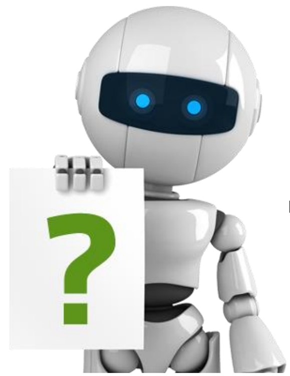
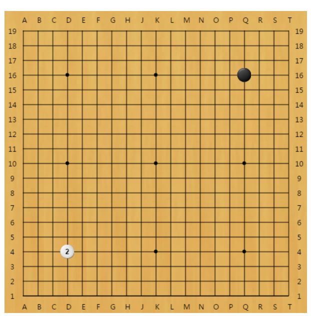
Introduction of Deep Reinforcement Learning (RL)

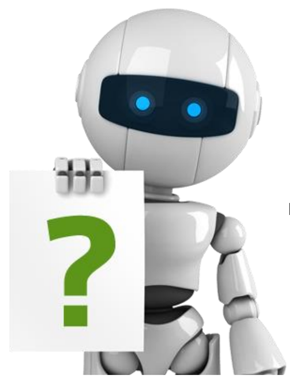
Hung-yi Lee

Supervised Learning → RL



? ⟷ Human label Cat





? ⟷ Human label "3-3"?

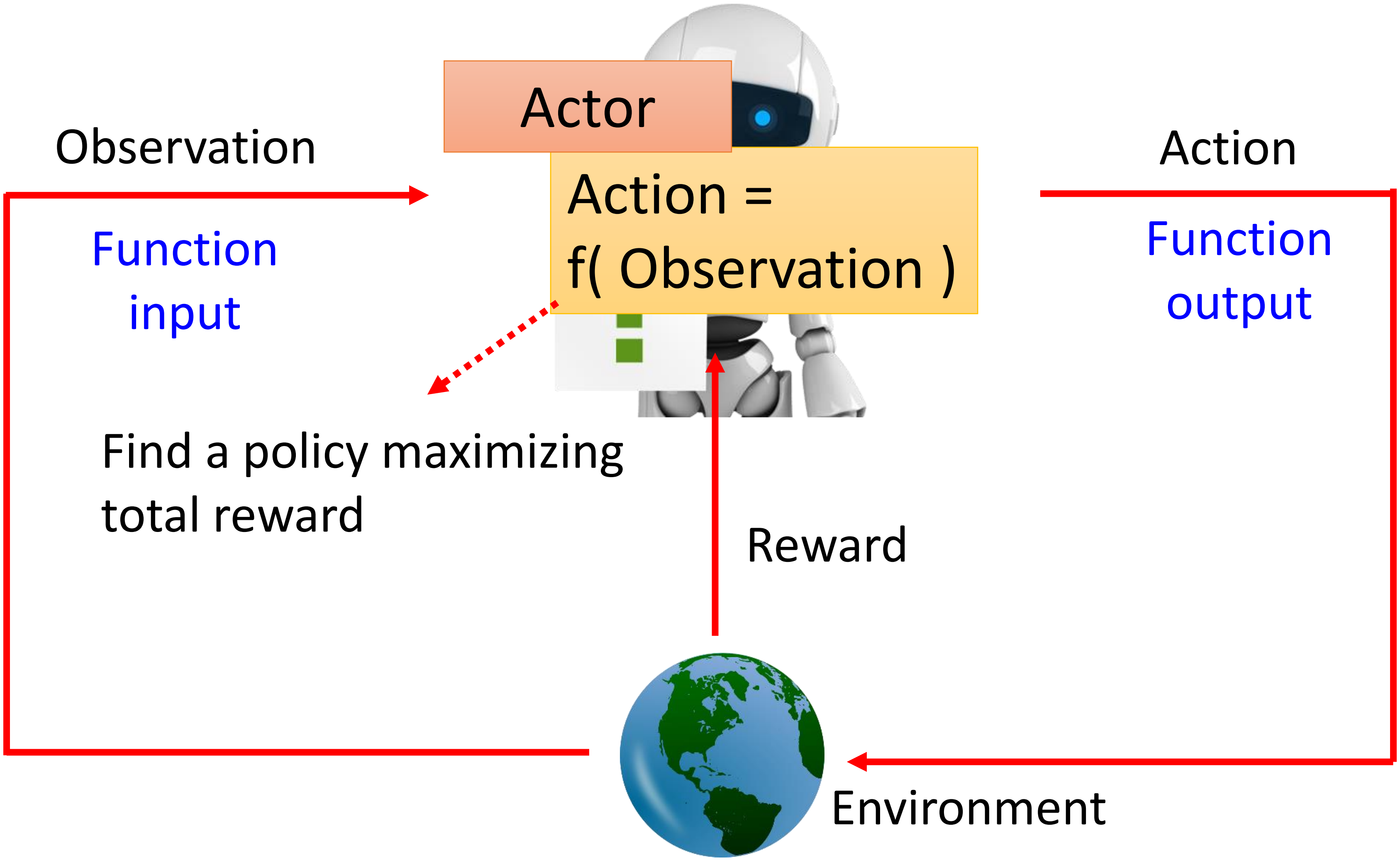
It is challenging to label data in some tasks.

…… machine can know the results are good or not.

Outline

- What is RL? (Three steps in ML)
- Policy Gradient
- Actor-Critic
- Reward Shaping
- No Reward: Learning from Demonstration

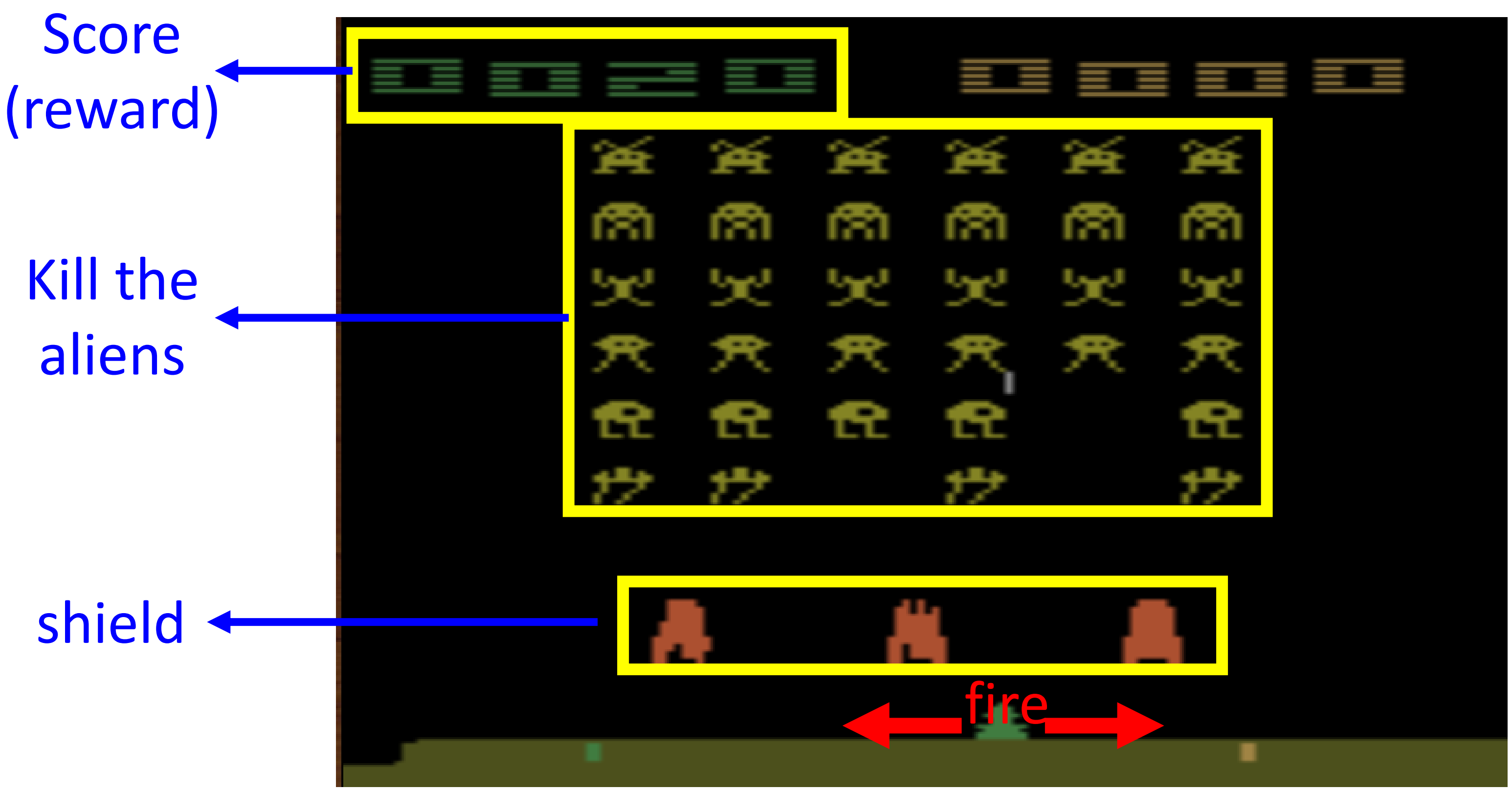
Machine Learning
≈ Looking for a Function



