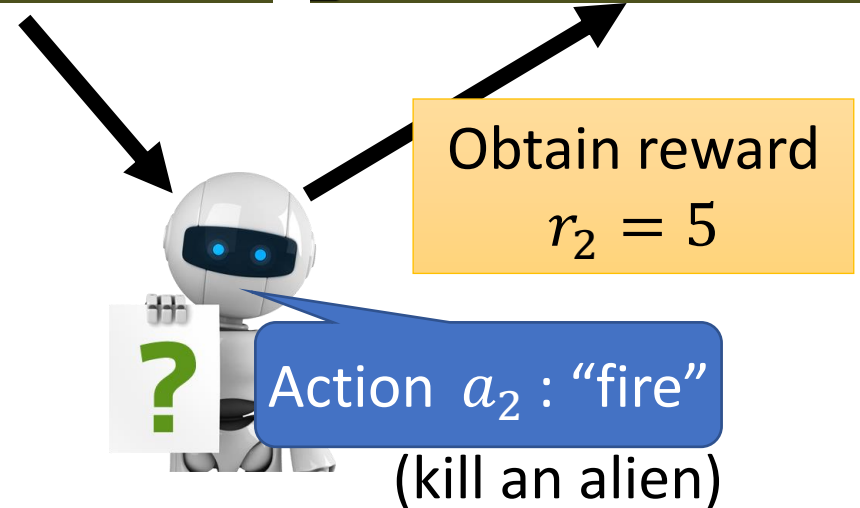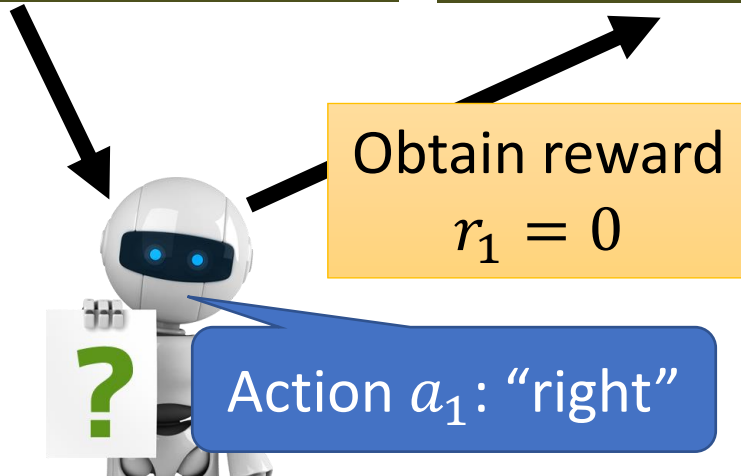- Output neural network : each action corresponds to a neuron in output layer

Step 2: Define "Loss"

Start with observation s_1

Observation s_2

Observation s_3

Obtain reward $r_1 = 0$

Obtain reward $r_2 = 5$

Action a_1: "right"

Action a_2: "fire"

(kill an alien)

Step 2: Define "Loss"

Start with observation s_1

Observation s_2

Observation s_3

This is an **_episode_**.

After many turns

Game Over
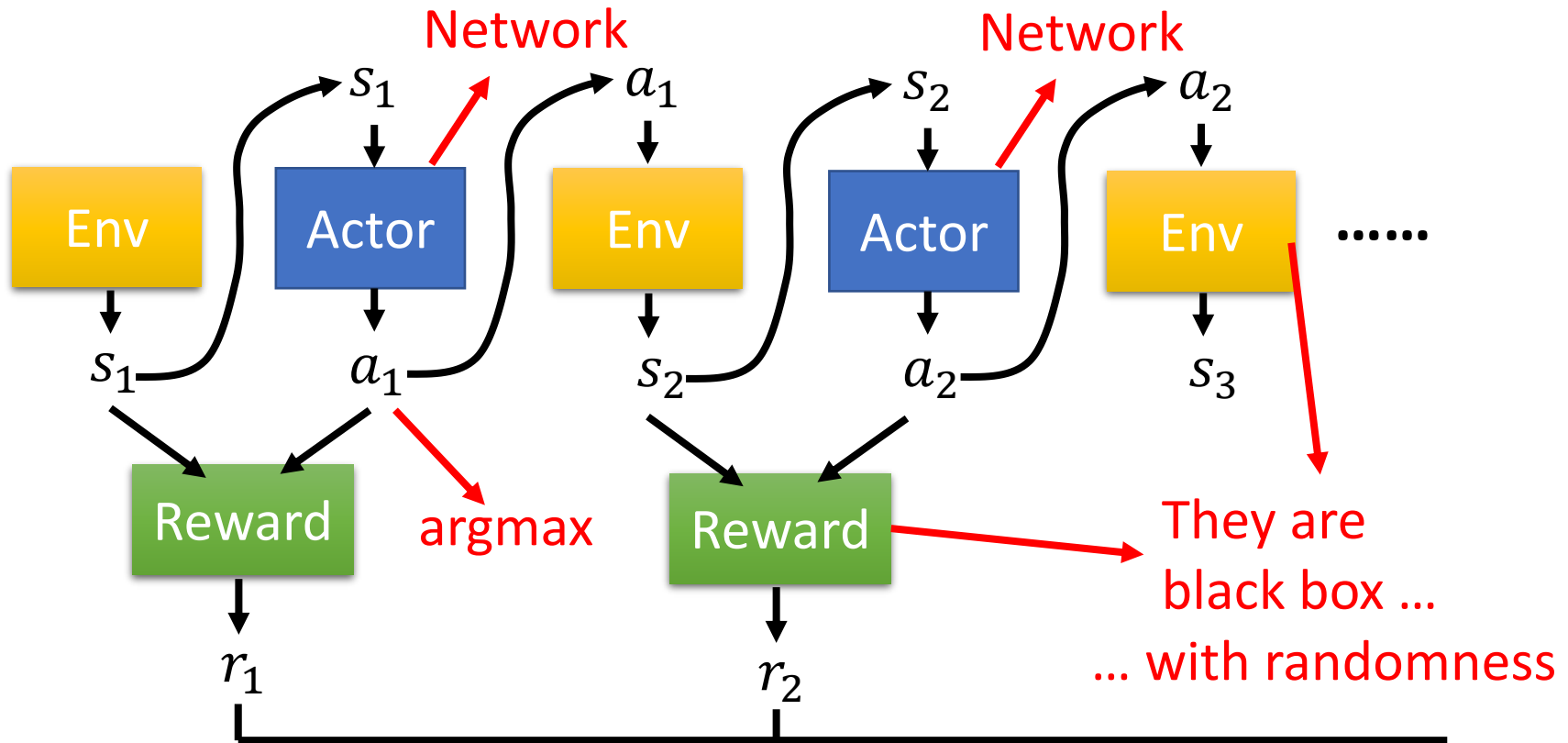(spaceship destroyed)

Obtain reward r_T

Action a_T

Total reward:

$$R = \sum_{t=1}^{T} r_t$$

What we want to maximize

Step 3: Optimization

Trajectory
$$\tau = \{s_1, a_1, s_2, a_2, \cdots\}$$

Network

Network

s_1 a_1 s_2 a_2

Env Actor Env Actor Env ......

s_1 a_1 s_2 a_2 s_3

Reward argmax Reward They are black box ... ... with randomness

r_1 r_2

How to do the optimization here is the main challenge in RL.

$$R(\tau) = \sum_{t=1}^{T} r_t$$

c.f. GAN

Outline

To learn more about policy gradient:
https://youtu.be/W8XF3ME8G2I

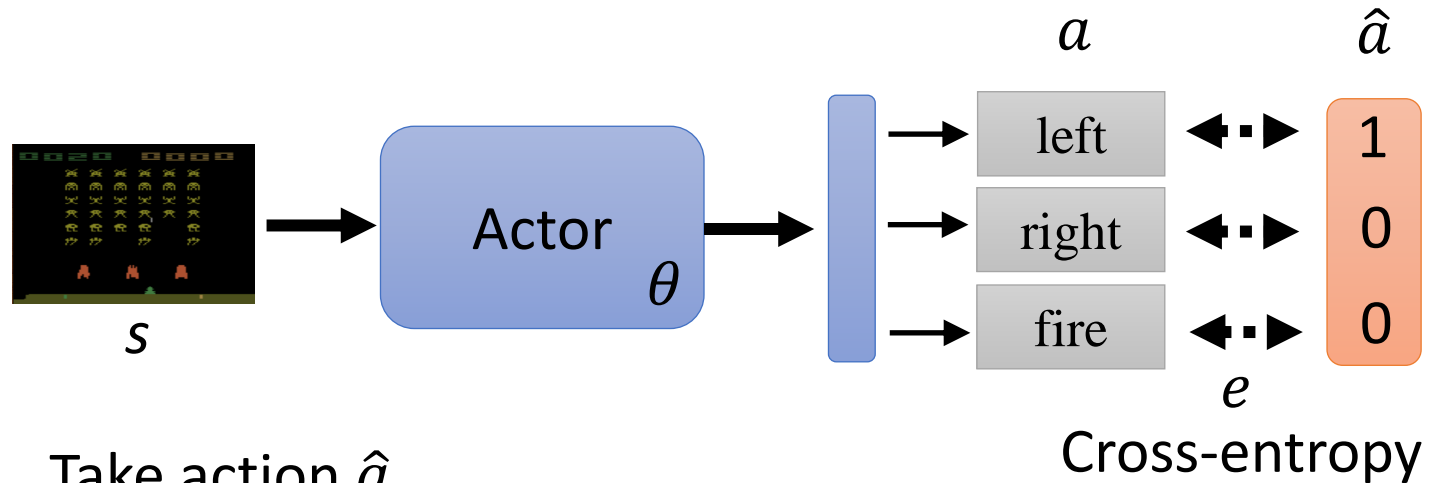What is RL? (Three steps in ML)

Policy Gradient

Actor-Critic

Reward Shaping

No Reward: Learning from Demonstration

How to control your actor

- Make it take (or don't take) a specific action $\hat{a}$ given specific observation s.
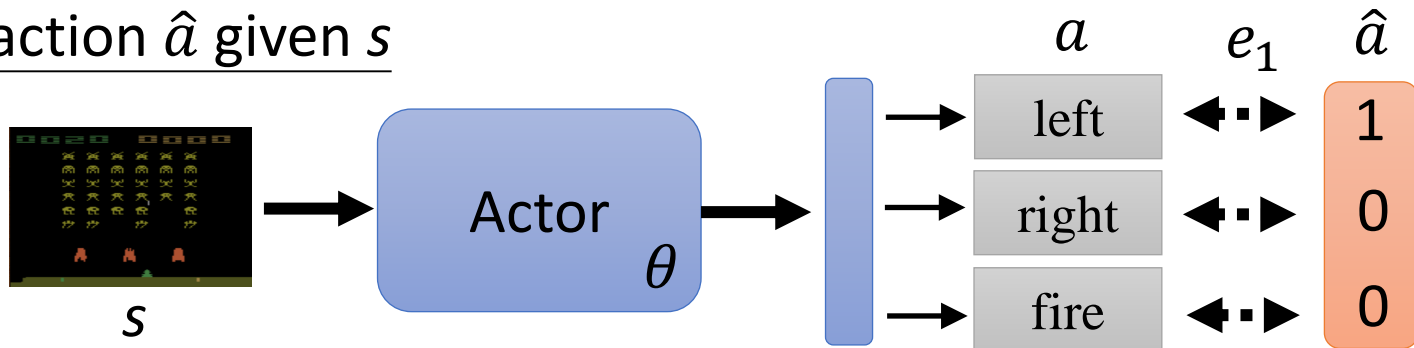