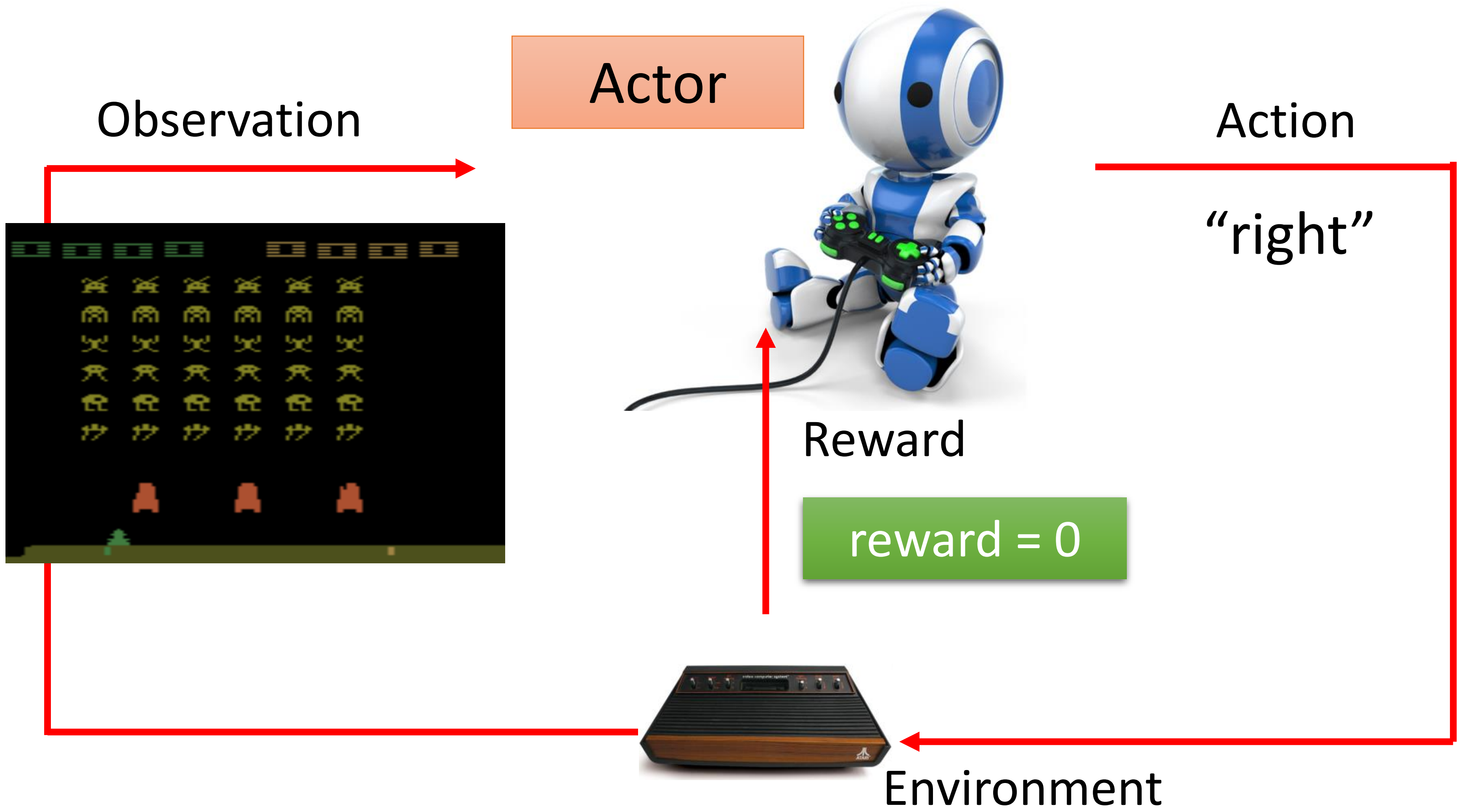
Example: Playing Video Game

- Space invader

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

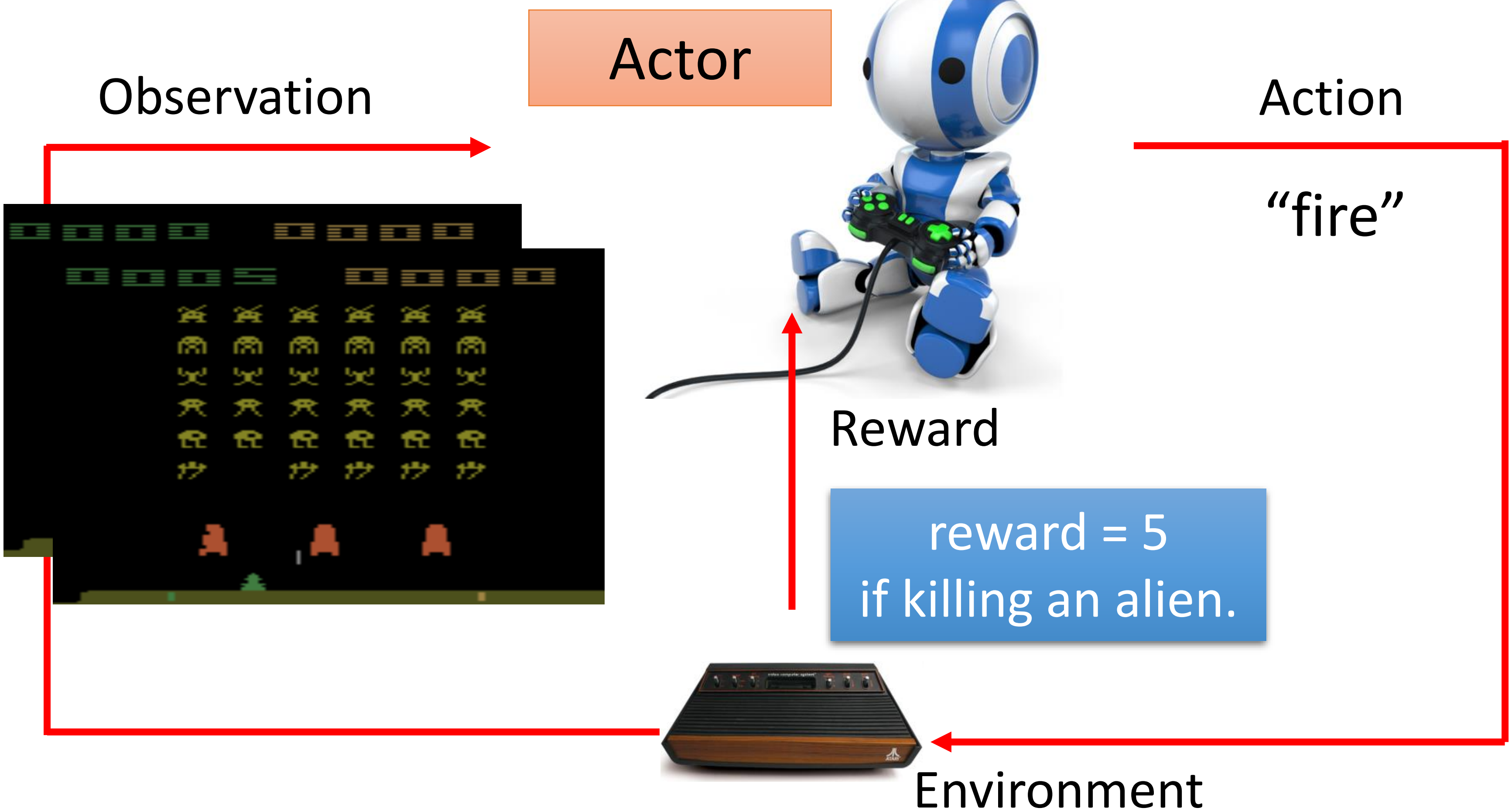


Example: Playing Video Game

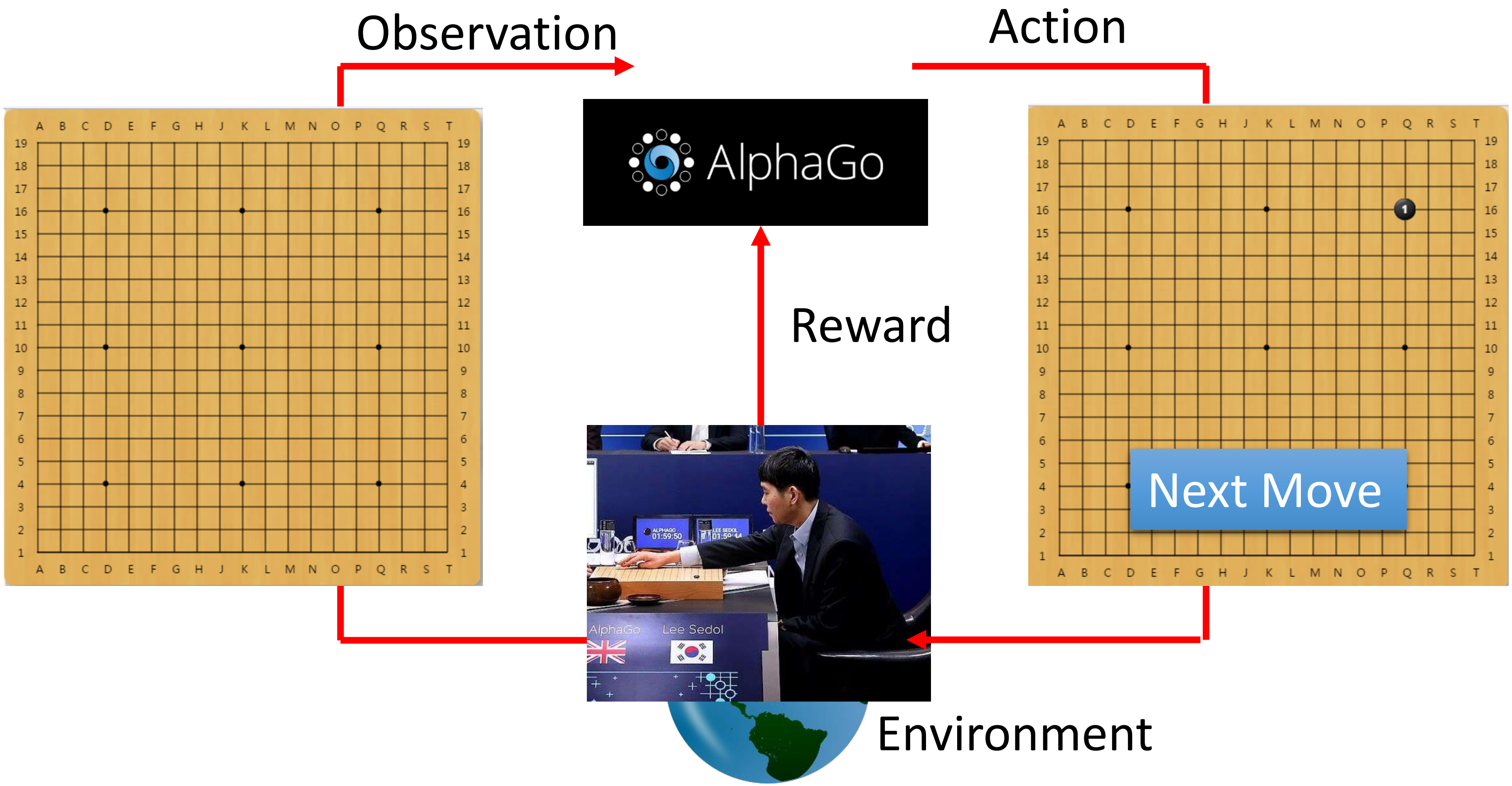


Example: Playing Video Game

Find an actor maximizing expected reward.

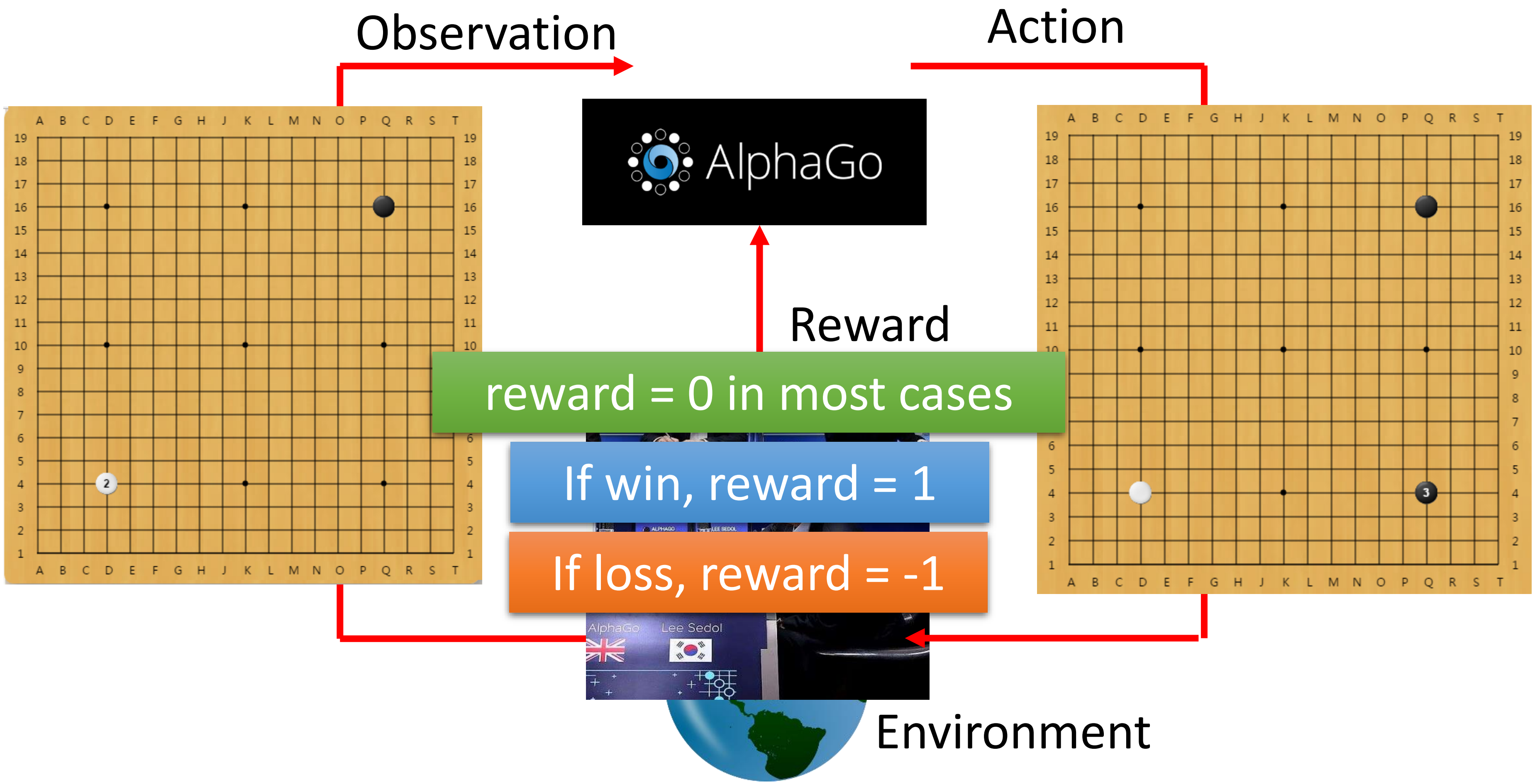


Example: Learning to play Go



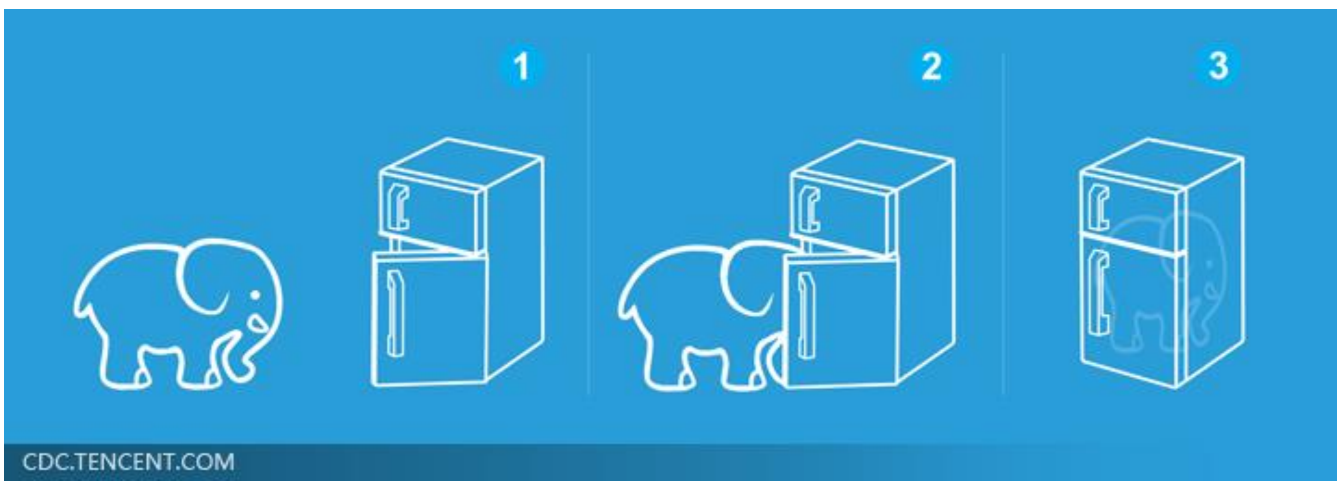
Example: Learning to play Go

Find an actor maximizing expected reward.