Take action $\hat{a}$

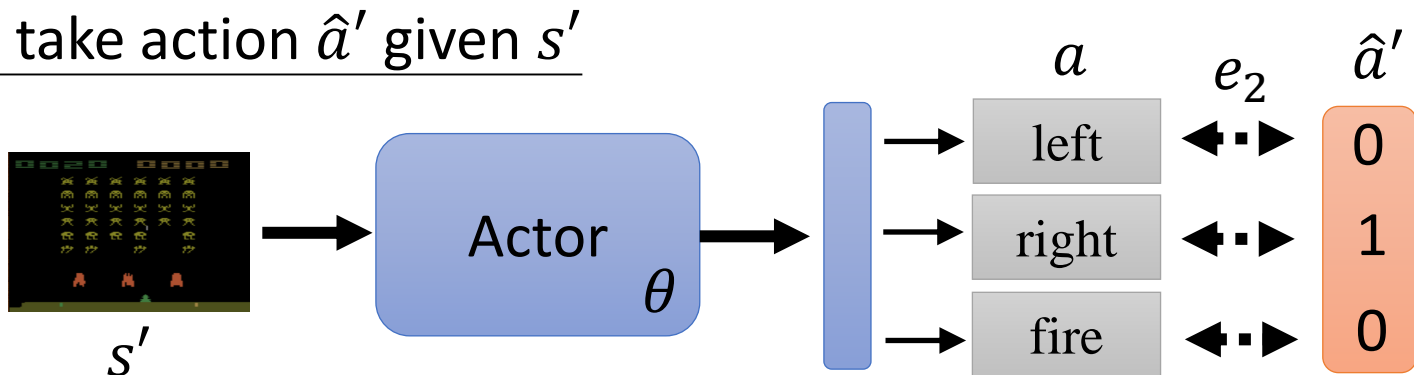$$L = e$$

Don't take action $\hat{a}$

$$L = -e$$

$$\theta^* = arg \min_{\theta} L$$

How to control your actor

Take action $\hat{a}$ given s

Don't take action $\hat{a}'$ given s'

$$L = e_1 - e_2 \qquad \theta^* = arg \min_{\theta} L$$

How to control your actor

Training Data

$\{s_1, \hat{a}_1\}$ +1   **Yes**

$\{s_2, \hat{a}_2\}$ -1   **No**

$\{s_3, \hat{a}_3\}$ +1   **Yes**

$\vdots$ $\vdots$

$\{s_N, \hat{a}_N\}$ -1   **No**

$$s \longrightarrow \boxed{\text{Actor} \quad \theta} \longrightarrow a$$

$$L = + e_1 - e_2 + e_3 \cdots - e_N$$

$$\theta^* = arg \min_{\theta} L$$

How to control your actor

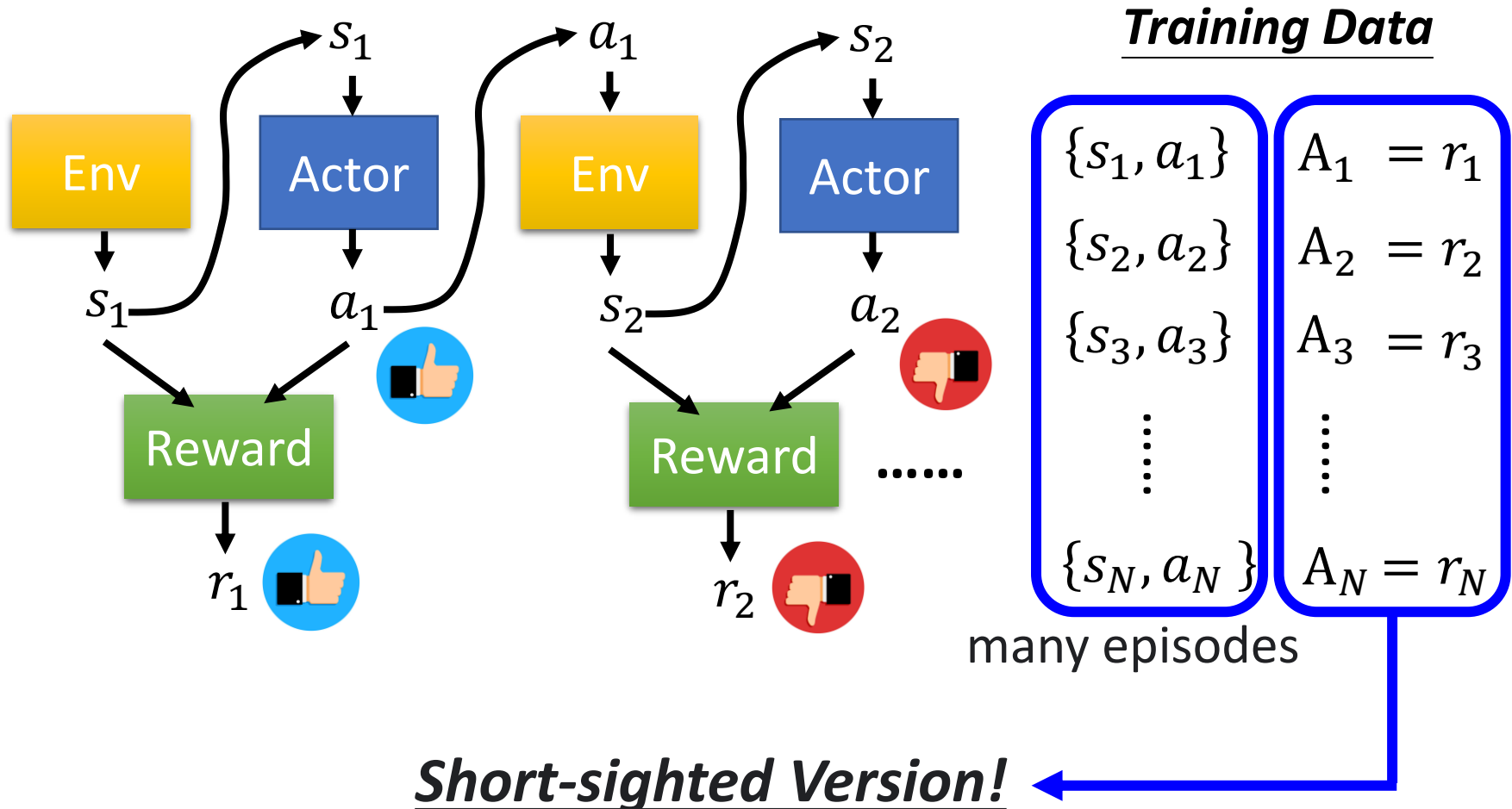**_Training Data_**

$\{s_1, \hat{a}_1\}$ A_1 +1.5

$\{s_2, \hat{a}_2\}$ A_2 - 0.5

$\{s_3, \hat{a}_3\}$ A_3 +0.5

$\vdots$ $\vdots$

$\{s_N, \hat{a}_N\}$ A_N -10

?   ?

$s \longrightarrow$ Actor $\longrightarrow a$
θ

$$L = \sum A_n e_n$$

$$\theta^* = arg \min_\theta L$$

Version 0

Training Data

$$\{s_1, a_1\} \quad A_1 = r_1$$
$$\{s_2, a_2\} \quad A_2 = r_2$$
$$\{s_3, a_3\} \quad A_3 = r_3$$
$$\vdots \qquad\quad \vdots$$
$$\{s_N, a_N\} \quad A_N = r_N$$
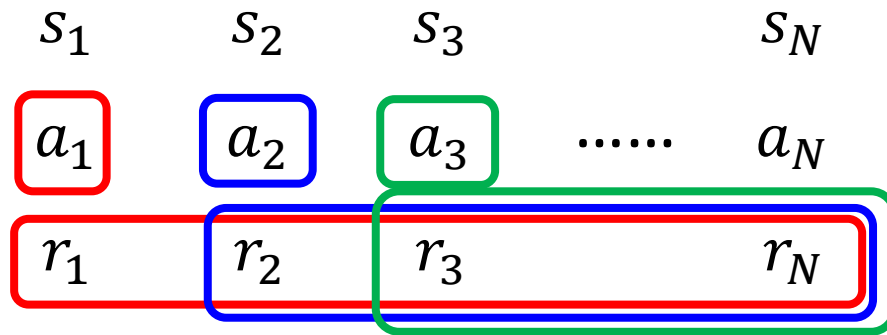
many episodes

Short-sighted Version!

Version 0
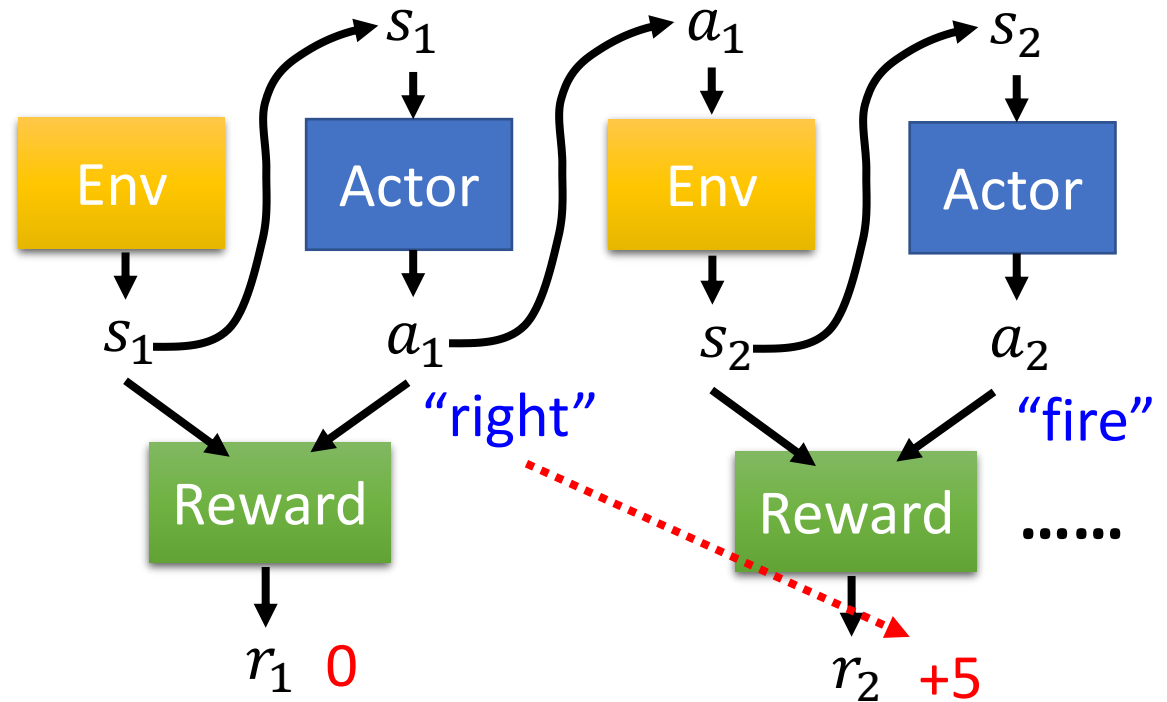
- An action **affect the subsequent observations and thus subsequent rewards.**

- **Reward delay:** Actor has to sacrifice immediate reward to gain more long-term reward.

- In space invader, only "fire" obtains reward, so vision 0 will learn an actor that always "fire"

Version 1

s_1 s_2 s_3 s_N

a_1 a_2 a_3 ...... a_N

r_1 r_2 r_3 r_N

$\{s_1, a_1\}$ $A_1 = G_1$

$\{s_2, a_2\}$ $A_2 = G_2$

$\{s_3, a_3\}$ $A_3 = G_3$

$\vdots$ $\vdots$

$G_1 = r_1 + r_2 + r_3 + \ldots\ldots + r_N$

$\{s_N, a_N\}$ $A_N = G_N$

$G_2 = r_2 + r_3 + \ldots\ldots + r_N$

$G_3 = r_3 + \ldots\ldots + r_N$

$$G_t = \sum_{n=t}^{N} r_n$$

cumulated reward

Version 2

Training Data

$s_1 \quad s_2 \quad s_3 \quad\quad\quad s_N$

$a_1 \quad a_2 \quad a_3 \quad \ldots\ldots \quad a_N$

$r_1 \quad r_2 \quad r_3 \quad\quad\quad r_N$

$\{s_1, a_1\} \quad A_1 = G_1'$

$\{s_2, a_2\} \quad A_2 = G_2'$

$\{s_3, a_3\} \quad A_3 = G_3'$

$\vdots \quad\quad \vdots$

$\{s_N, a_N\} \quad A_N = G_N'$
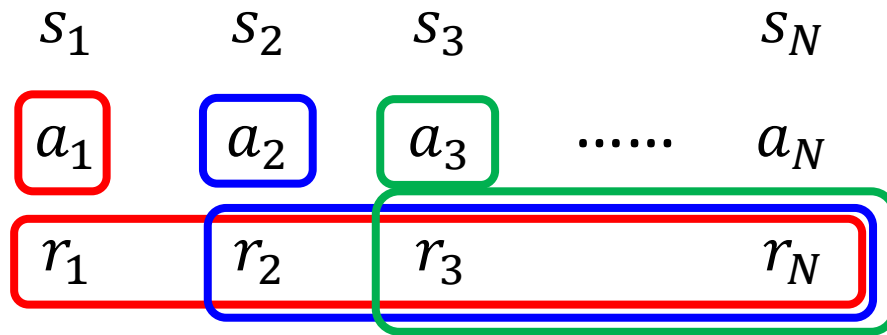
$G_1 = r_1 + r_2 + r_3 + \ldots\ldots + r_N$

$G_1' = r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots\ldots$

Discount factor $\gamma < 1$

$$G_t' = \sum_{n=t}^{N} \gamma^{n-t} r_n$$

Version 3

s_1 s_2 s_3 s_N

a_1 a_2 a_3 ...... a_N

r_1 r_2 r_3 r_N
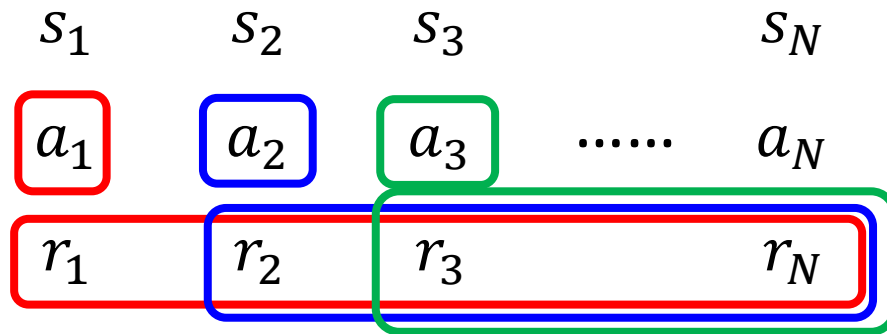
Good or bad reward is "relative"

If all the $r_n \geq 10$

$\qquad r_n = 10$ is negative ...

Minus by a baseline b ???