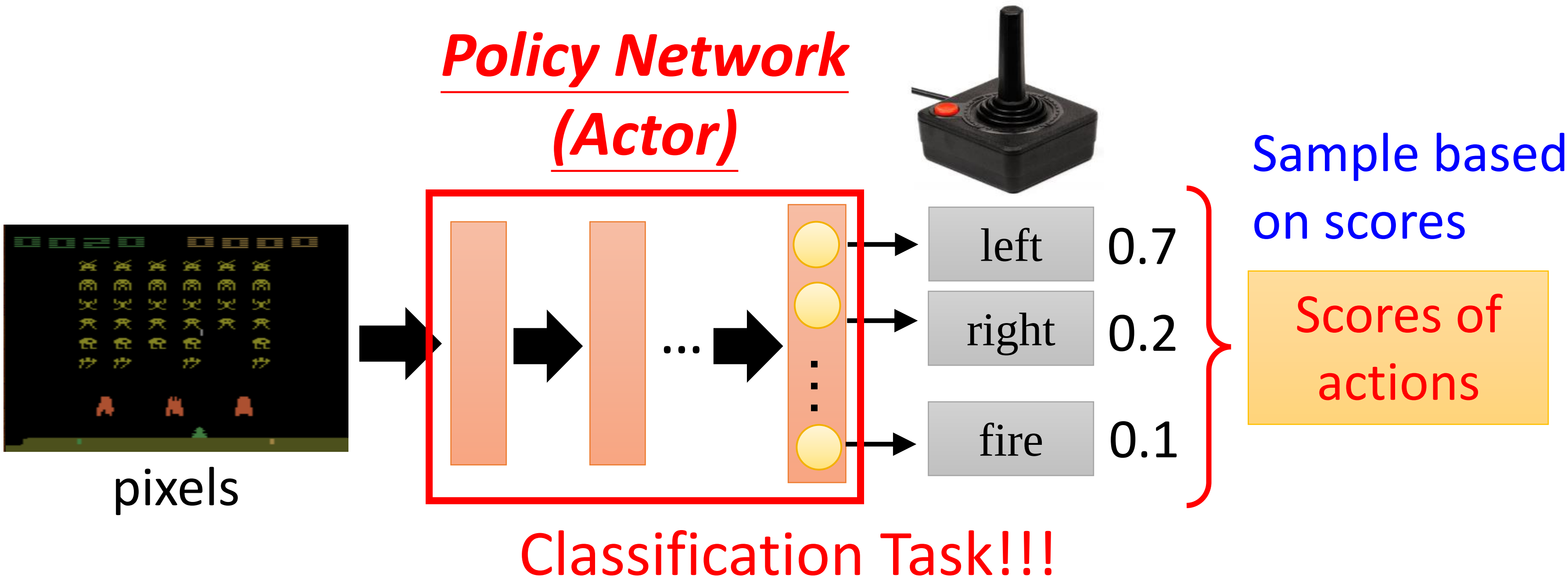


Machine Learning is so simple ……

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

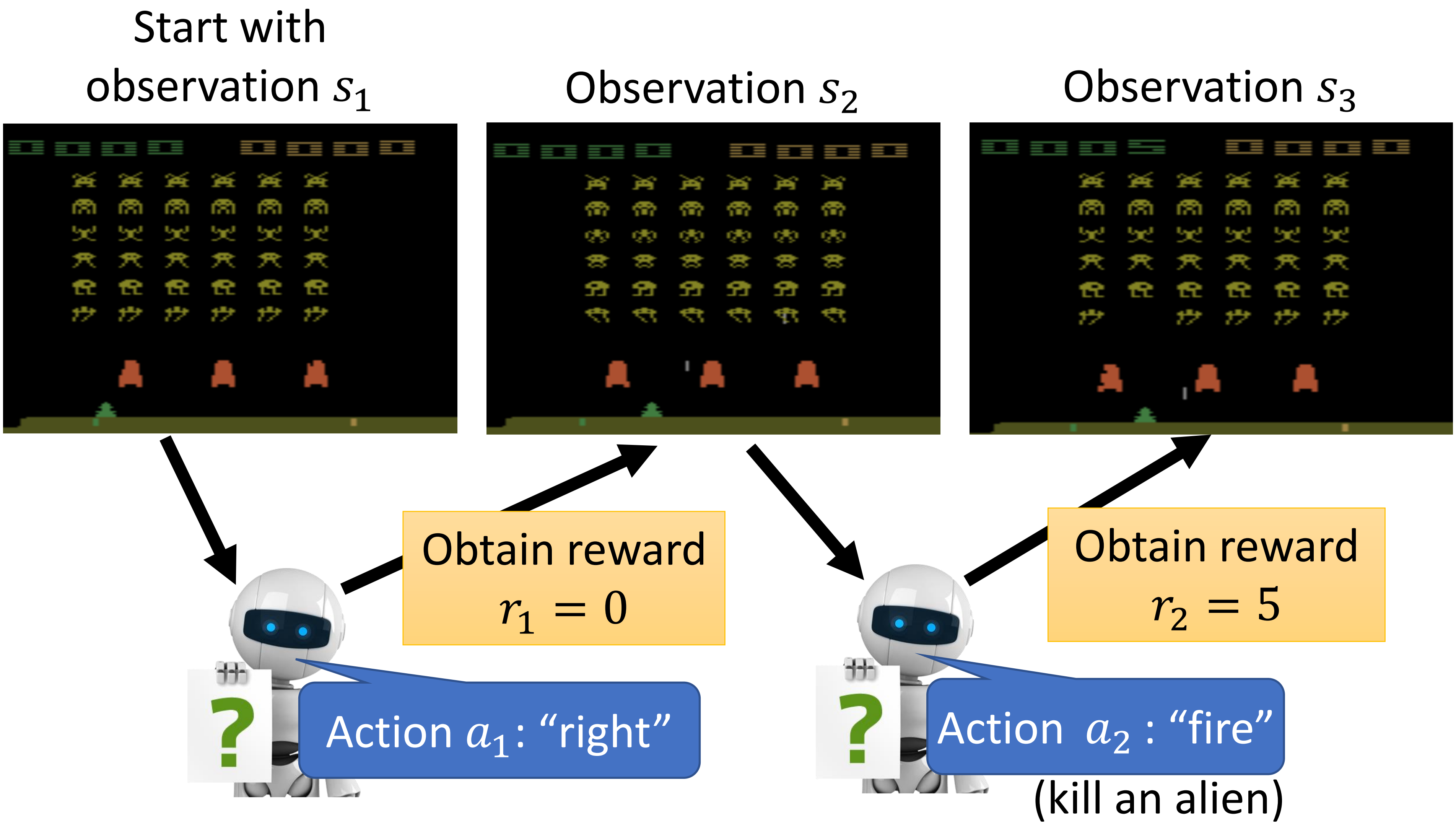


Step 1: Function with Unknown



- 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

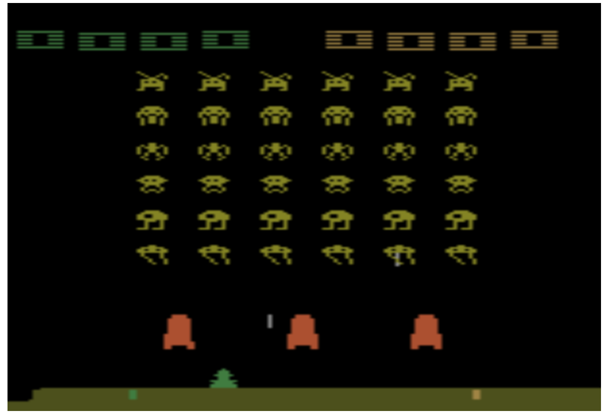
Step 2: Define "Loss"



Step 2: Define "Loss"

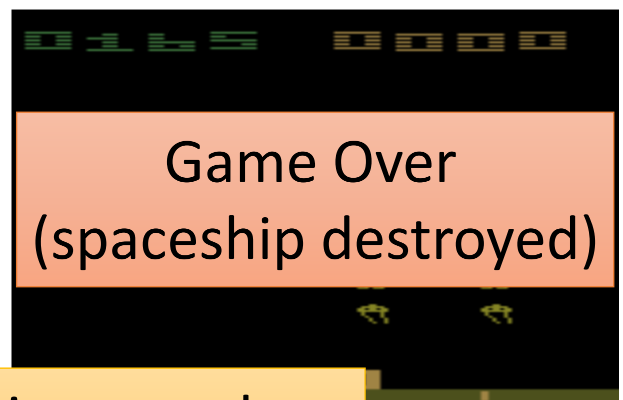
Start with observation s_1

Observation s_2



Observation s_3

After many turns



Obtain reward r_T

Action a_T

This is an ***episode***.

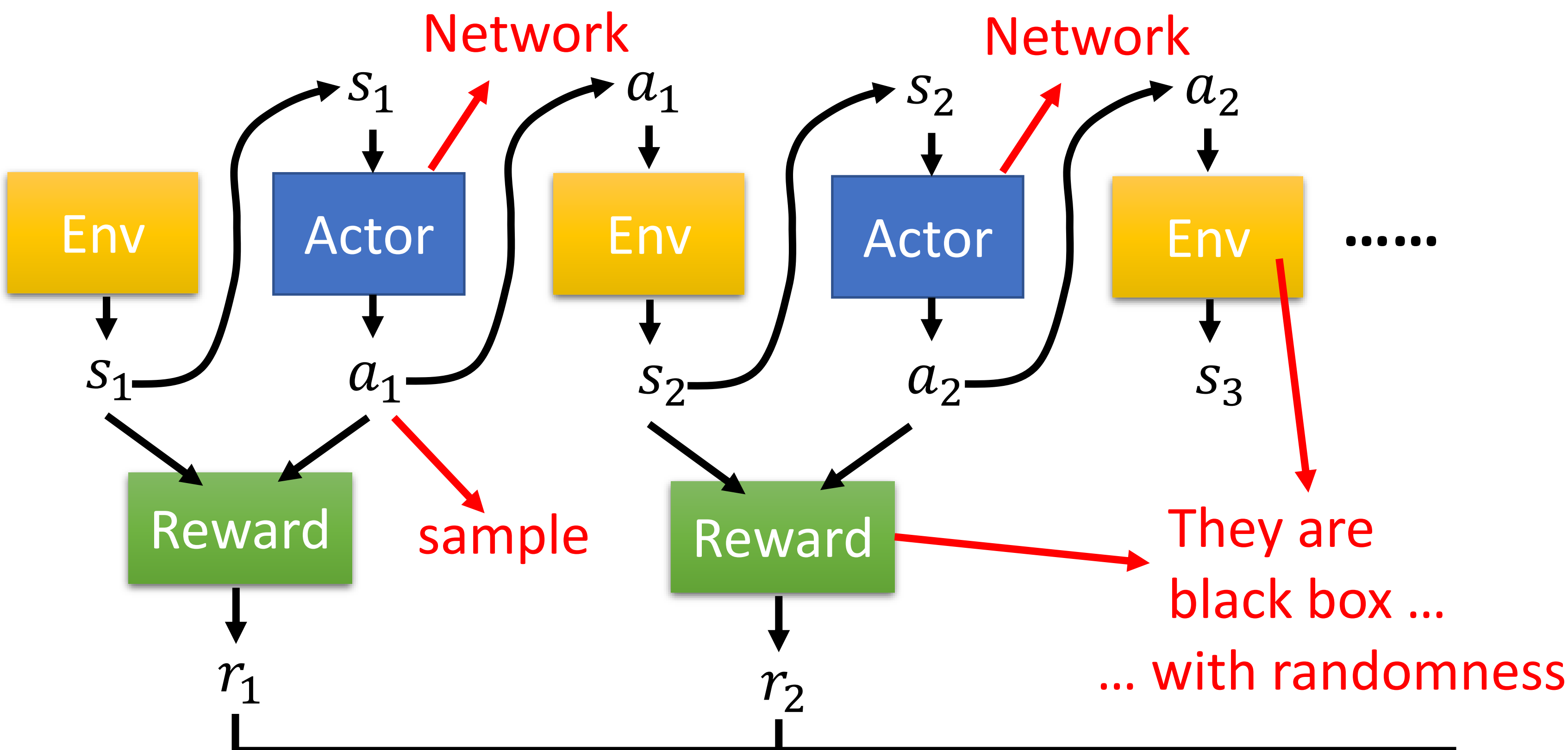
Total reward (*return*): $R = \sum_{t=1}^{T} r_t$

What we want to maximize

Step 3: Optimization

Trajectory

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



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

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

c.f. GAN

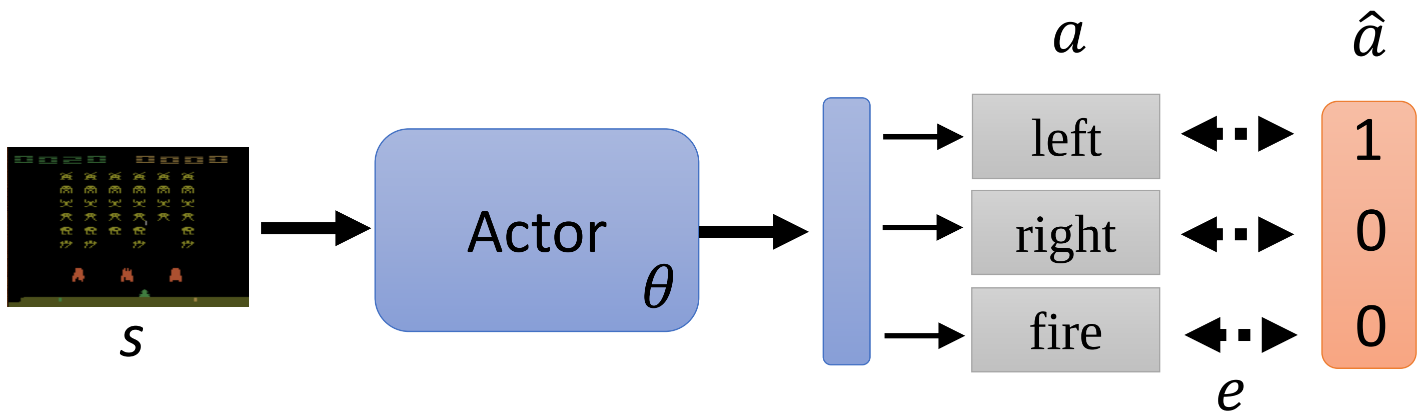
Outline

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 $\hat{a}$

$$L = e$$

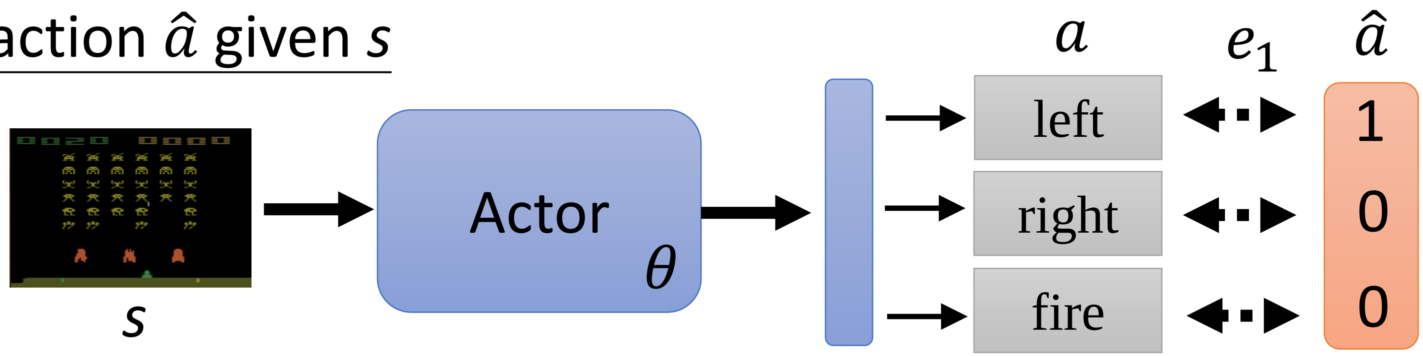
Don't take action $\hat{a}$

$$L = -e$$

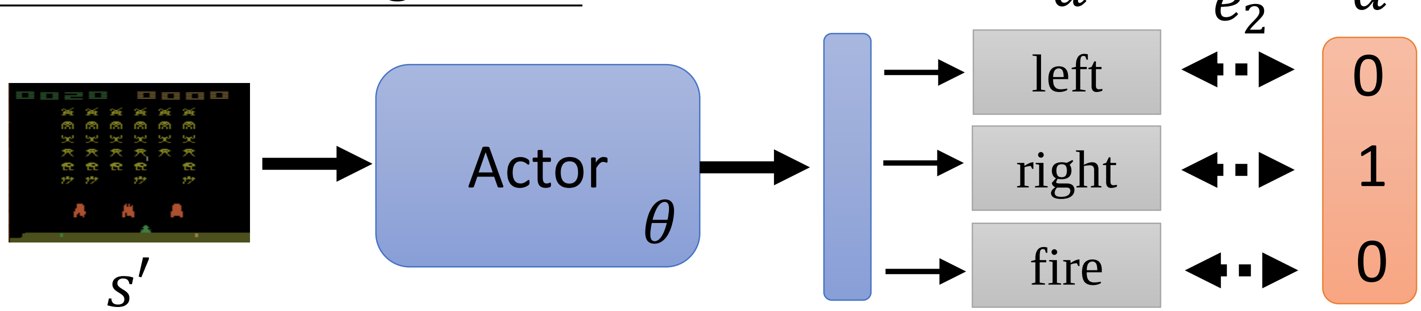
$$\theta^* = arg \min_{\theta} L$$

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

$$L = +e_1 - e_2 + e_3 \cdots - e_{\mathrm{N}}$$

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

How to control your actor

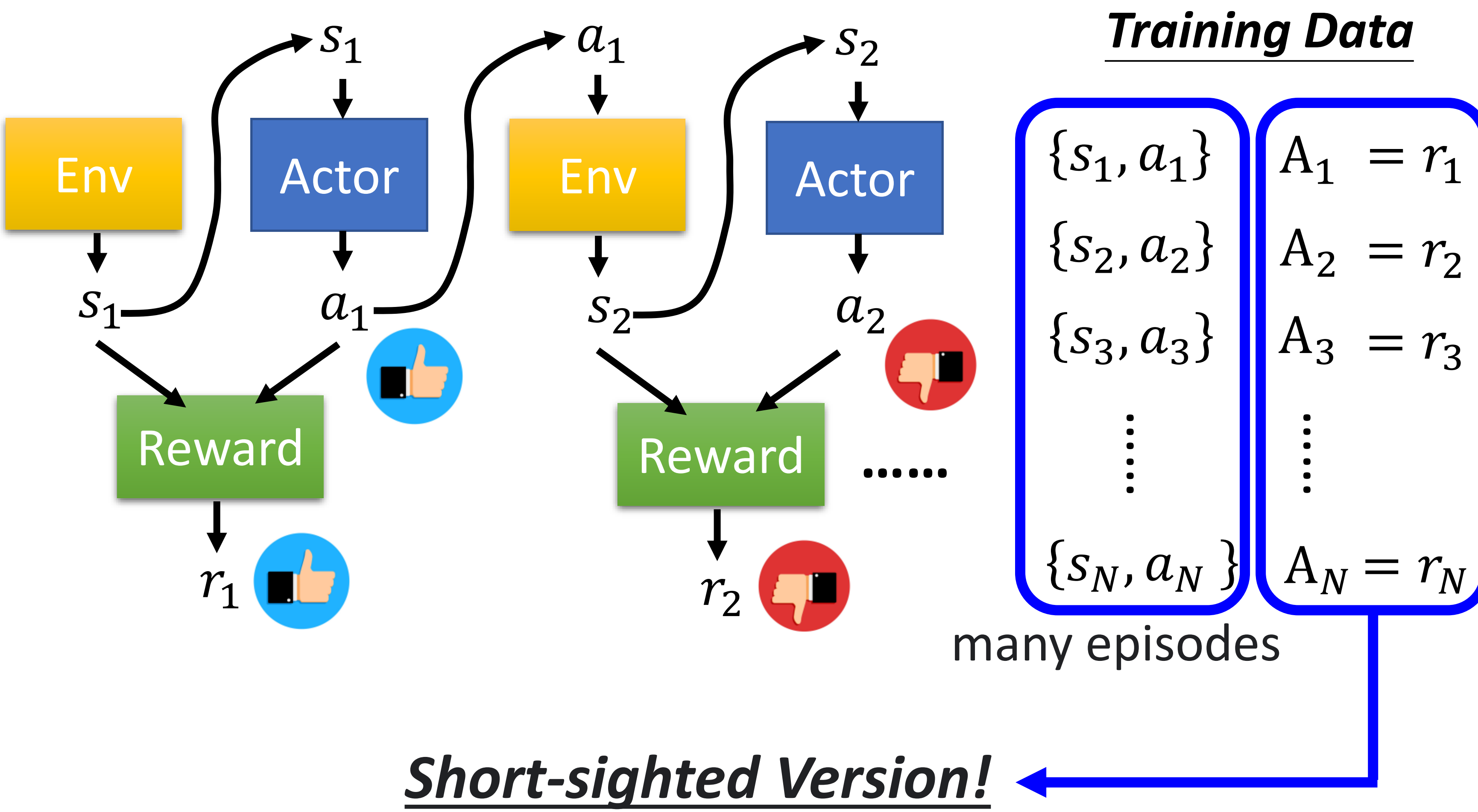
Training Data

$\{s_1, \hat{a}_1\}$	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
?	?	

$$L = \sum A_n e_n$$

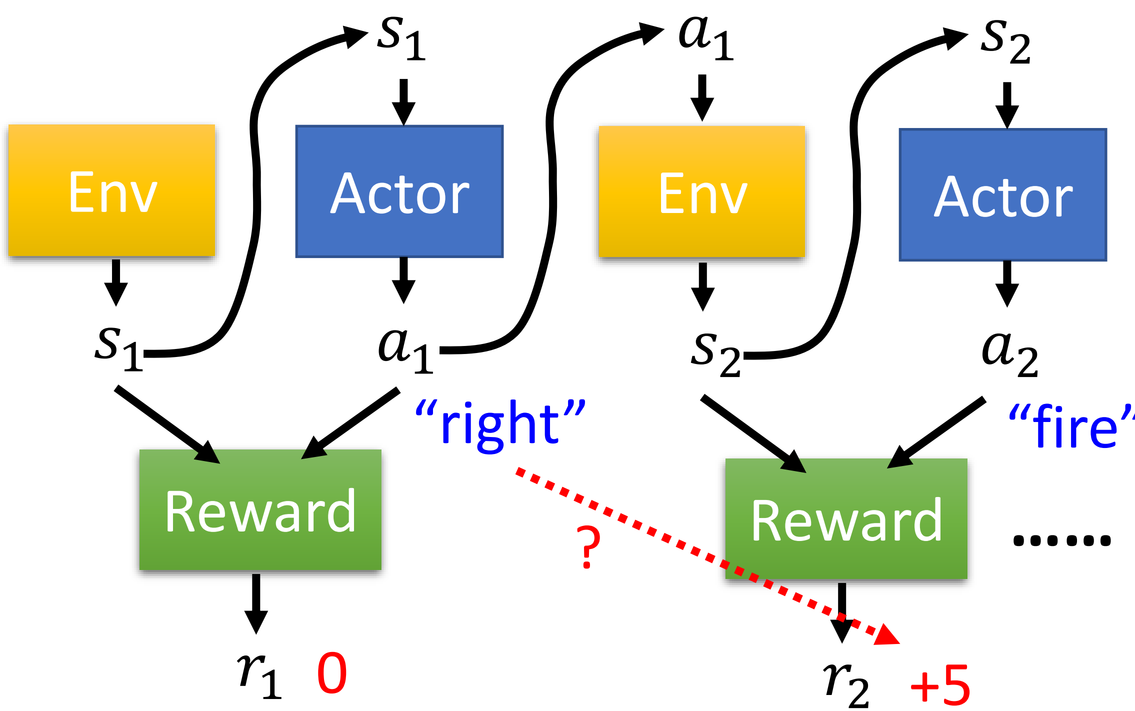
$$\theta^* = arg \min_{\theta} L$$

Version 0



Env
Actor
Env
Actor
s_1
a_1
s_2
a_2
Reward
Reward
r_1
r_2
......
Training Data
$\{s_1, a_1\}$
$\{s_2, a_2\}$
$\{s_3, a_3\}$
$\{s_N, a_N\}$
$A_1 = r_1$
$A_2 = r_2$
$A_3 = r_3$
$A_N = r_N$
many episodes
Short-sighted Version!

Version 0



- An action affects 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" yields positive reward, so vision 0 will learn an actor that always "fire".

Version 1

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

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

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

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

cumulated reward

Training Data

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

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

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

$$\vdots \qquad \vdots$$

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

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

Version 2

s_1	s_2	s_3		s_N
$\boxed{a_1}$	$\boxed{a_2}$	$\boxed{a_3}$	......	a_N
r_1	r_2	r_3		r_N

Also the credit of a_1 ?

$$G_1 = r_1 + r_2 + r_3 + \ldots\ldots + \boxed{r_N}$$

$$G'_1 = r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots\ldots$$

Discount factor $\gamma < 1$

Training Data

$\{s_1, a_1\}$ $\quad A_1 = G'_1$

$\{s_2, a_2\}$ $\quad A_2 = G'_2$

$\{s_3, a_3\}$ $\quad A_3 = G'_3$

$\vdots \qquad \vdots$

$\{s_N, a_N\}$ $\quad A_N = G'_N$

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

Version 3

s_1 s_2 s_3 s_N

$\boxed{a_1}$ $\boxed{a_2}$ $\boxed{a_3}$ …… a_N

r_1 r_2 r_3 r_N

Good or bad reward is "relative"

If all the $r_n \geq 10$

$r_n = 10$ is negative …

Minus by a baseline b ???

Make G'_t have positive and negative values

Training Data

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

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

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

⋮ ⋮

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

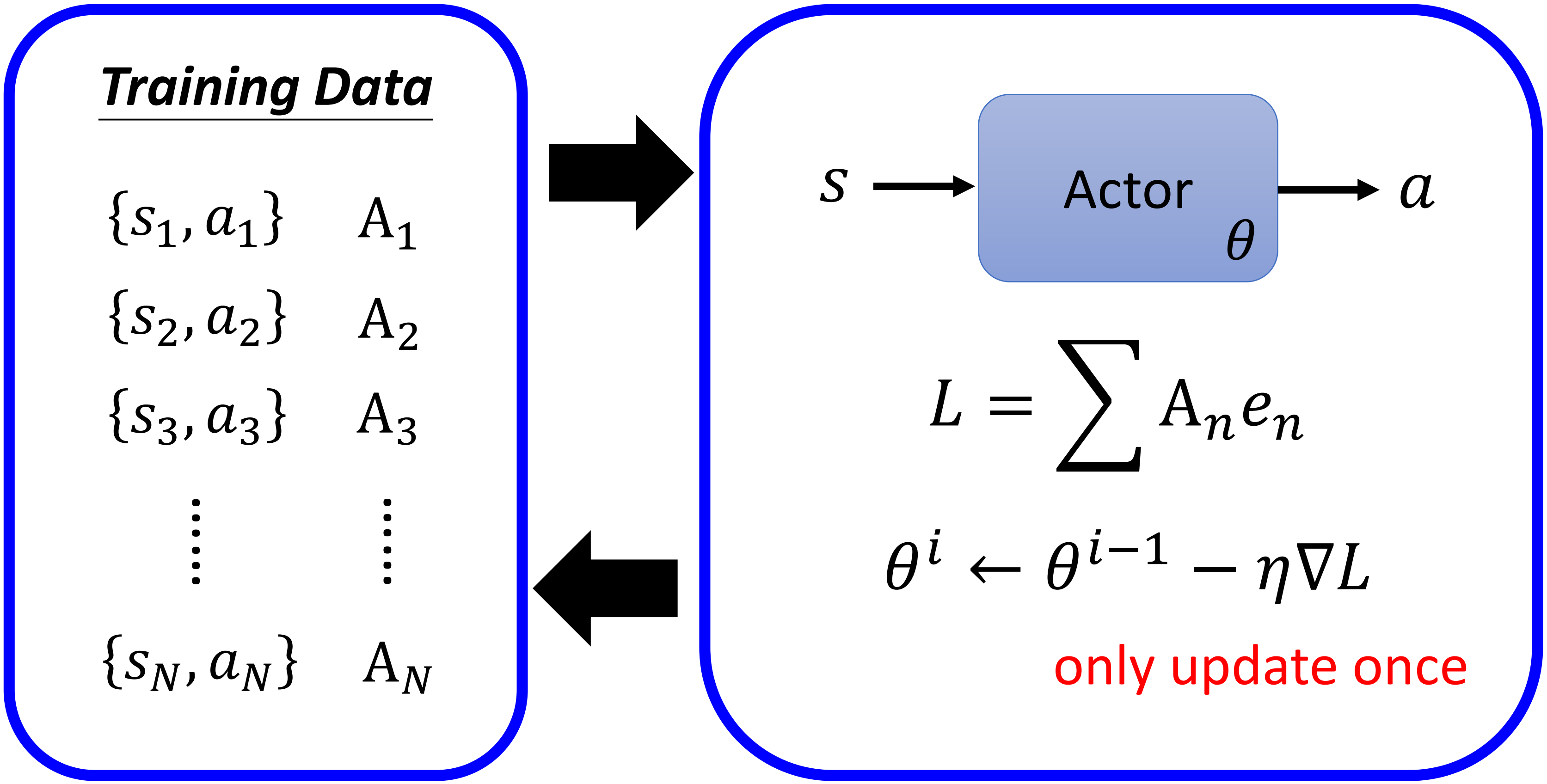
$$G'_t = \sum_{n=t}^{N} \gamma^{n-t} r_n$$

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

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

Policy Gradient



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 $A_1, A_2, \dots, A_N$
  - Compute loss L
  - $\boxed{\theta^i} \leftarrow \boxed{\theta^{i-1}} - \eta \nabla L$

Experience of θ^{i-1}

May not be good for θ^i

One man's meat is another man's poison.