Make G'_t have positive and negative values

Training Data

$\{s_1, a_1\} \qquad A_1 = G'_1 - b$

$\{s_2, a_2\} \qquad A_2 = G'_2 - b$

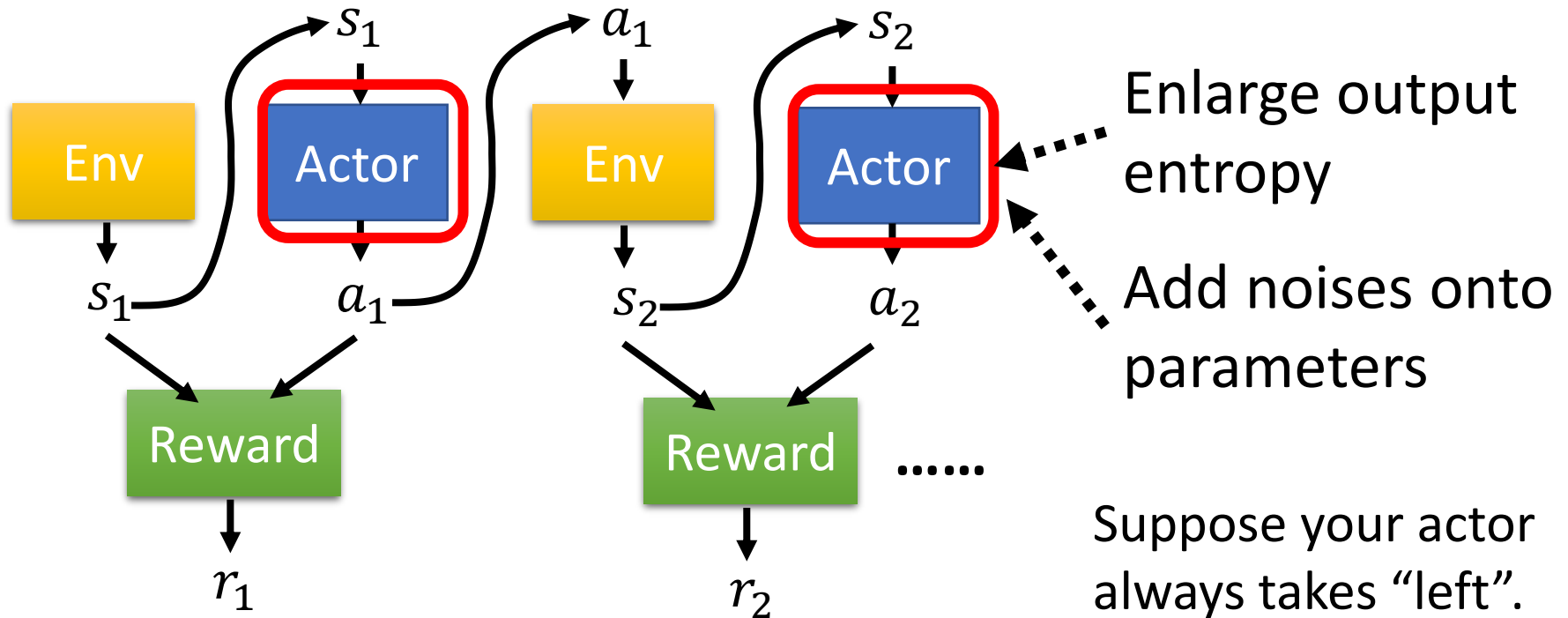$\{s_3, a_3\} \qquad A_3 = G'_3 - b$

$\qquad \vdots \qquad\qquad \vdots$

$\{s_N, a_N\} \qquad A_N = G'_N - b$

$$G'_t = \sum_{n=t}^{N} \gamma^{n-t} r_n$$

Exploitation v.s Exploration

Enlarge output entropy
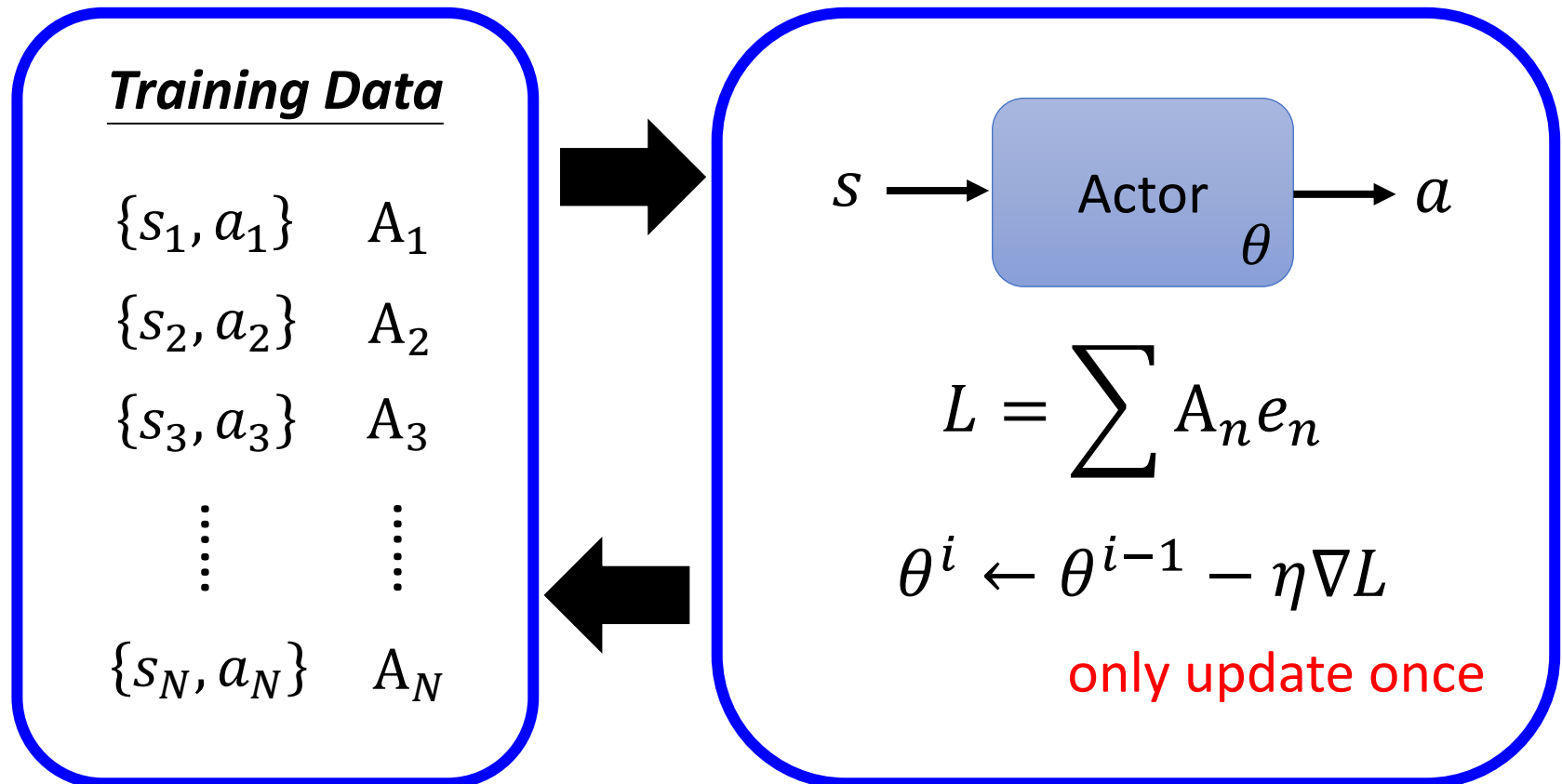
Add noises onto parameters

Suppose your actor always takes "left".

We never know what would happen if taking "fire".

The actor needs to have randomness during data collection.

Policy Gradient

Training Data

$\{s_1, a_1\}$ A_1

$\{s_2, a_2\}$ A_2

$\{s_3, a_3\}$ A_3

$\vdots$ $\vdots$

$\{s_N, a_N\}$ A_N

$s \longrightarrow$ Actor $\longrightarrow a$

θ

$$L = \sum A_n e_n$$

$$\theta^i \leftarrow \theta^{i-1} - \eta \nabla L$$

only update once

Each time you update the model parameters, you need to collect the whole training set again.

Policy Gradient

- Initialize actor network parameters θ^0

- For training iteration $i = 1$ to T

  - Using actor $\boxed{\theta^{i-1}}$ to interact
  - Obtain data $\{s_1, a_1\}, \{s_2, a_2\}, \dots, \{s_N, a_N\}$
  - Compute $A_1, A_2, \dots, A_N$

  Experience of θ^{i-1}

  - Compute loss L
  - $\boxed{\theta^i} \leftarrow \boxed{\theta^{i-1}} - \eta \nabla L$

  Data collection is in the "for loop" of training iterations.

  Cannot use to train θ^i

棋魂第八集

棋魂第八集
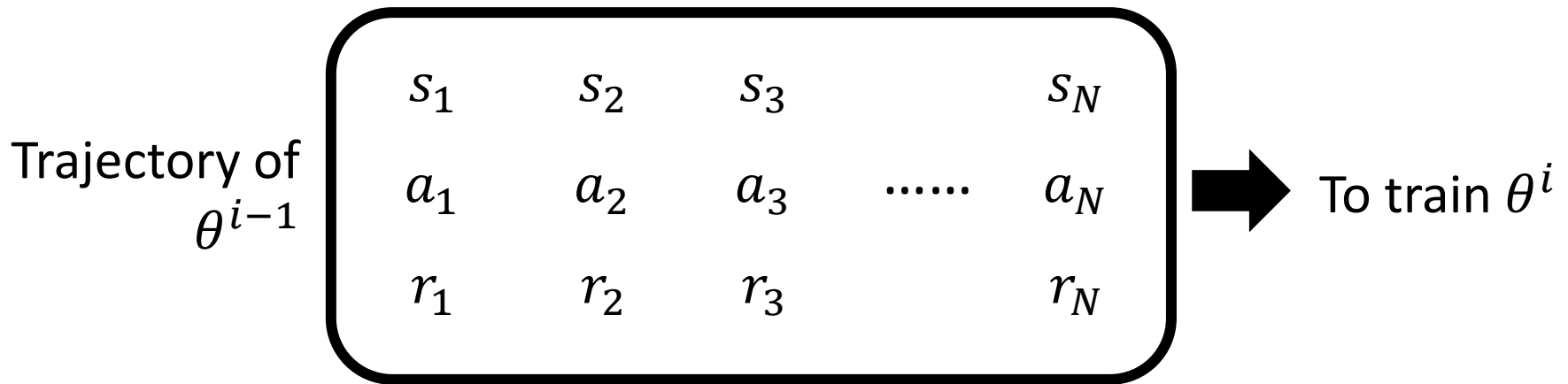
Policy Gradient

- Initialize actor network parameters θ^0
- For training iteration $i = 1$ to T
  - Using actor θ^{i-1} to interact
  - Obtain data $\{s_1, a_1\}, \{s_2, a_2\}, \dots, \{s_N, a_N\}$
  - Compute $A_1, A_2, \dots, A_N$
  - Compute loss L
  - $\theta^i \leftarrow \theta^{i-1} - \eta \nabla L$

May not observe by θ^i

Trajectory of θ^{i-1}

s_1	s_2	s_3		s_N
a_1	a_2	a_3	$\dots\dots$	a_N
r_1	r_2	r_3		r_N
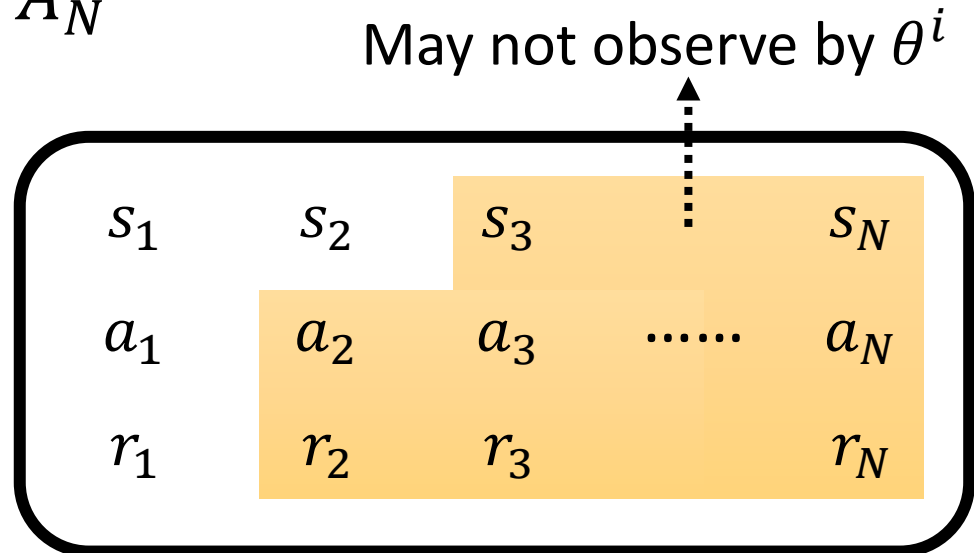
On-policy v.s. Off-policy

- The actor to train and the actor for interacting is the same. → On-policy

- Can the actor to train and the actor for interacting be different? → Off-policy

Trajectory of
θ^{i-1}

$$
\begin{array}{cccc}
s_1 & s_2 & s_3 & s_N \\
a_1 & a_2 & a_3 & \cdots\cdots & a_N \\
r_1 & r_2 & r_3 & r_N
\end{array}
$$

➡ To train θ^i

In this way, we do not have to collection data after each update.