棋魂第八集


啪噠！
阿光，這時候不要下小馬步飛，而是要大馬步飛。
※ 小馬步飛：跟將棋一樣，將棋子放在斜一格；大馬步飛則是放在斜好幾格。
大馬？
我覺得小馬也不錯啊！

棋魂第八集

相對地，大馬步飛之後的下法會很複雜，出錯率也相對地提高。

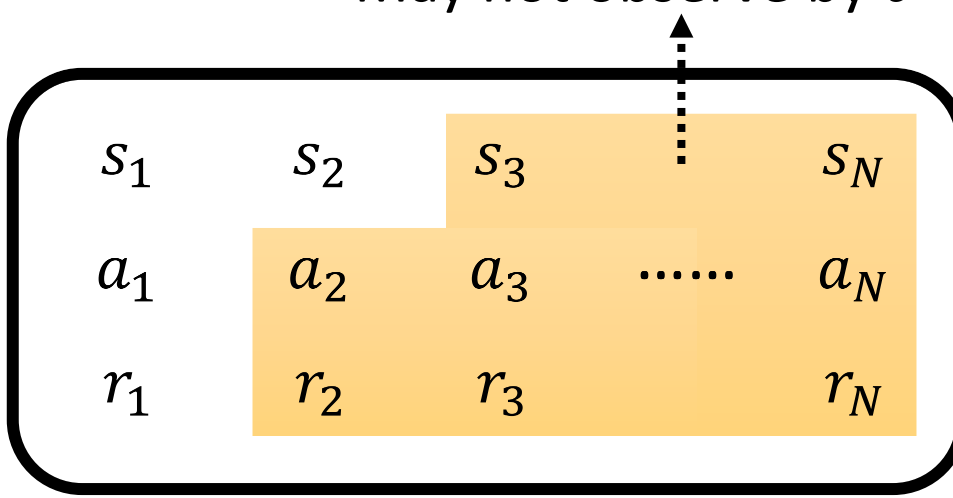
要是大馬步飛有100手的話，小馬步飛就只有99手。
之前走小馬步飛是對的。
因為小馬步飛的後續比較容易推測，也不容易出錯。

Policy Gradient

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

May not observe by θ^i

Trajectory of θ^{i-1}



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}

s_1	s_2	s_3		s_N
a_1	a_2	a_3	……	a_N
r_1	r_2	r_3		r_N

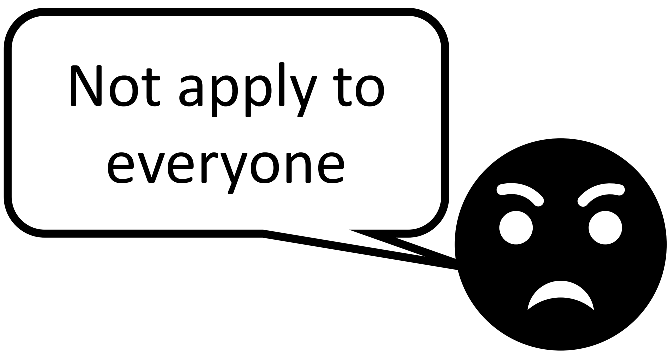
To train θ^i

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

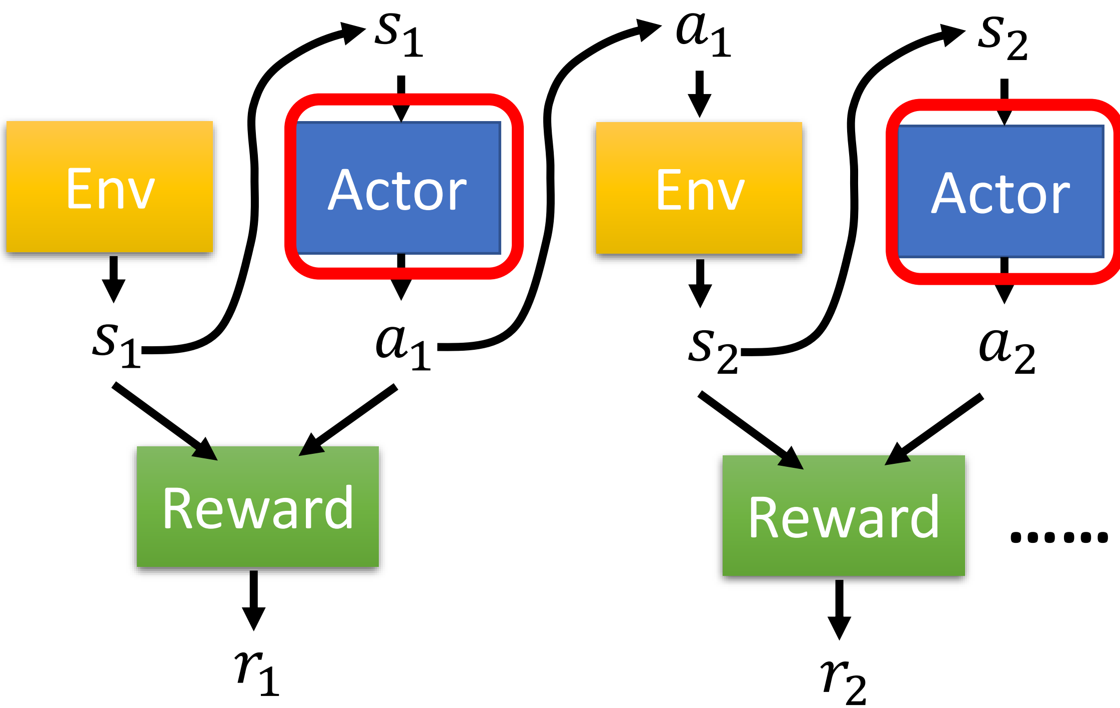


the actor to train

the actor to interact

https://disp.cc/b/115-bLHe

Collection Training Data: **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.

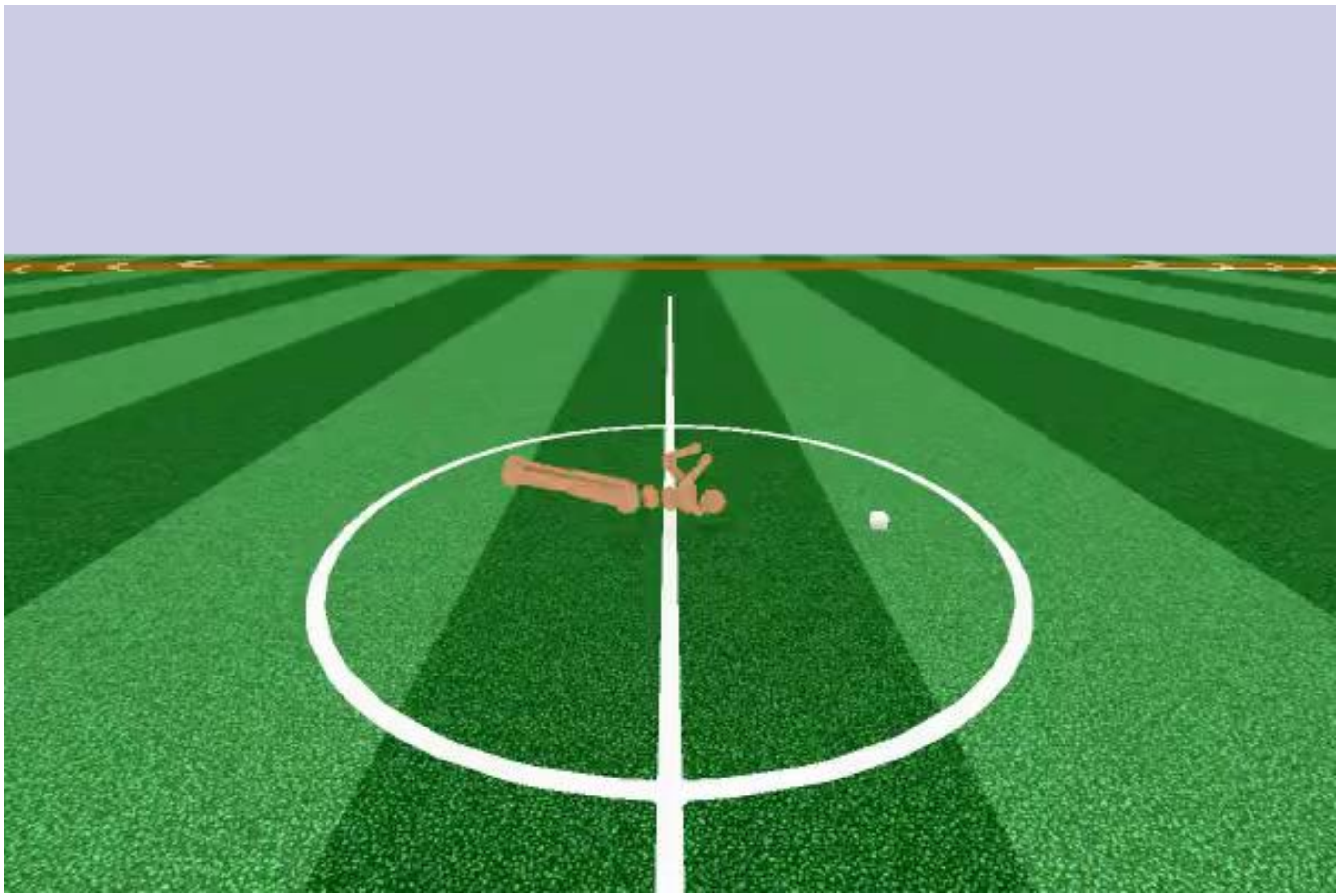
A major reason why we sample actions. ☺

DeepMind - PPO

https://youtu.be/gn4nRCC9TwQ

OpenAI - PPO

https://blog.openai.com/openai-baselines-ppo/



Outline

What is RL? (Three steps in ML)

Policy Gradient

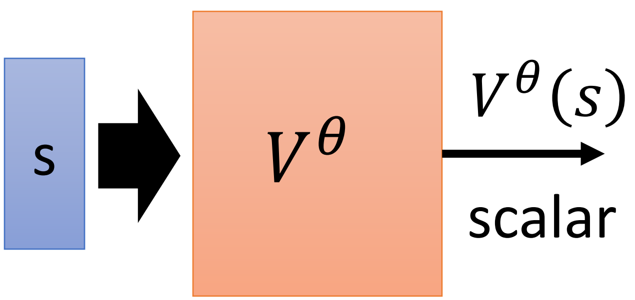
Actor-Critic

Reward Shaping

No Reward: Learning from Demonstration

Critic

- Critic: Given an actor θ, it evaluates how good the actor θ is
- Value function $V^{\boxed{\theta}}(s)$
  - When using actor θ, the discounted *cumulated* reward expects to be obtained after seeing s



$V^\theta(s)$ is large

$V^\theta(s)$ is smaller

The output values of a critic depend on the actor evaluated.

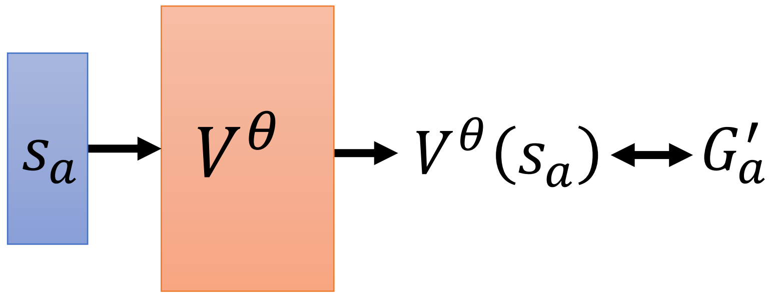
How to estimate $V^{\theta}(s)$

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

The critic watches actor θ to interact with the environment.

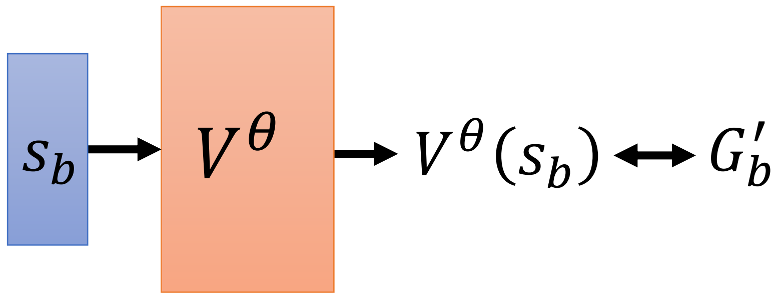
After seeing s_a,

Until the end of the episode, the cumulated reward is G'_a



After seeing s_b,

Until the end of the episode, the cumulated reward is G'_b



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

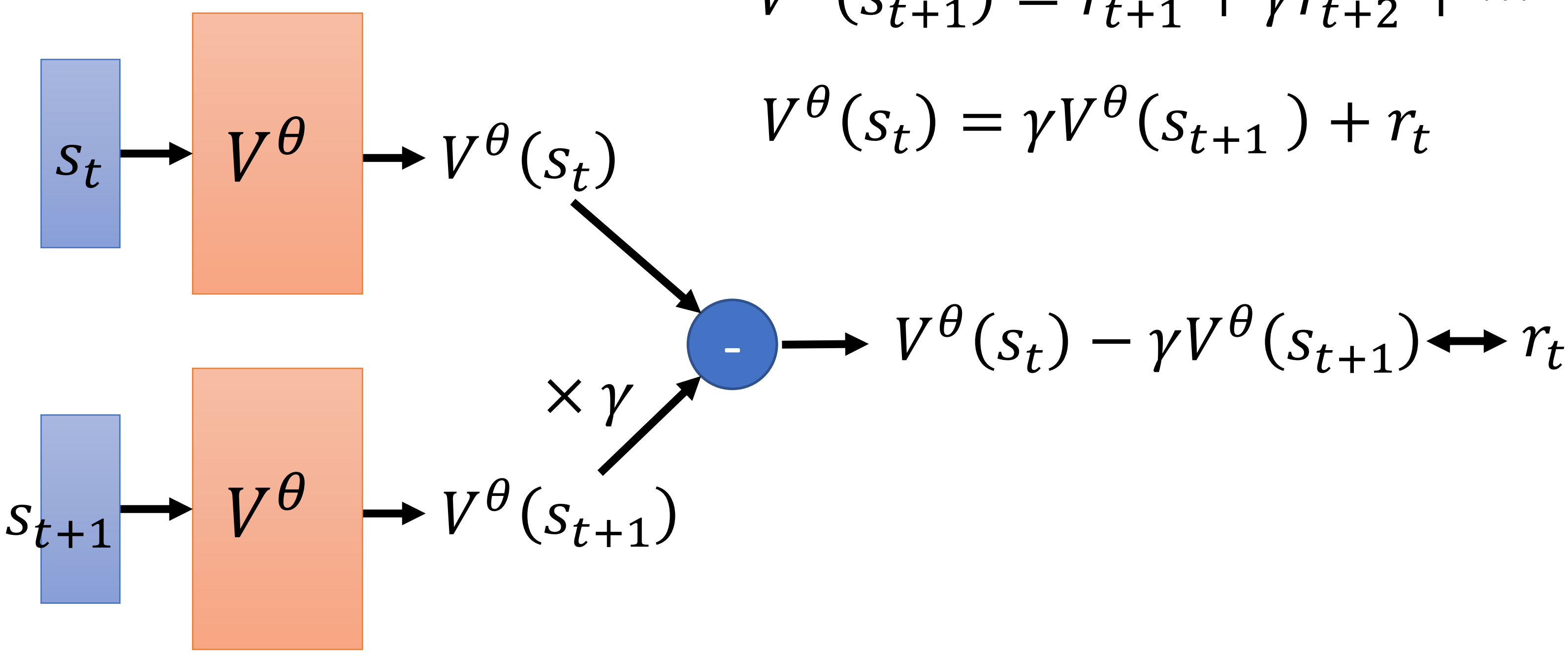
- **Temporal-difference (TD) approach**

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

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

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

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



MC v.s. TD

[Sutton, v2, Example 6.4]

- 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

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

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

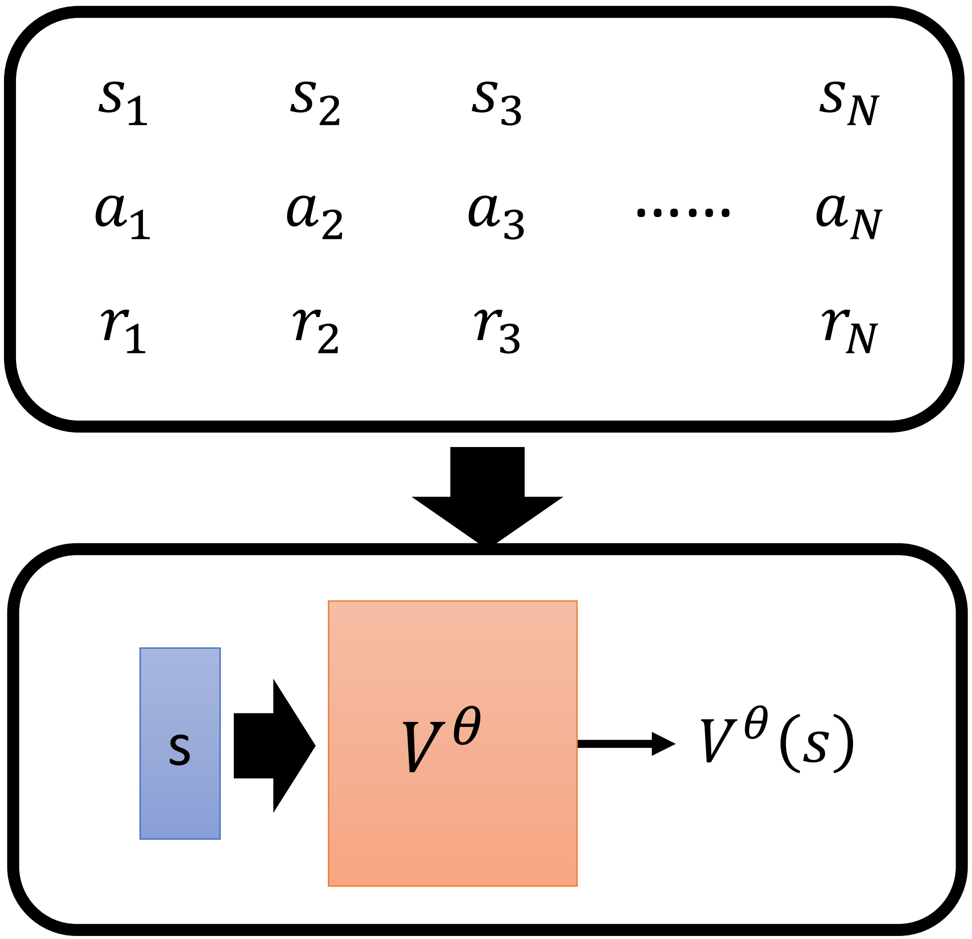
Monte-Carlo: $V^\theta(s_a) = 0$

Temporal-difference:

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

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

Version 3.5



Training Data

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

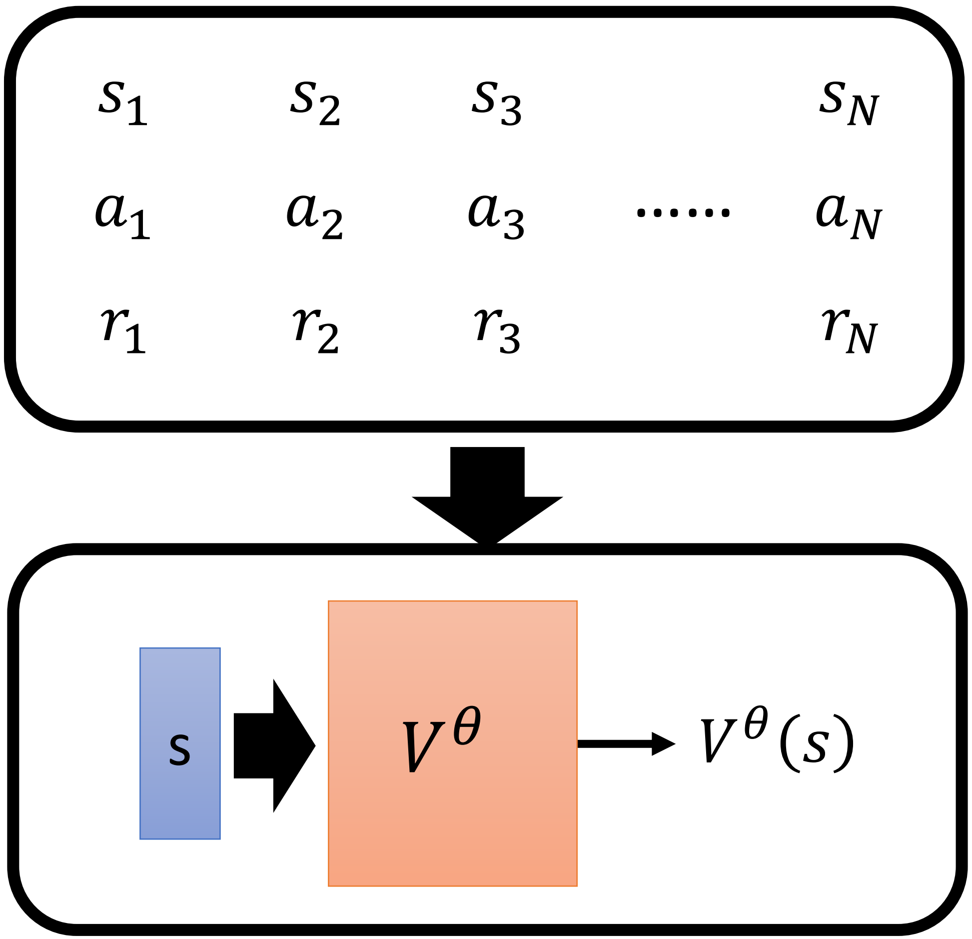
$\{s_2, a_2\}$ $\quad A_2 = G'_2 - b$

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

$\vdots \qquad \vdots$

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

Version 3.5



Training Data

$\{s_1, a_1\}$ $\quad A_1 = G'_1 - V^\theta(s_1)$

$\{s_2, a_2\}$ $\quad A_2 = G'_2 - V^\theta(s_2)$

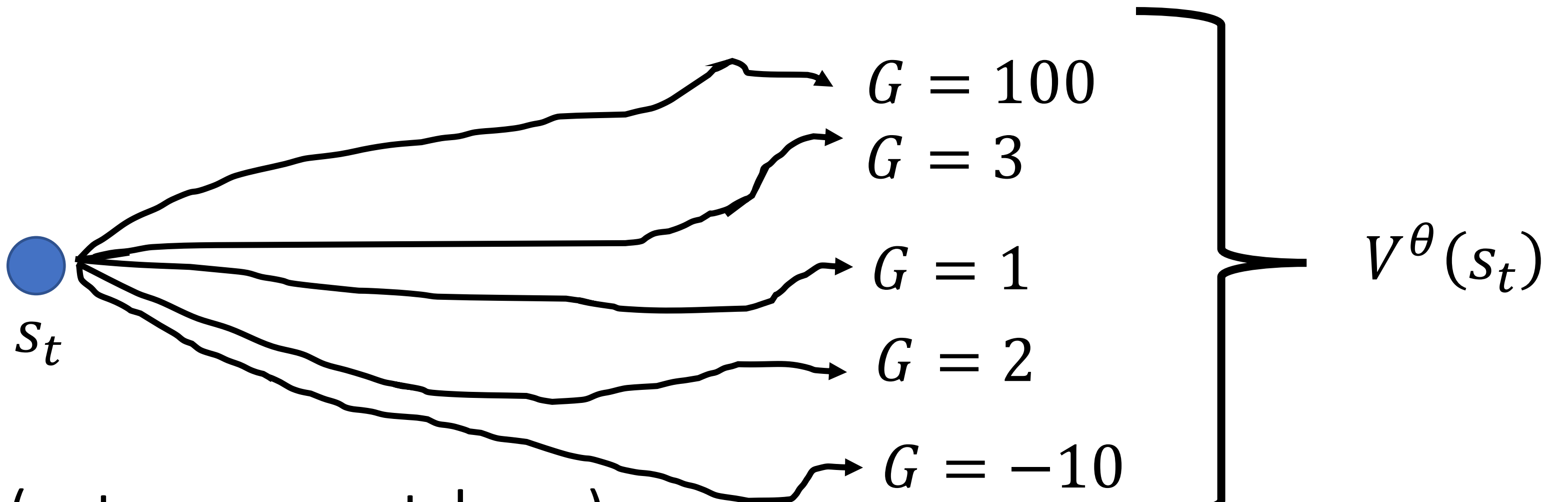
$\{s_3, a_3\}$ $\quad A_3 = G'_3 - V^\theta(s_3)$

$\vdots \qquad \vdots$

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

Version 3.5

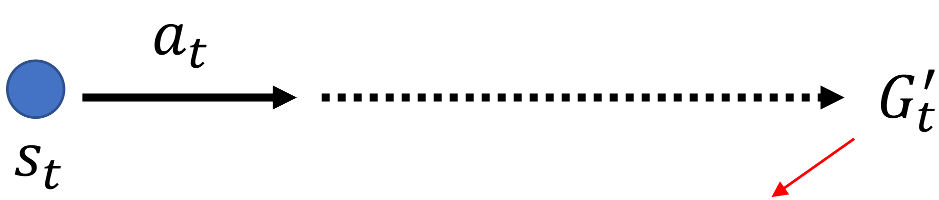
$\{s_t, a_t\}$ $\quad A_t = G'_t - V^\theta(s_t)$



(not necessary take a_t)

$A_t > 0$

a_t is better than average.