Off-policy → Proximal Policy Optimization (PPO)

• The actor to train has to know its difference from the actor to interact.

video:
https://youtu.be/OAKAZhFmYoI

Not apply to everyone

the actor to train

如果喜歡一個女孩
就勇敢去告白
我用這個方法
從來沒失敗過

一-克里斯·伊凡
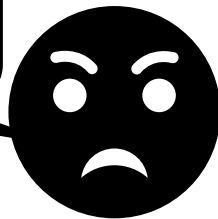
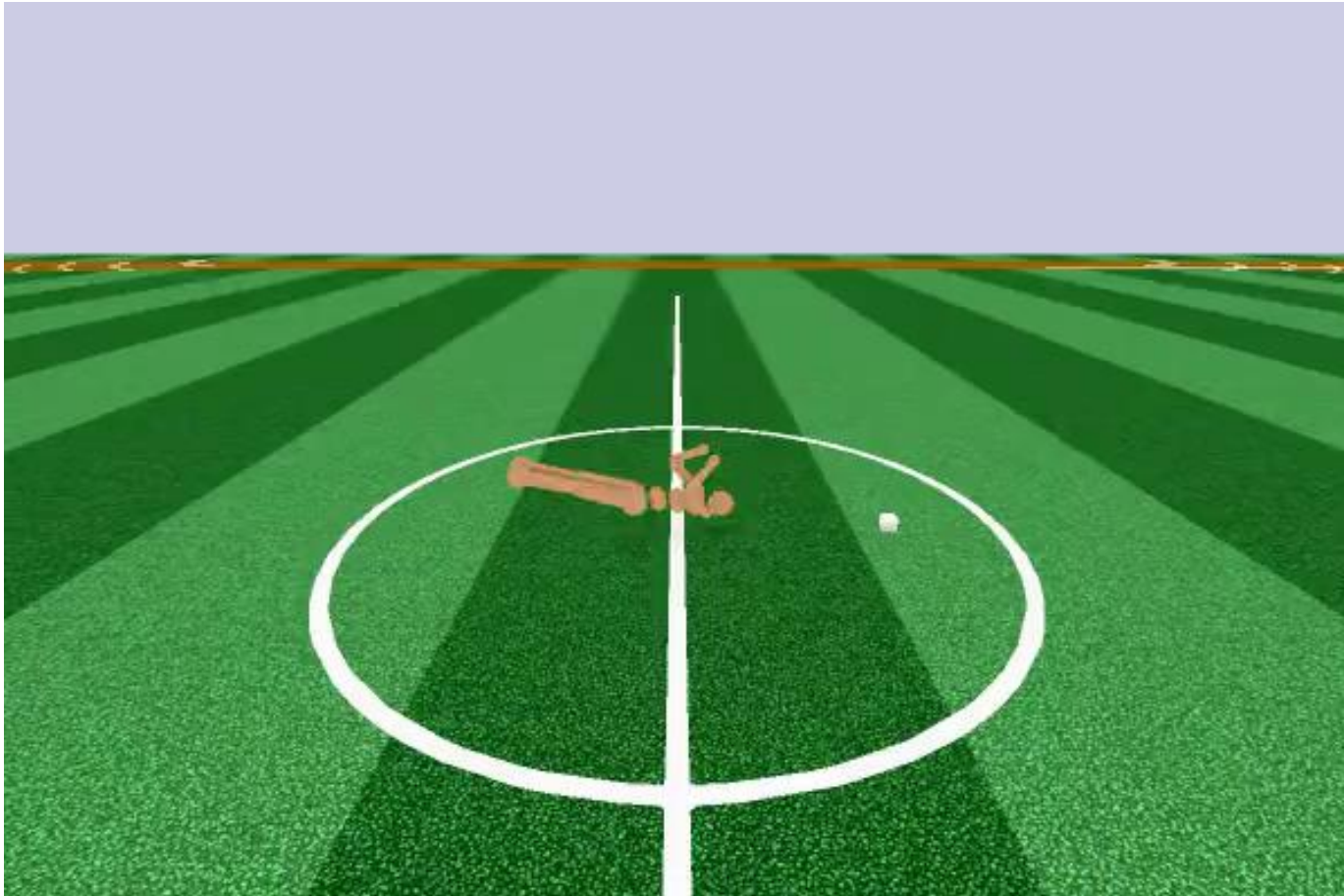the actor to interact

DeepMind - PPO

It might
look goofy ...

OpenAI – PPO

Outline

What is RL? (Three steps in ML)

Policy Gradient

Actor-Critic

Reward Shaping

No Reward: Learning from Demonstration

Critic

- A critic does not directly determine the action.

- Given an actor θ, it evaluates how good the actor is

- Value function $V^{\theta}(s)$
  - When using actor θ, the discounted *cumulated* reward expects to be obtained after seeing s

$$s \rightarrow V^{\theta} \rightarrow V^{\theta}(s)$$

scalar
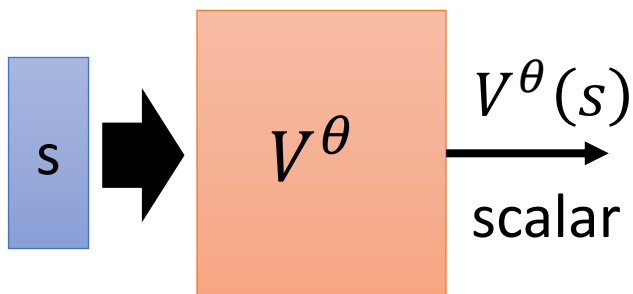
$V^{\theta}(s)$ is large
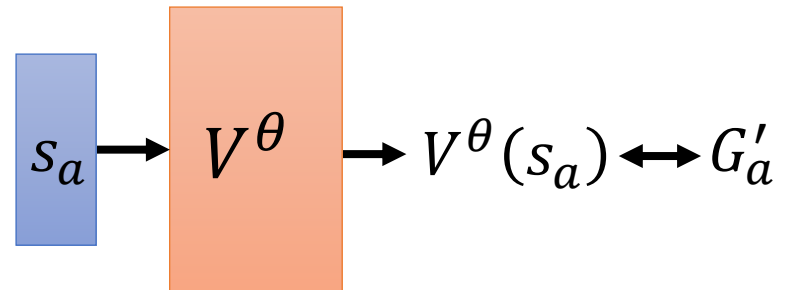
$V^{\theta}(s)$ is smaller

How to estimate $V^\theta(s)$

- **Monte-Carlo (MC) based approach**

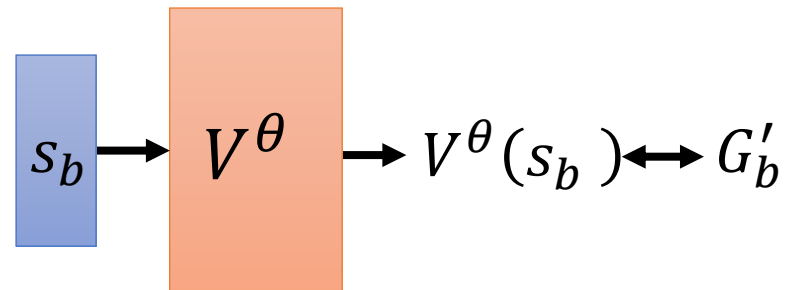 The critic watches actor θ to interact with the environment.

After seeing s_a,

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

$$s_a \rightarrow V^\theta \rightarrow V^\theta(s_a) \leftrightarrow G_a'$$

After seeing s_b,

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

$$s_b \rightarrow V^\theta \rightarrow V^\theta(s_b) \leftrightarrow G_b'$$
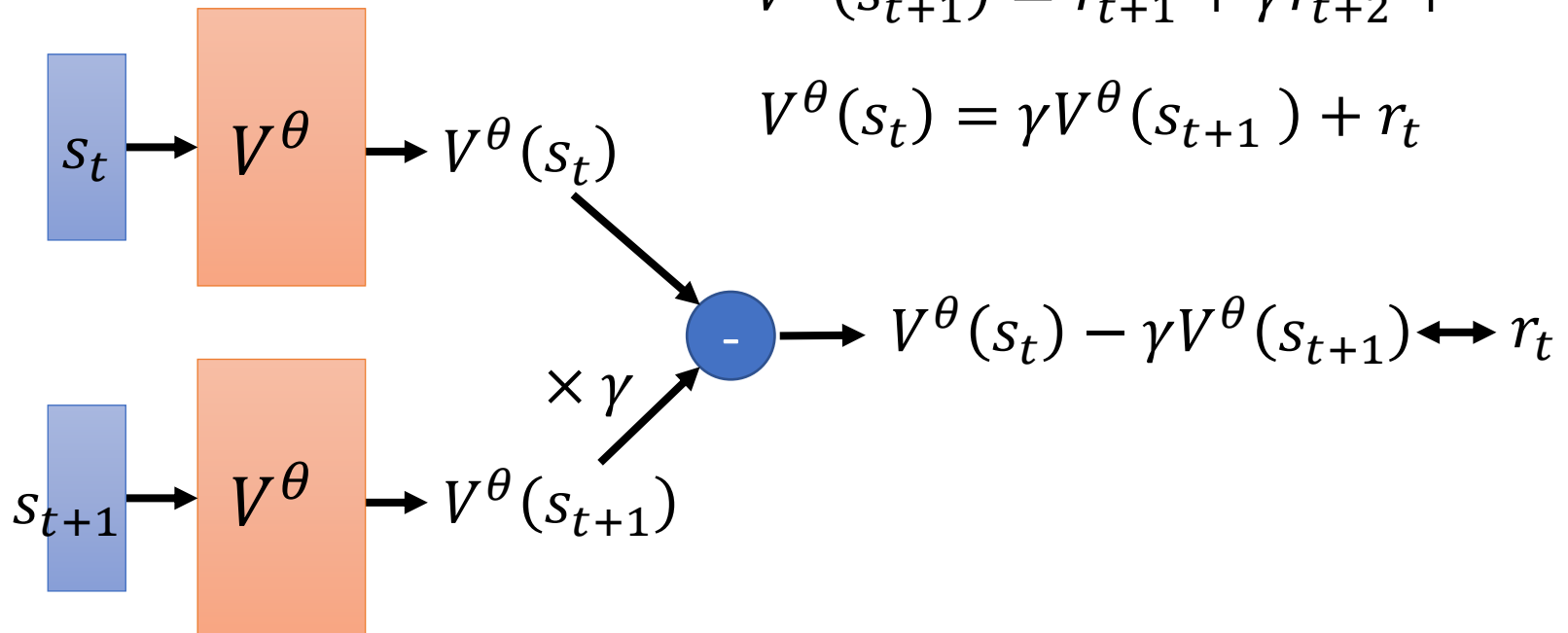
How to estimate $V^\pi(s)$

- **Temporal-difference (TD) approach**

$\cdots s_t, a_t, r_t, s_{t+1} \cdots$

$$V^\theta(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+1} \ldots$$

$$V^\theta(s_{t+1}) = r_{t+1} + \gamma r_{t+2} + \cdots$$

$$V^\theta(s_t) = \gamma V^\theta(s_{t+1}) + r_t$$

$s_t \rightarrow V^\theta \rightarrow V^\theta(s_t)$

$s_{t+1} \rightarrow V^\theta \rightarrow V^\theta(s_{t+1})$

$\times \gamma$

$V^\theta(s_t) - \gamma V^\theta(s_{t+1}) \leftrightarrow r_t$

MC v.s. TD

- The critic has observed 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

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

$$V^\theta(s_a) =? \quad \textcolor{red}{0?} \quad \textcolor{red}{3/4?}$$

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

Temporal-difference:

$$V^\theta(s_a) = V^\theta(s_b) + r$$

$$\textcolor{red}{3/4} \qquad \textcolor{red}{3/4} \qquad \textcolor{red}{0}$$

(Assume γ = 1, and the actions are ignored here.)

Version 3.5

$$s_1 \quad s_2 \quad s_3 \qquad\qquad s_N$$

$$a_1 \quad a_2 \quad a_3 \quad\cdots\cdots\quad a_N$$

$$r_1 \quad r_2 \quad r_3 \qquad\qquad r_N$$

$s \longrightarrow V^\theta \longrightarrow V^\theta(s)$
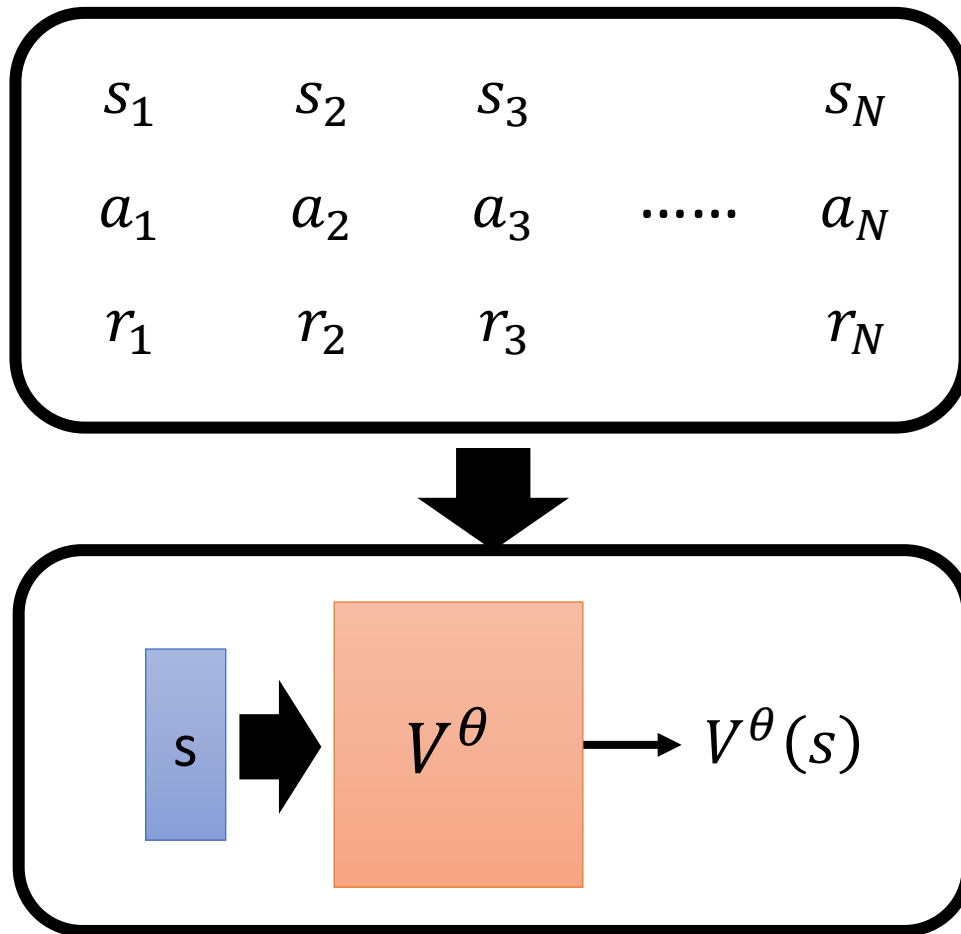
Training Data

$\{s_1, a_1\} \quad \mathrm{A}_1 = G_1' - b$

$\{s_2, a_2\} \quad \mathrm{A}_2 = G_2' - b$

$\{s_3, a_3\} \quad \mathrm{A}_3 = G_3' - b$

$\vdots \qquad\qquad \vdots$

$\{s_N, a_N\} \quad \mathrm{A}_N = G_N' - b$

Version 3.5

$$s_1 \quad s_2 \quad s_3 \quad\quad\quad s_N$$

$$a_1 \quad a_2 \quad a_3 \quad \cdots\cdots \quad a_N$$

$$r_1 \quad r_2 \quad r_3 \quad\quad\quad r_N$$

$$s \rightarrow V^\theta \rightarrow V^\theta(s)$$
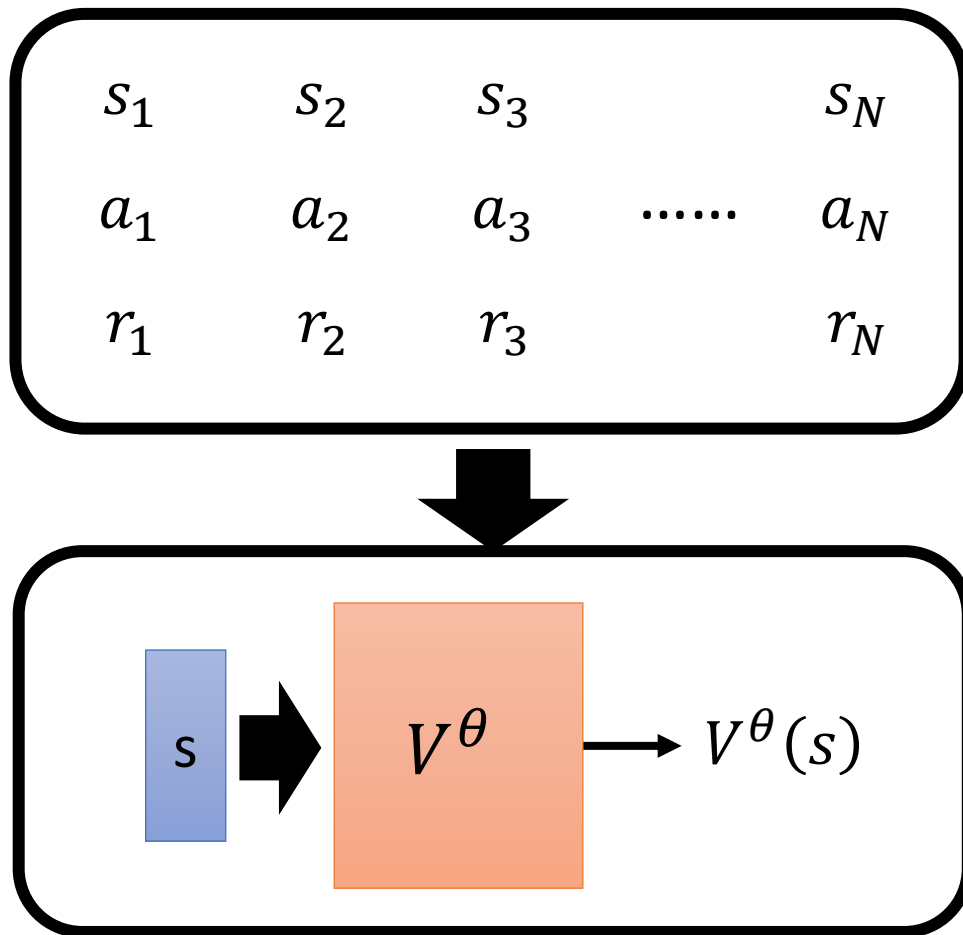
Training Data

$\{s_1, a_1\} \quad A_1 = G_1' - V^\theta(s_1)$

$\{s_2, a_2\} \quad A_2 = G_2' - V^\theta(s_2)$

$\{s_3, a_3\} \quad A_3 = G_3' - V^\theta(s_3)$

$\vdots \qquad\qquad \vdots$

$\{s_N, a_N\} \quad A_N = G_N' - V^\theta(s_N)$

Version 3.5

$\{s_t, a_t\}$ $A_t = G'_t - V^\theta(s_t)$

$G = 100$
$G = 3$
$G = 1$
$G = 2$
$G = -10$

$V^\theta(s_t)$

s_t

(not necessary take a_t)
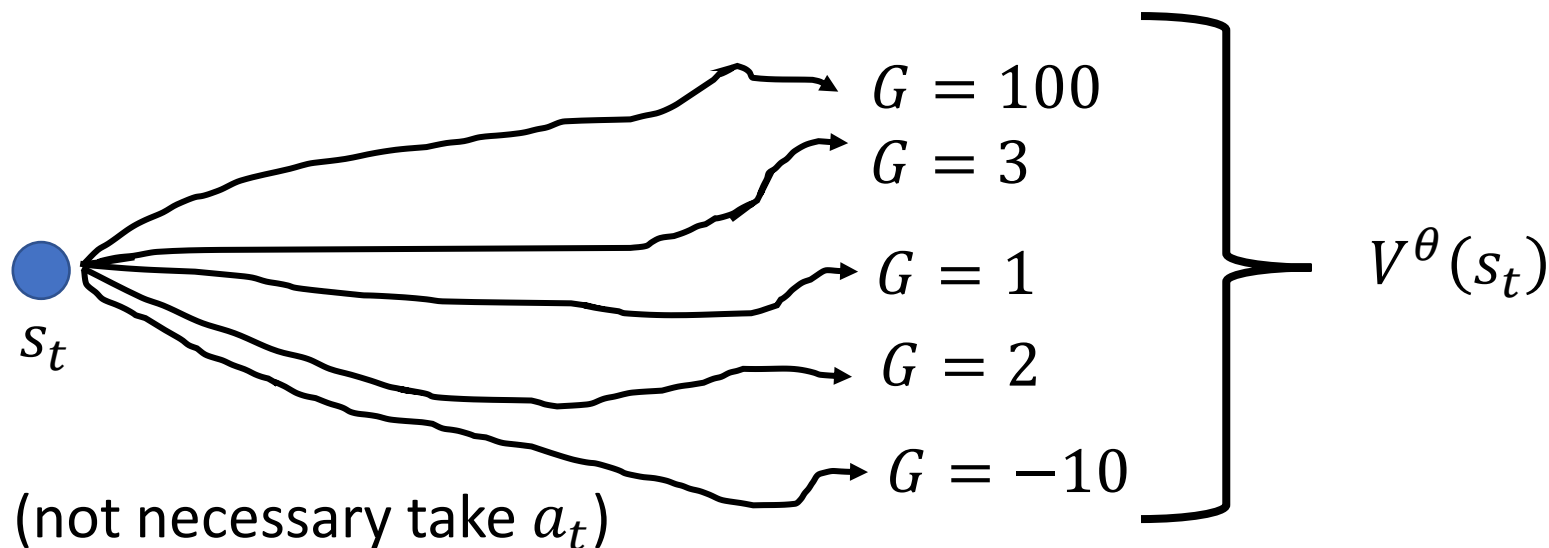
$A_t > 0$

a_t is better than average.
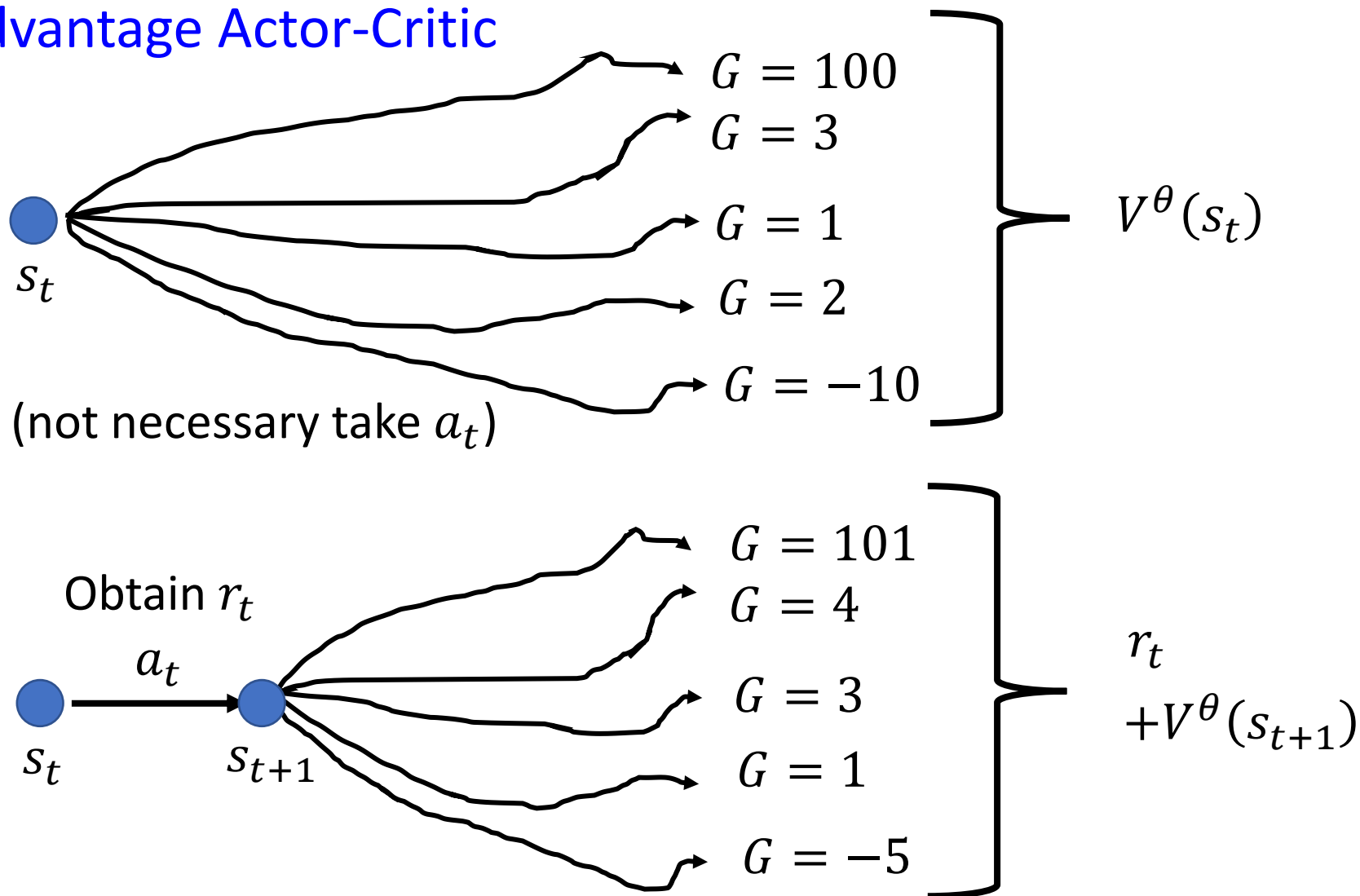
a_t

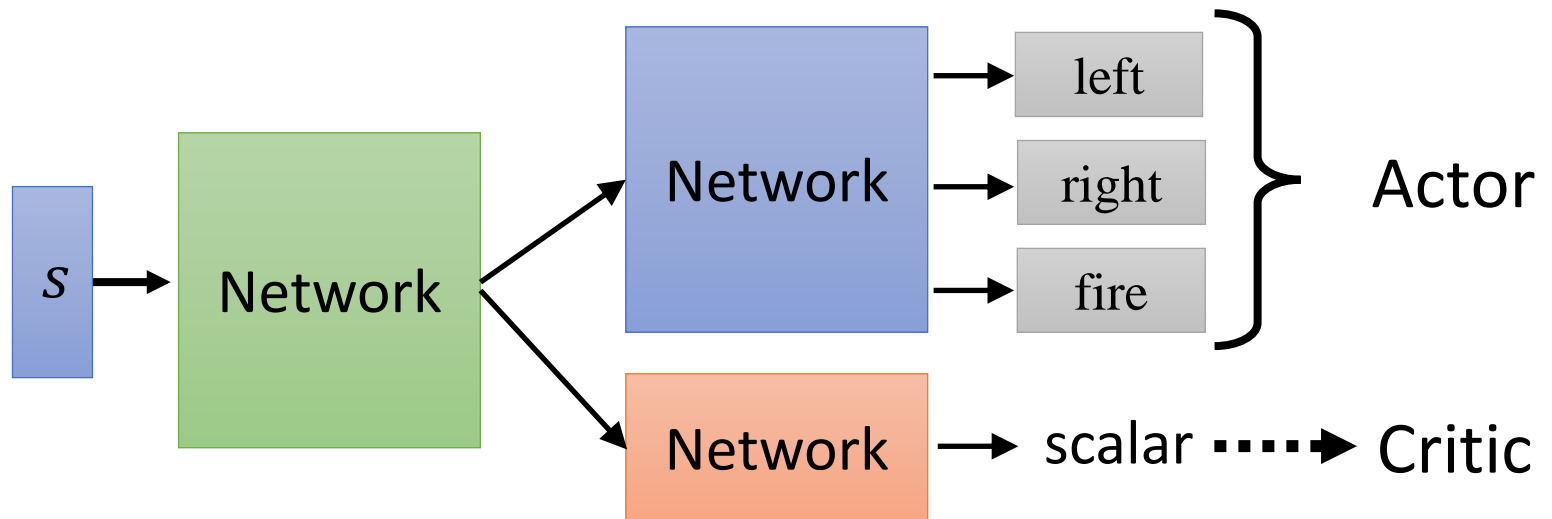s_t G'_t $A_t < 0$
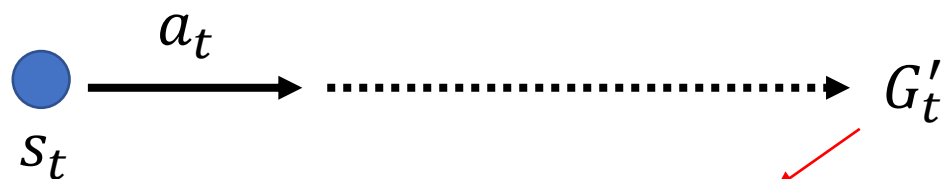
Just a sample

a_t is worse than average.