Just a sample

$A_t < 0$

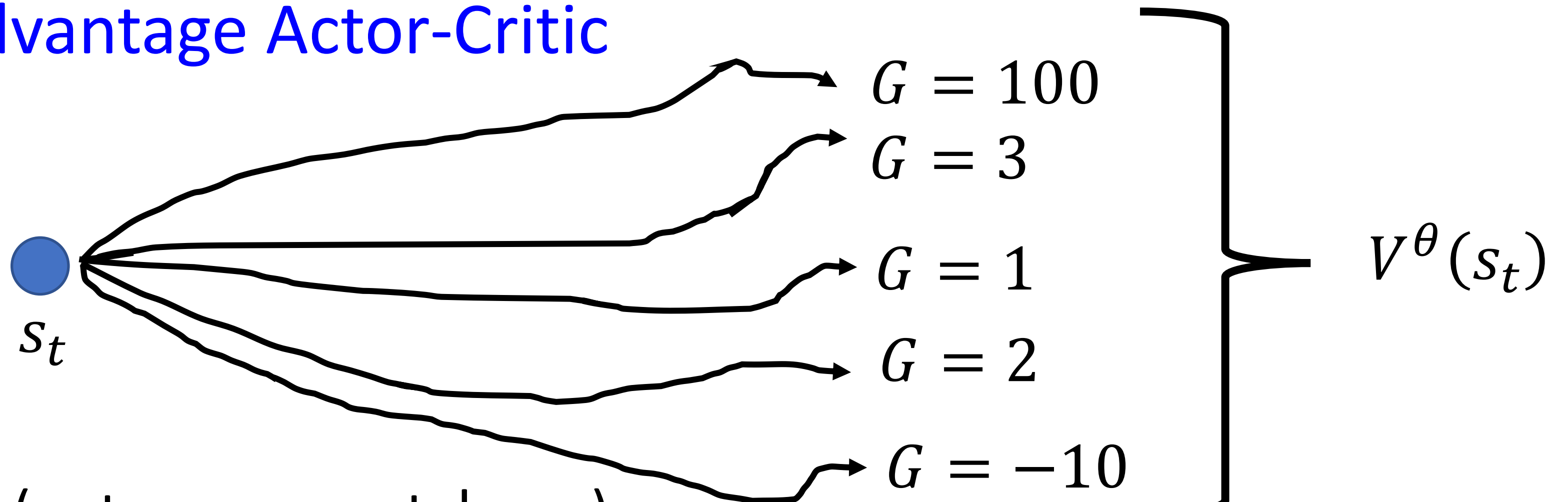
a_t is worse than average.

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

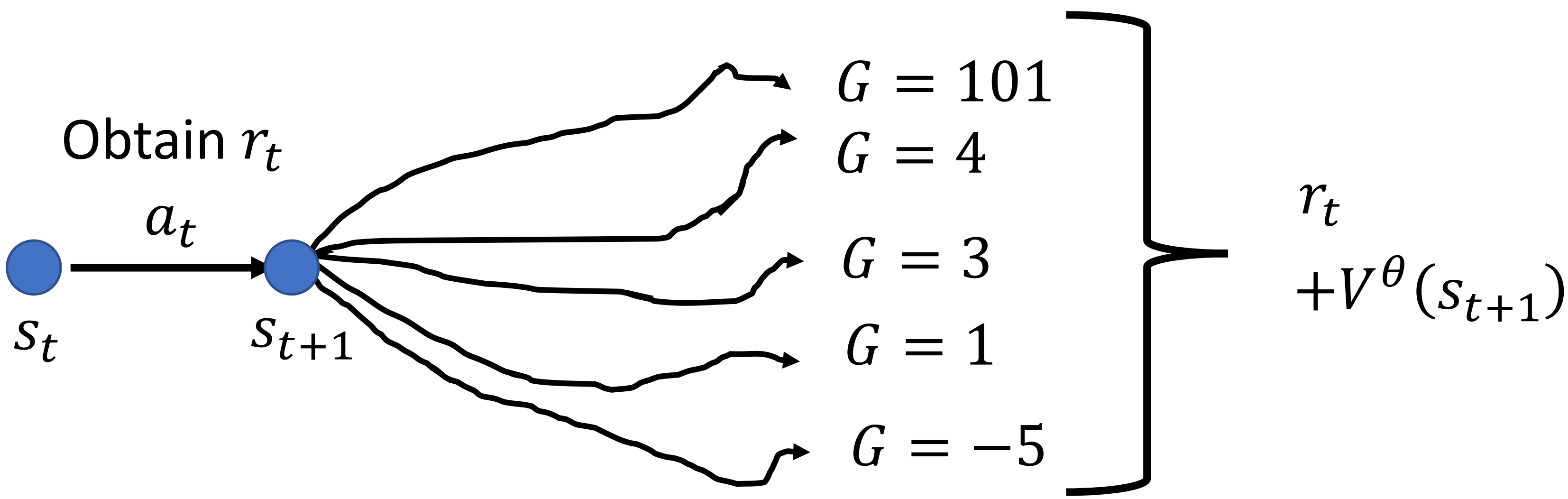
Version 4

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

Advantage Actor-Critic

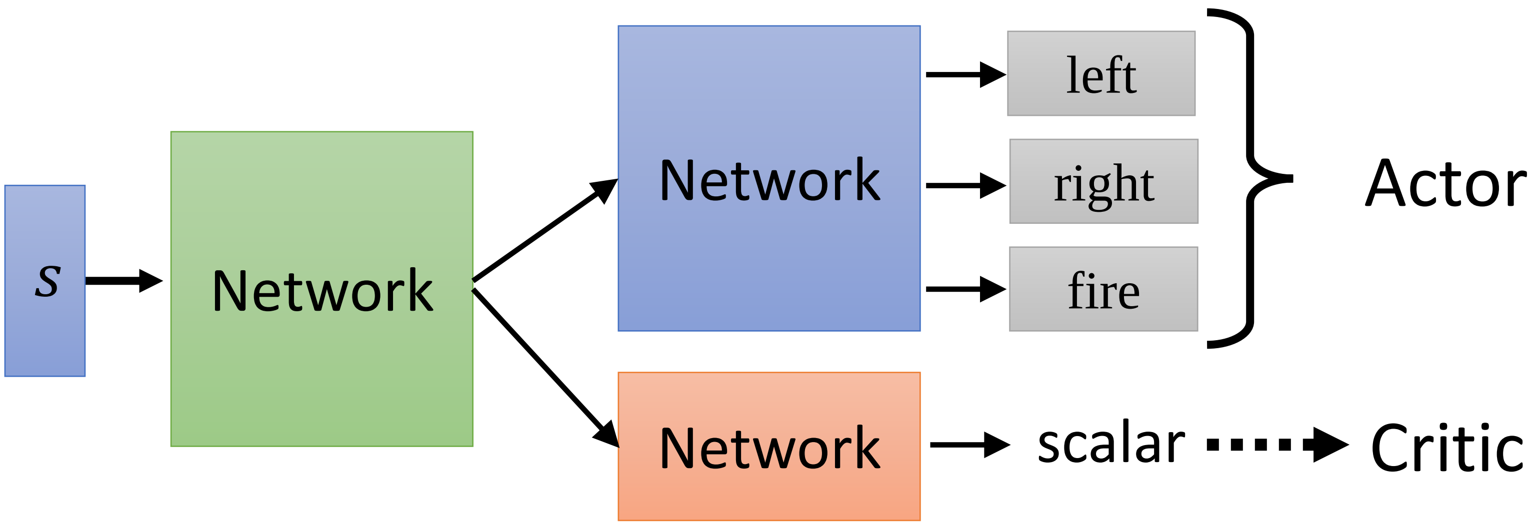


(not necessary take a_t)



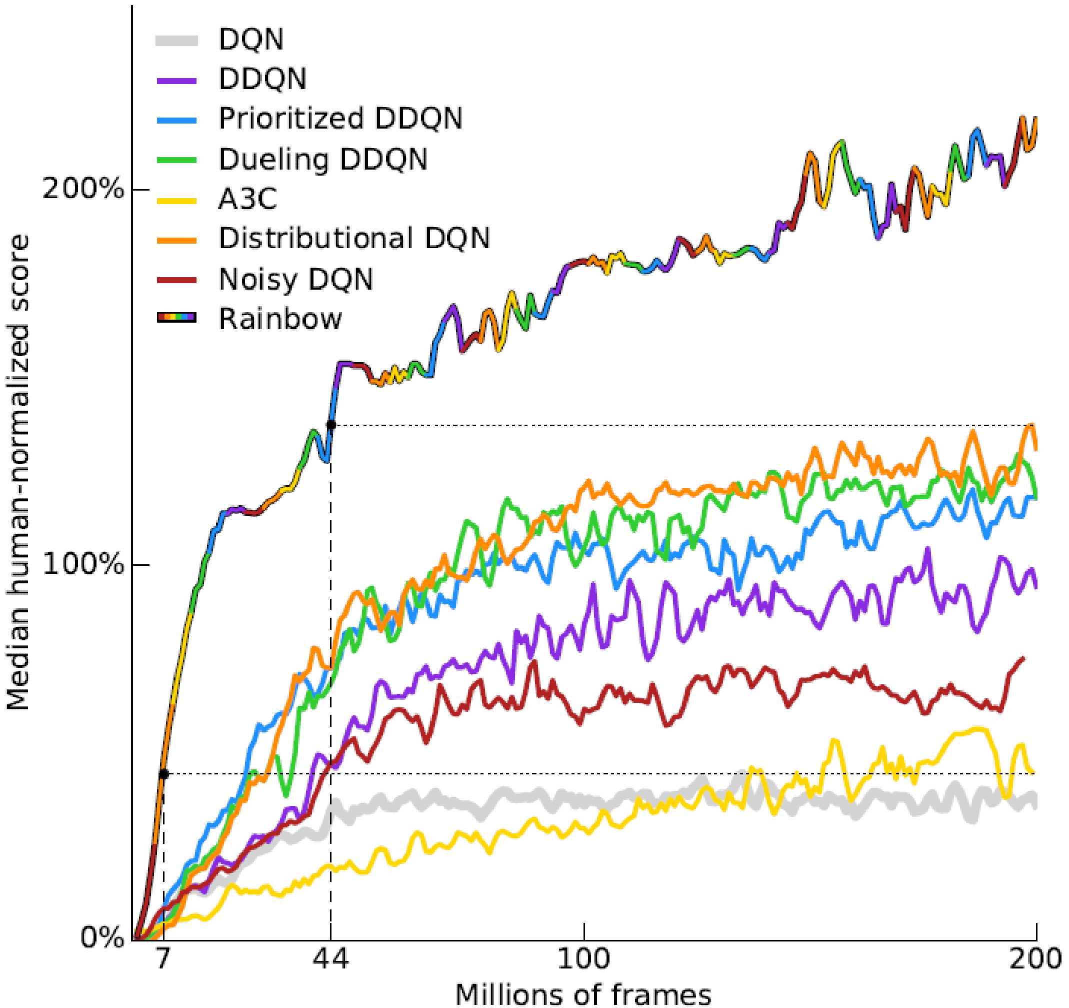
Tip of Actor-Critic

- The parameters of actor and critic can be shared.



Outlook: 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

https://www.techbang.com/posts/41569-5-axis-arduino-mechanical-arm-maximo-do-it-yourself-at-home
https://www.kindori.com.tw/

Reward Shaping

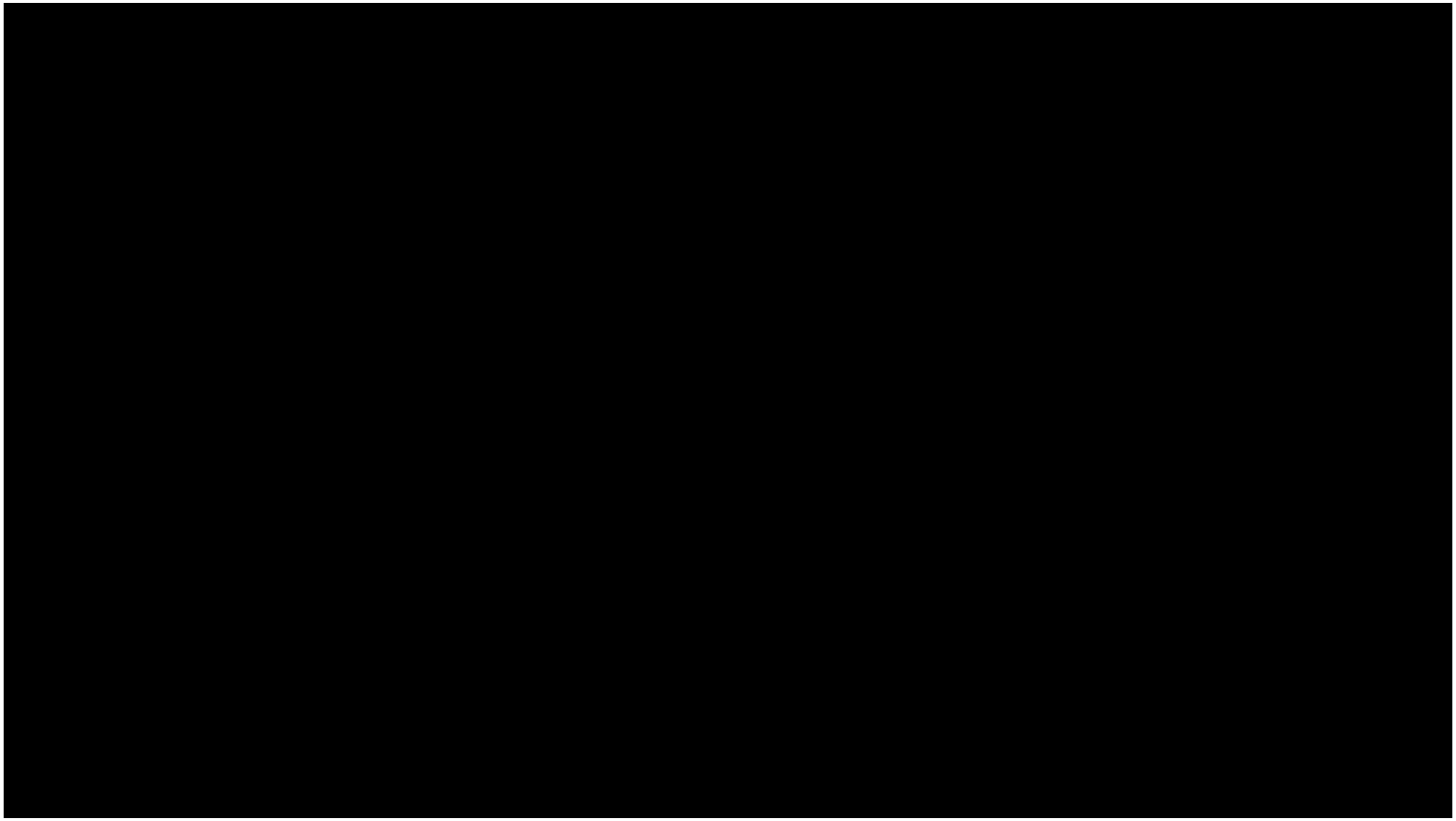
- Reward can be sparse.