Version 4

$$r_t + V^\theta(s_{t+1}) - V^\theta(s_t)$$

$$\{s_t, a_t\} \quad A_t = \cancel{G_t' - V^\theta(s_t)}$$

Advantage Actor-Critic

s_t

$G = 100$
$G = 3$
$G = 1$
$G = 2$
$G = -10$

$V^\theta(s_t)$

(not necessary take a_t)

Obtain r_t

a_t

$s_t \quad s_{t+1}$

$G = 101$
$G = 4$
$G = 3$
$G = 1$
$G = -5$

r_t
$+V^\theta(s_{t+1})$

Tip of Actor-Critic

- The parameters of actor and critic can be shared.

Deep Q Network (DQN)

Video:
https://youtu.be/o_g9JUMw1Oc
https://youtu.be/2-zGCx4iv_k
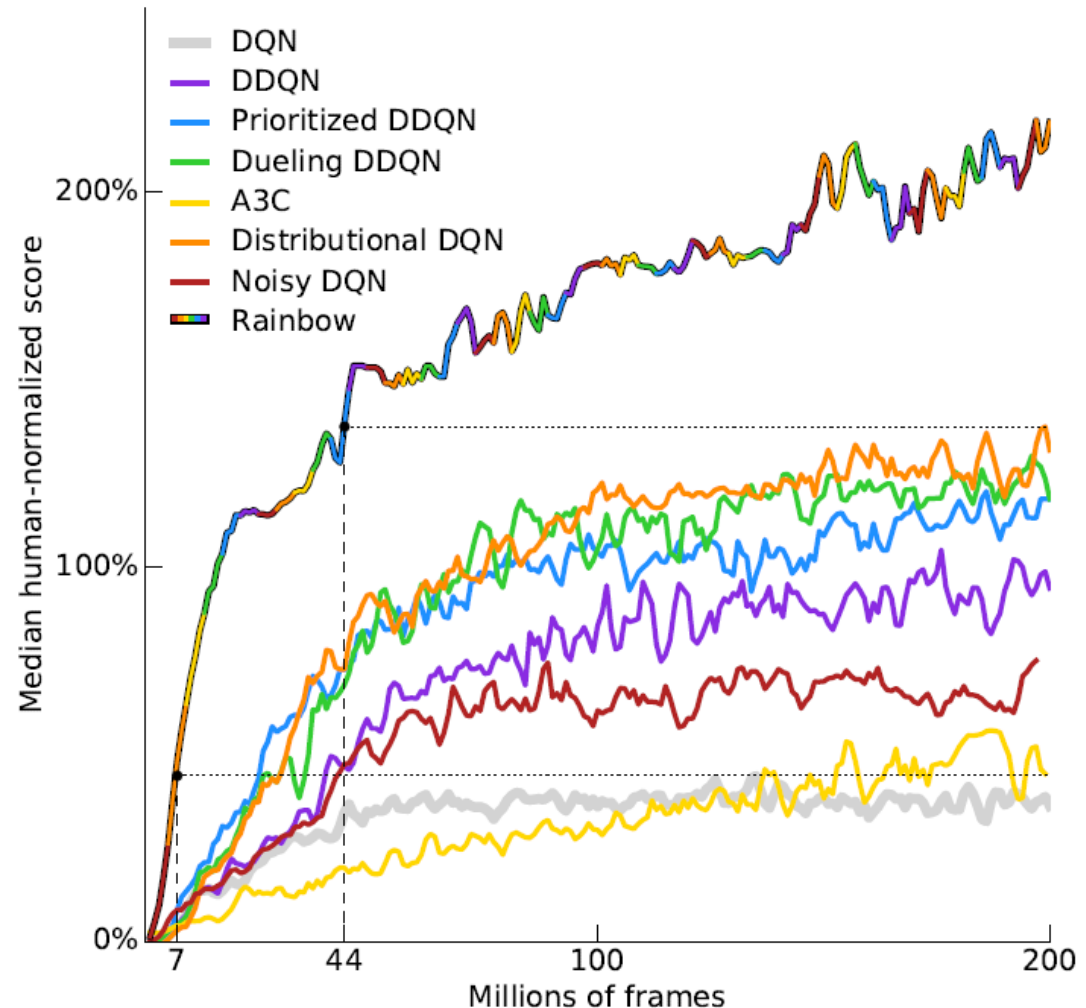
https://arxiv.org/abs/1710.02298

Outline

What is RL? (Three steps in ML)

Policy Gradient

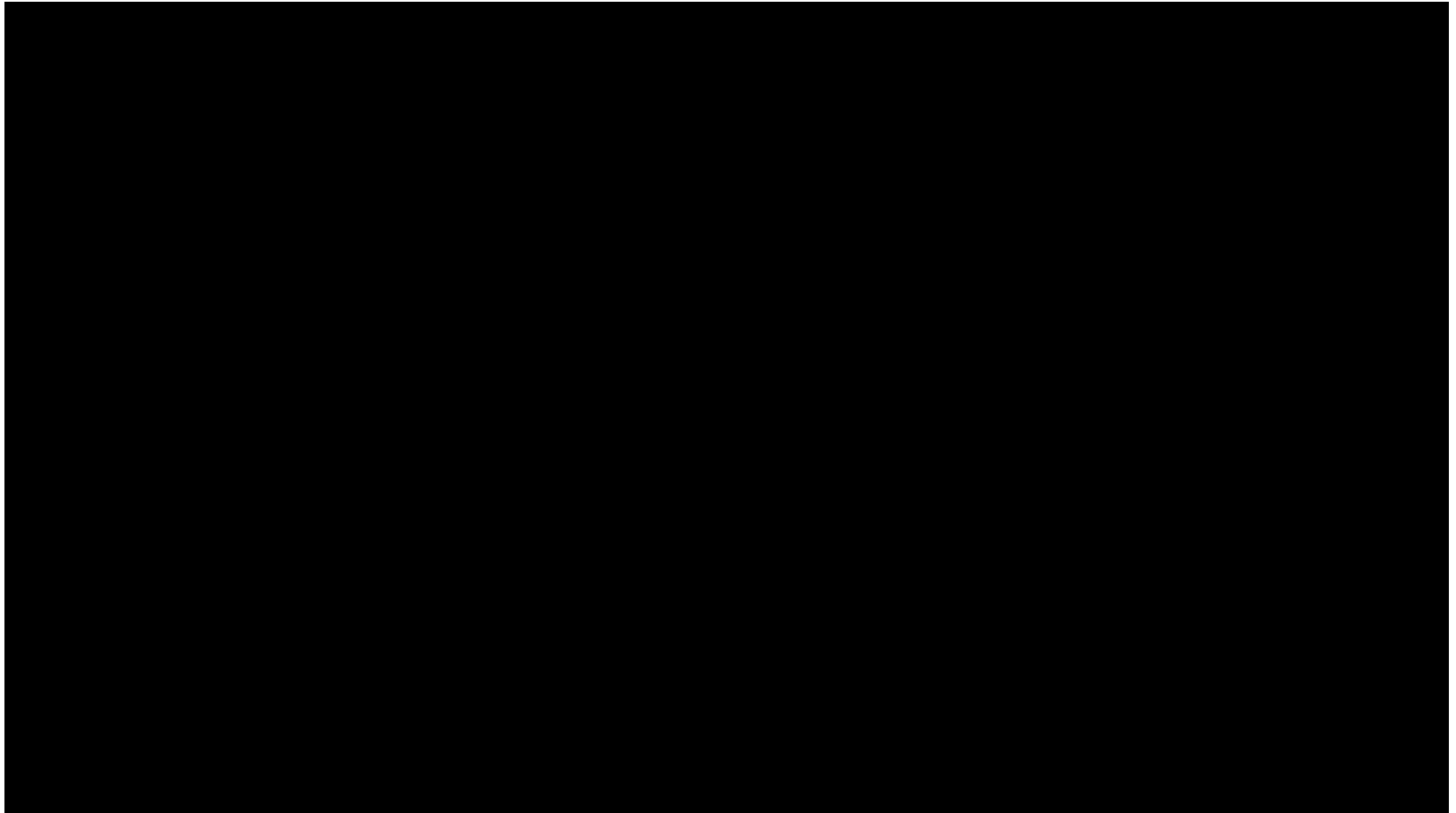Actor-Critic

Reward Shaping

No Reward: Learning from Demonstration

Reward Shaping

- Reward can be sparse.

- Reward in human's life is also very sparse. We are good at reward shaping.

Visual Doom AI Competition @ CIG 2016

https://www.youtube.com/watch?v=94EPSjQH38Y

Reward Shaping

VizDoom

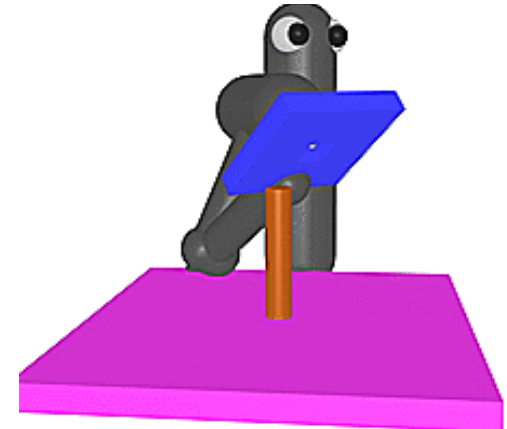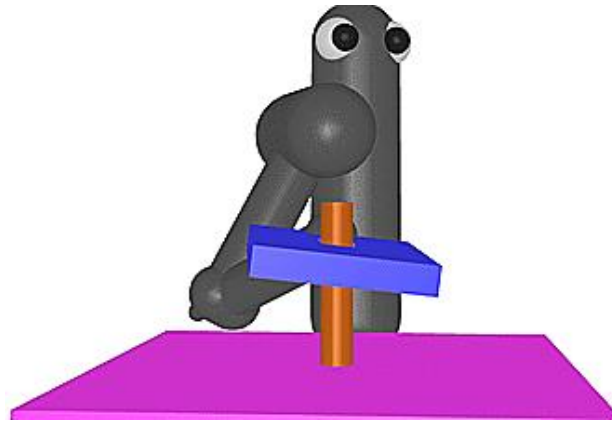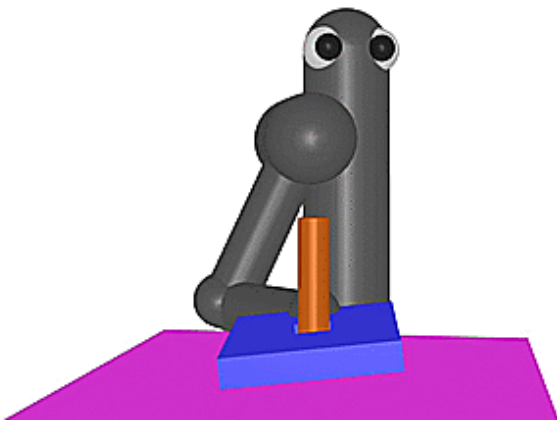| Parameters	Description	FlatMap | CIGTrack1
living	Penalize agent who just lives	-0.008 / action
health_loss	Penalize health decrement	-0.05 / unit
ammo_loss	Penalize ammunition decrement	-0.04 / unit
health_pickup	Reward for medkit pickup	0.04 / unit
ammo_pickup	Reward for ammunition pickup	0.15 / unit
dist_penalty	Penalize the agent when it stays	-0.03 / action
dist_reward	Reward the agent when it moves	9e-5 / unit distance
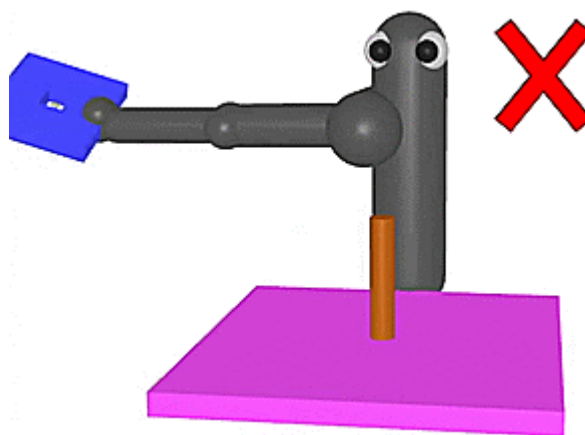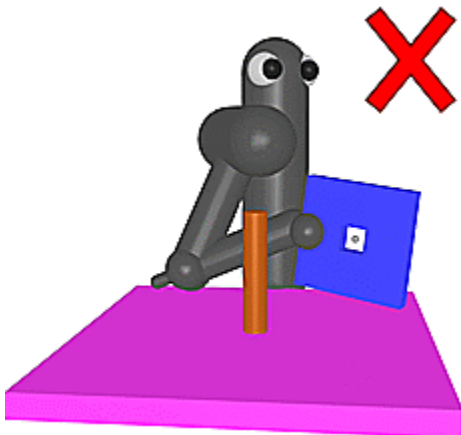
Get reward, when closer

Need domain knowledge
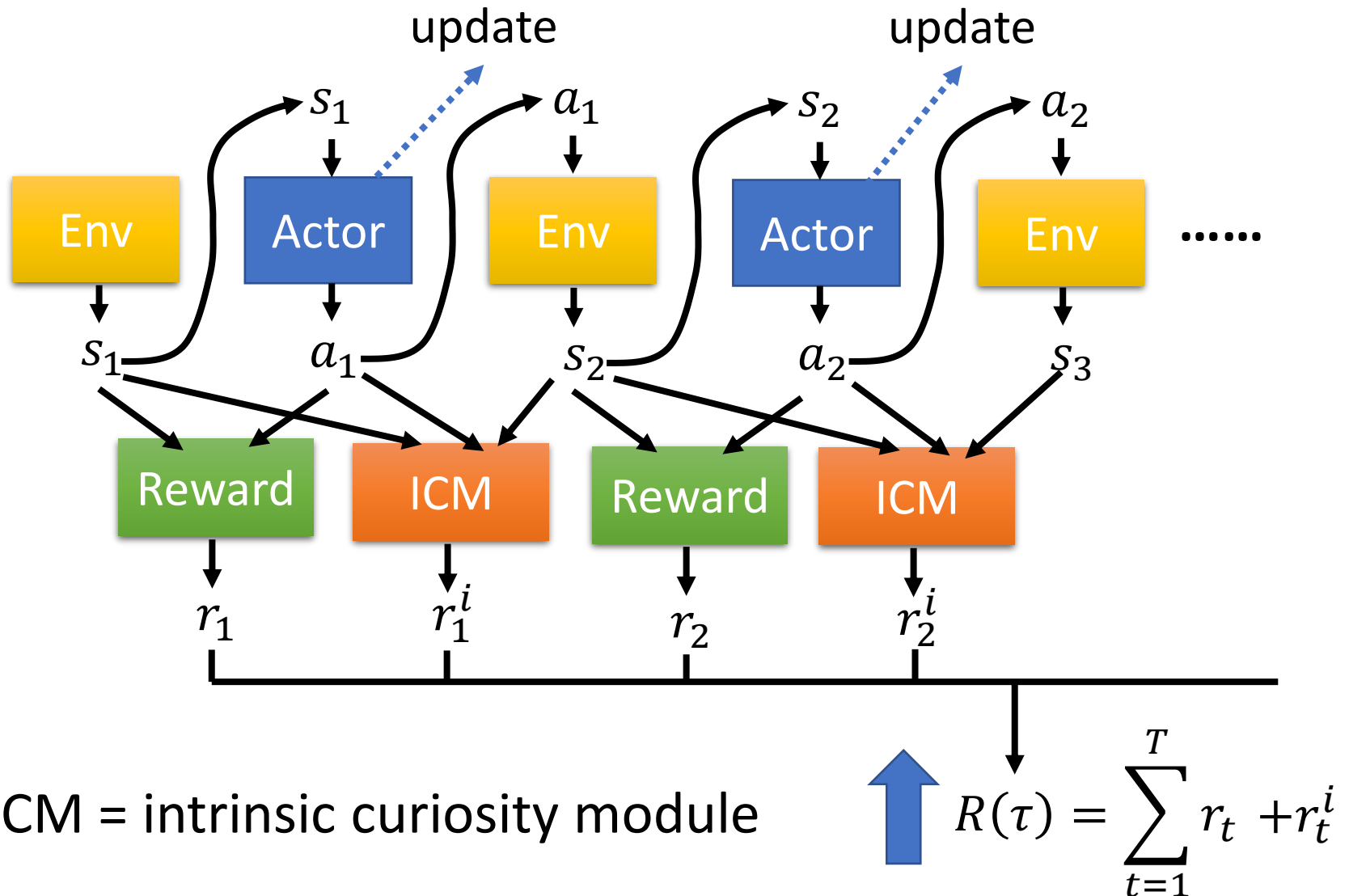
Curriculum Learning

- Starting from simple training examples, and then becoming harder and harder.

VizDoom

	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
Speed	0.2	0.2	0.4	0.4	0.6	0.8	0.8	1.0
Health	40	40	40	60	60	60	80	100

https://bair.berkeley.edu/blog/2017/12/20/reverse-curriculum/

Curiosity

ICM = intrinsic curiosity module

$$R(\tau) = \sum_{t=1}^{T} r_t + r_t^i$$

Outline

What is RL? (Three steps in ML)

Policy Gradient

Actor-Critic

Reward Shaping

No Reward: Learning from Demonstration

Motivation

- Even define reward can be challenging in some tasks.
- Hand-crafted rewards can lead to uncontrolled behavior.

Three Laws of Robotics:
1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.
3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Laws.
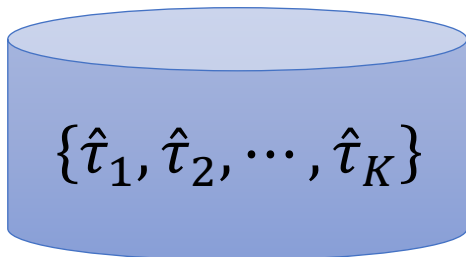
?????
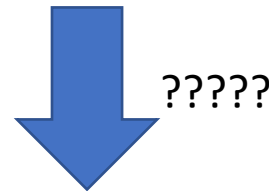
restraining individual human behavior and sacrificing some humans will ensure humanity's survival

Imitation Learning
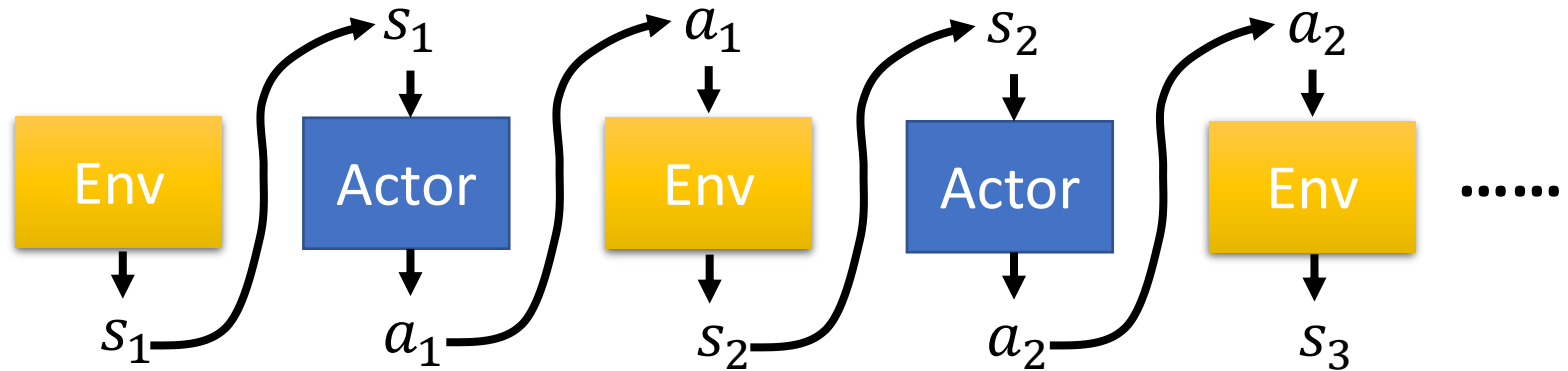
Actor can interact with the environment, but reward function is not available

We have demonstration of the expert.

$$\{\hat{\tau}_1, \hat{\tau}_2, \cdots, \hat{\tau}_K\}$$

Each $\hat{\tau}$ is a trajectory of the export.
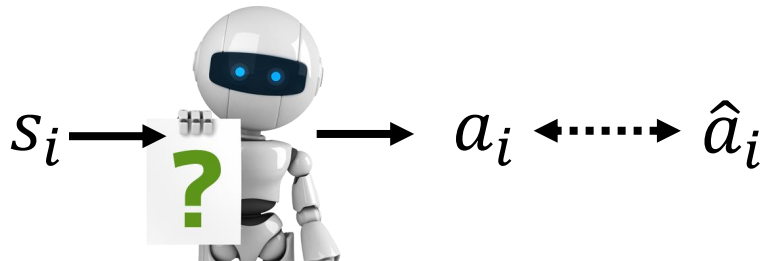
Self driving: record human drivers

Robot: grab the arm of robot

Isn't it Supervised Learning?

- Self-driving cars as example

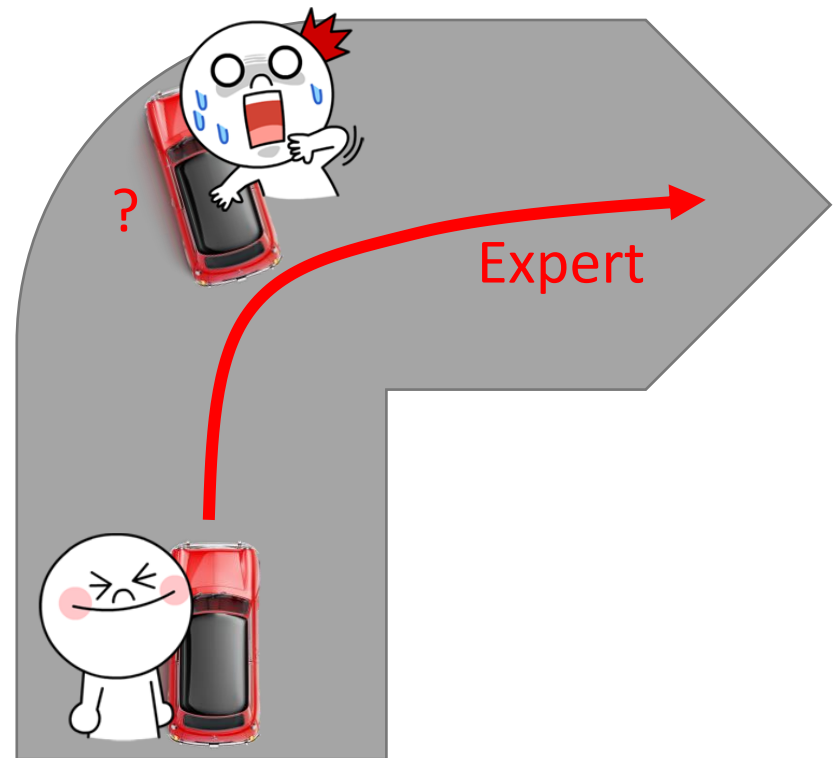$$\hat{\tau} = \{s_1, \hat{a}_1, s_2, \hat{a}_2, \cdots\}$$

forward

$s_i \longrightarrow$ $\longrightarrow a_i \longleftrightarrow \hat{a}_i$

Problem: The experts only sample limited observation.

Yes, also known at *Behavior Cloning*

?

Expert

More problem ……

The agent will copy every behavior, even irrelevant actions.

https://www.youtube.com/watch?v=j2FSB3bseek

Inverse Reinforcement Learning

demonstration of the expert

Environment

$\{\hat{\tau}_1, \hat{\tau}_2, \cdots, \hat{\tau}_K\}$

Reward Function

Reinforcement Learning

Expert

Using the reward function to find the ***optimal actor***.
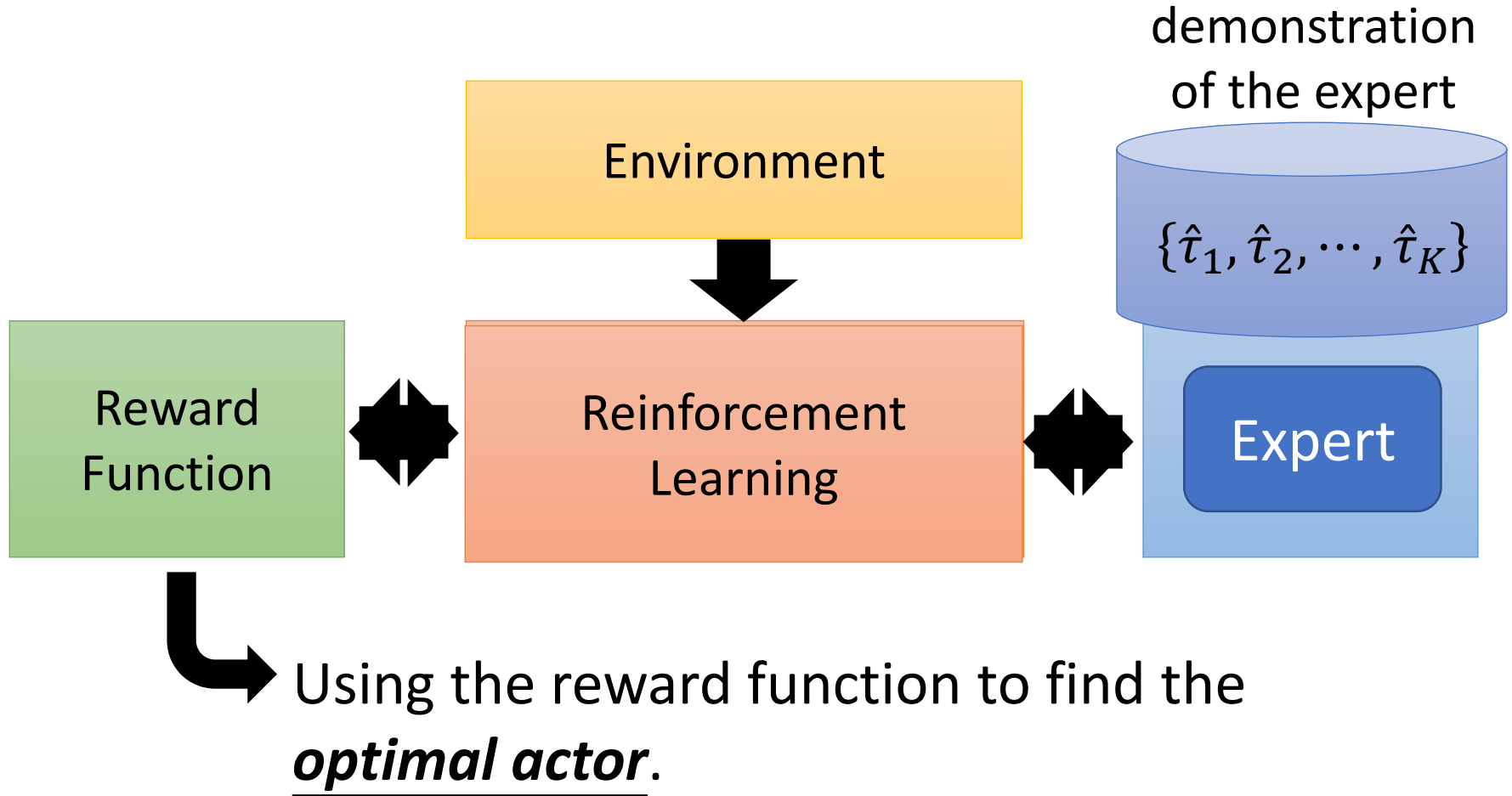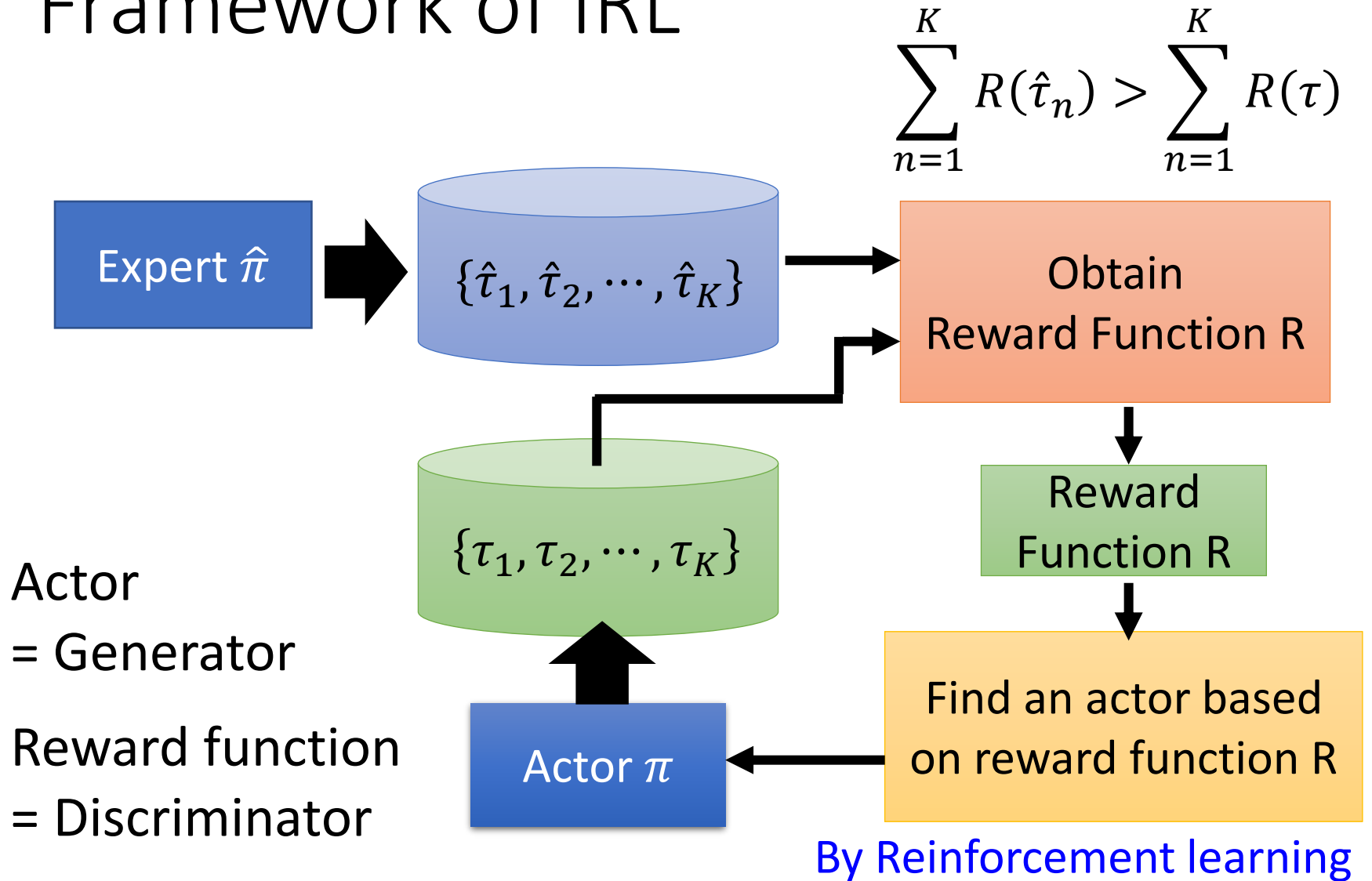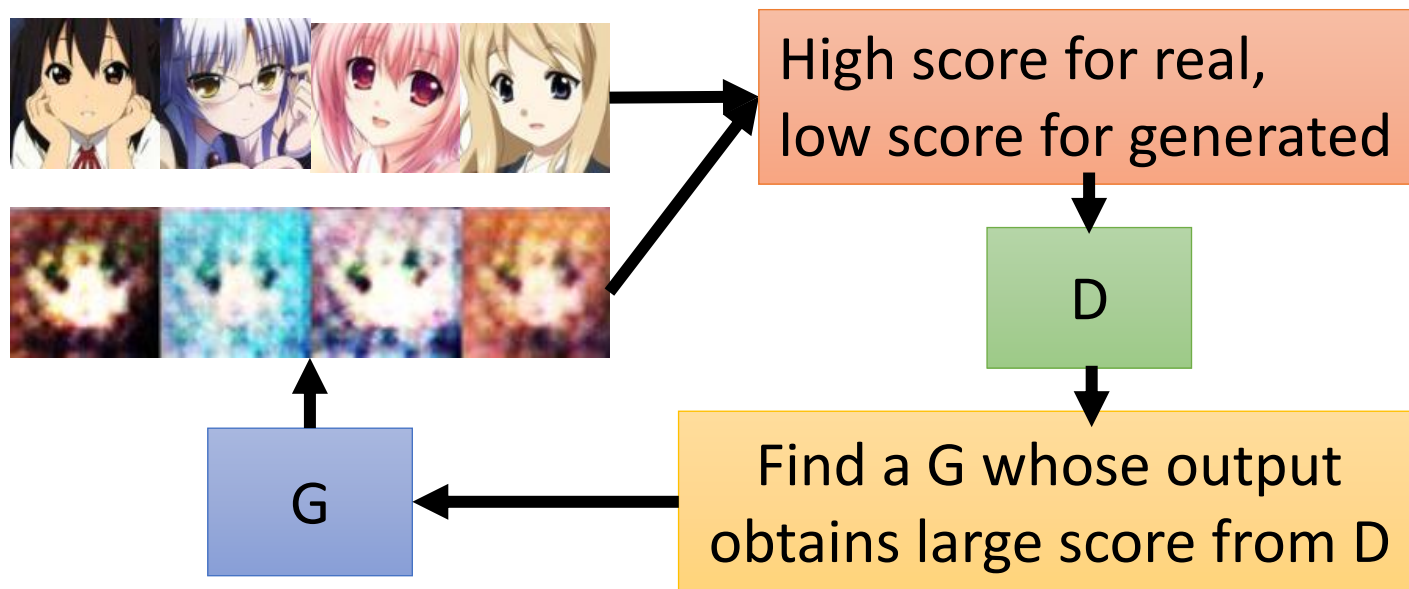
Inverse Reinforcement Learning

- Principle: ***The teacher is always the best***.

- Basic idea:
  - Initialize an actor
  - In each iteration
    - The actor interacts with the environments to obtain some trajectories.
    - Define a reward function, which makes the trajectories of the teacher better than the actor.
    - The actor learns to maximize the reward based on the new reward function.
  - Output the reward function and the actor learned from the reward function
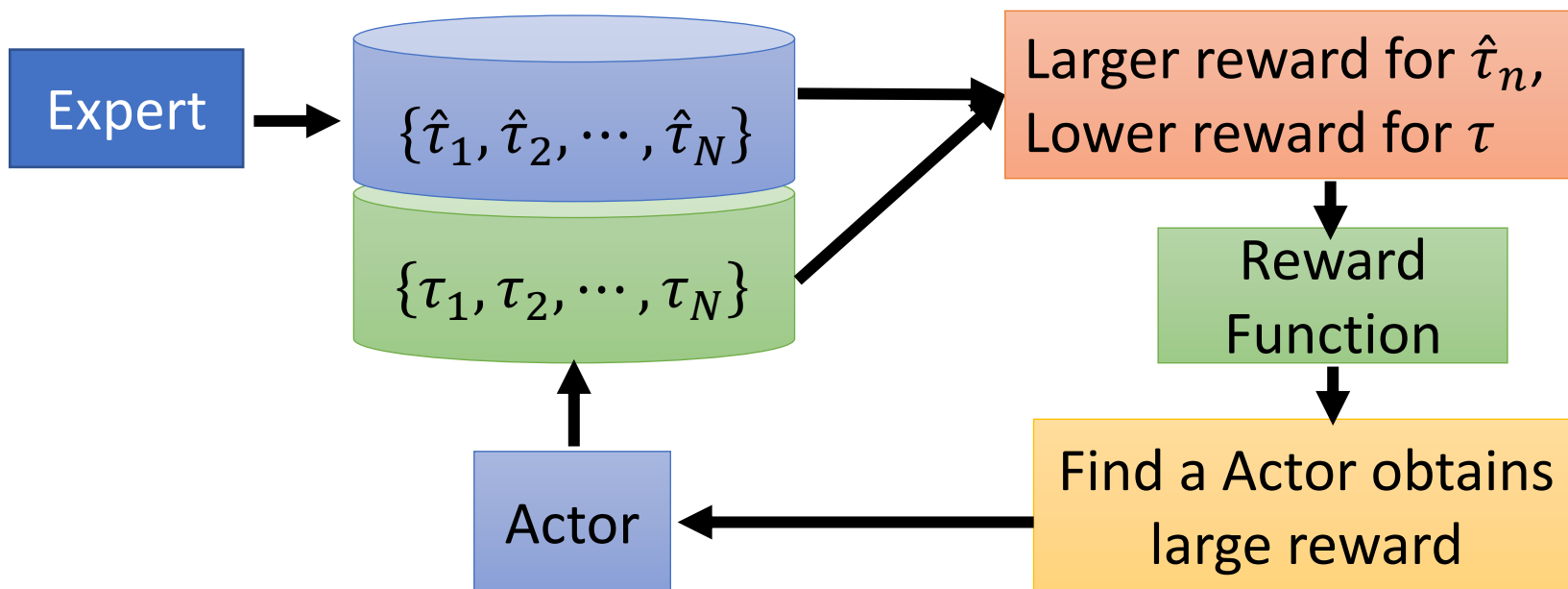
Framework of IRL

$$\sum_{n=1}^{K} R(\hat{\tau}_n) > \sum_{n=1}^{K} R(\tau)$$

Expert $\hat{\pi}$

$\{\hat{\tau}_1, \hat{\tau}_2, \cdots, \hat{\tau}_K\}$

Obtain Reward Function R

Reward Function R

$\{\tau_1, \tau_2, \cdots, \tau_K\}$

Actor = Generator

Reward function = Discriminator

Actor π

Find an actor based on reward function R

By Reinforcement learning

GAN

High score for real, low score for generated

D

Find a G whose output obtains large score from D

G

IRL

Expert

$\{\hat{\tau}_1, \hat{\tau}_2, \cdots, \hat{\tau}_N\}$

$\{\tau_1, \tau_2, \cdots, \tau_N\}$

Larger reward for $\hat{\tau}_n$, Lower reward for τ

Reward Function

Find a Actor obtains large reward

Actor

Robot

Concluding Remarks

What is RL? (Three steps in ML)

Policy Gradient

Actor-Critic

Sparse Reward

No Reward: Learning from Demonstration