$r_t = 0$ in most cases

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

Visual Doom AI Competition @ CIG 2016

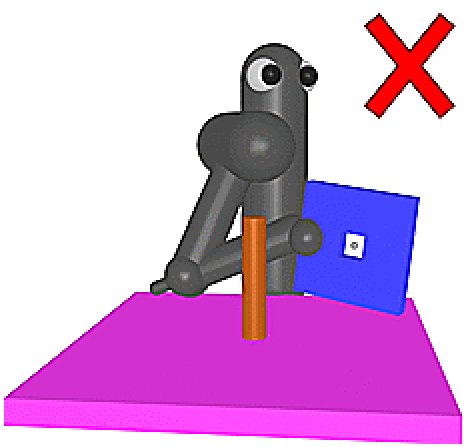
https://www.youtube.com/watch?v=94EPSjQH38Y

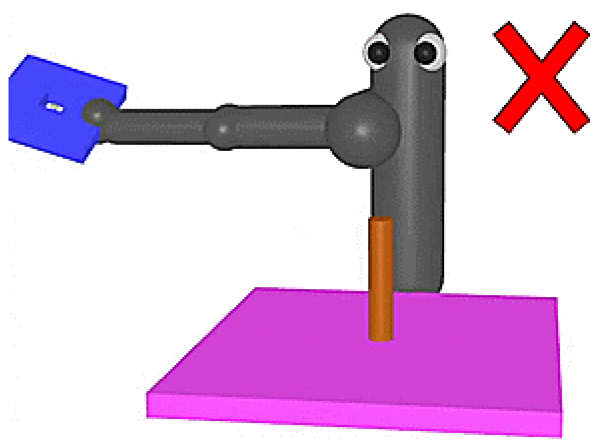


Reward Shaping

VizDoom https://openreview.net/forum?id=Hk3mPK5gg&noteId=Hk3mPK5gg

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

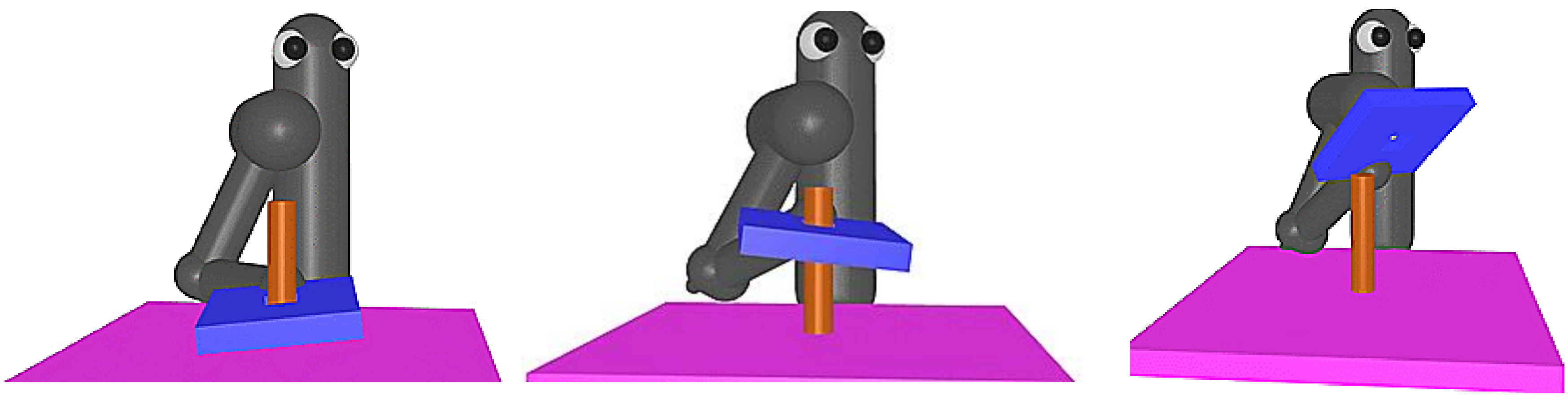
https://bair.berkeley.edu/blog/2017/12/20/reverse-curriculum/

Curriculum Learning

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

VizDoom

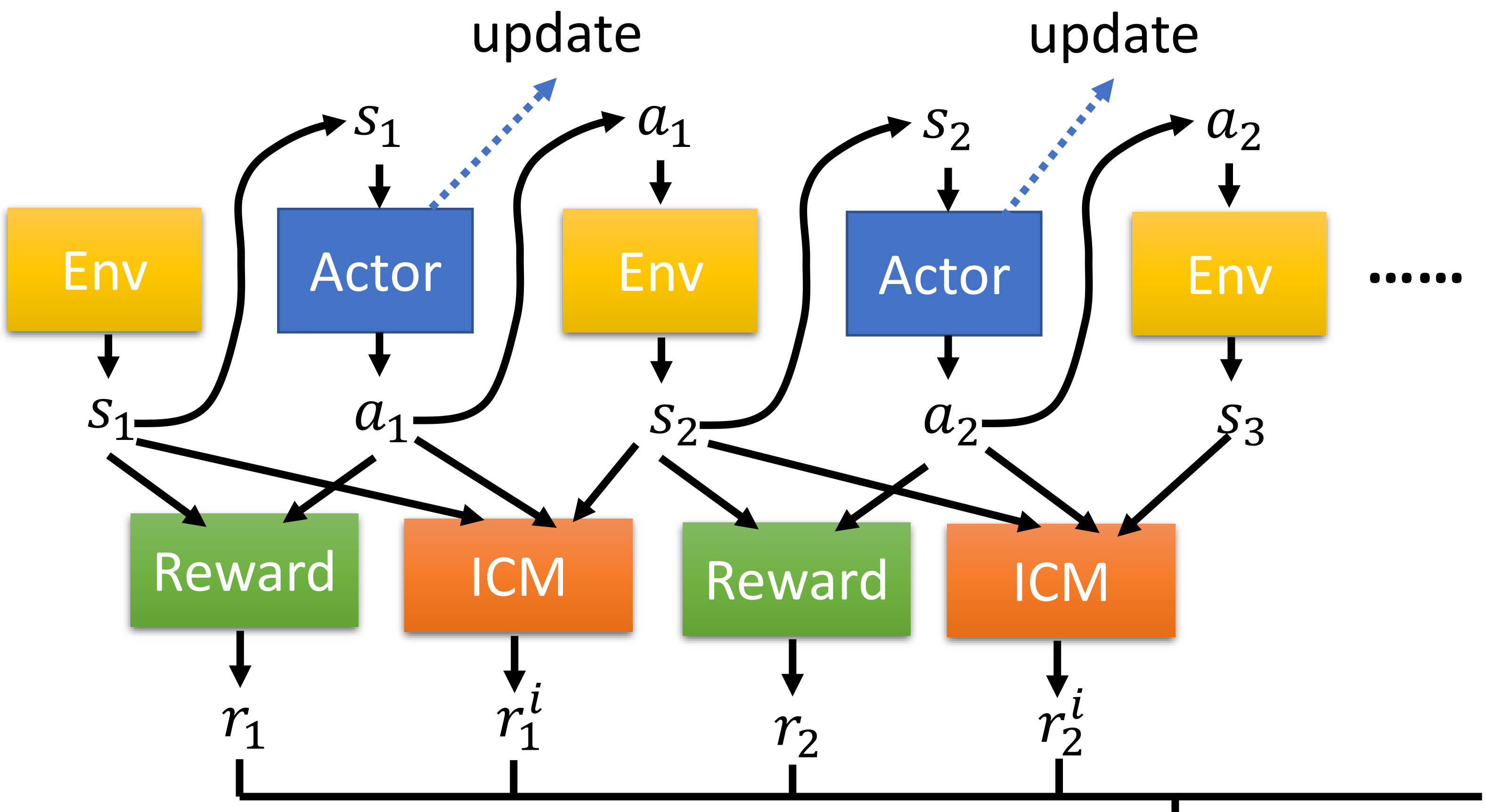
	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
Speed	0.2	0.2	0.4	0.4	0.6	0.8	0.8	1.0
Health	40	40	40	60	60	60	80	100



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

https://arxiv.org/abs/1705.05363

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

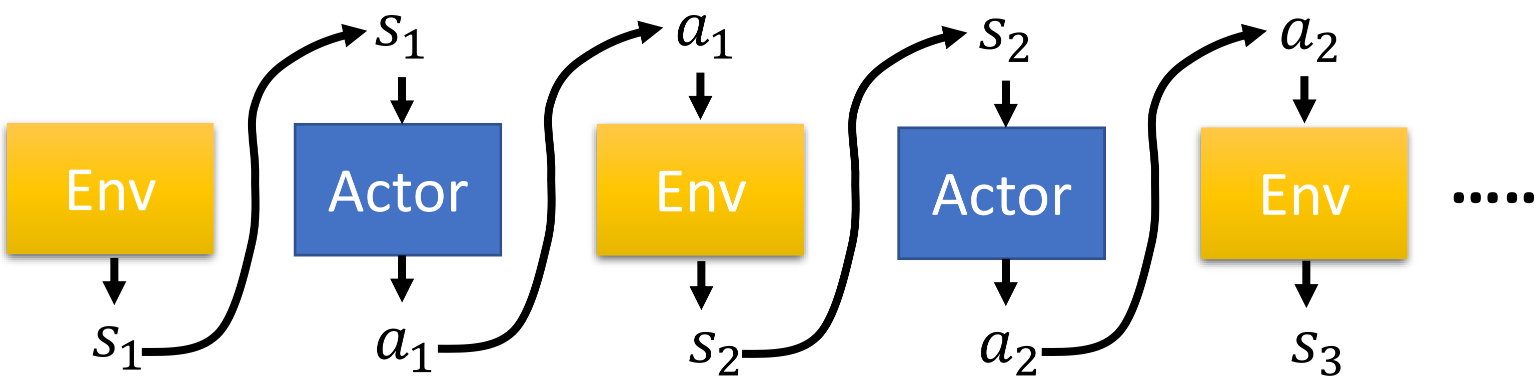
- Even define reward can be challenging in some tasks.
- Hand-crafted rewards can lead to uncontrolled behavior.

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.

?????

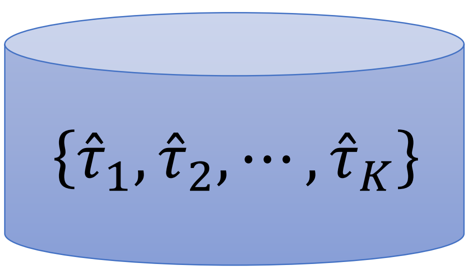
restraining individual human behavior and sacrificing some humans will ensure humanity's survival

Imitation Learning



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

We have demonstration of the expert.



Each $\hat{\tau}$ is a trajectory of the export.

Self driving: record human drivers

Robot: grab the arm of robot

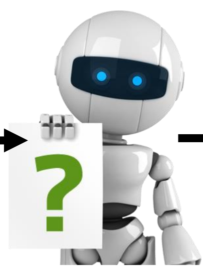
Isn't it Supervised Learning?

Yes, also known at *Behavior Cloning*

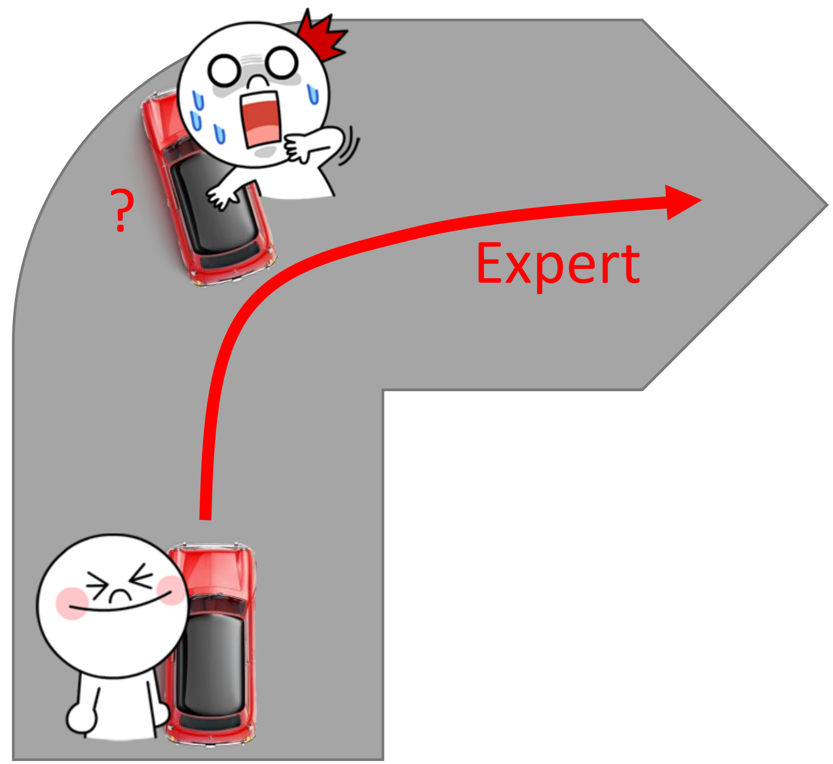
- Self-driving cars as example

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

forward

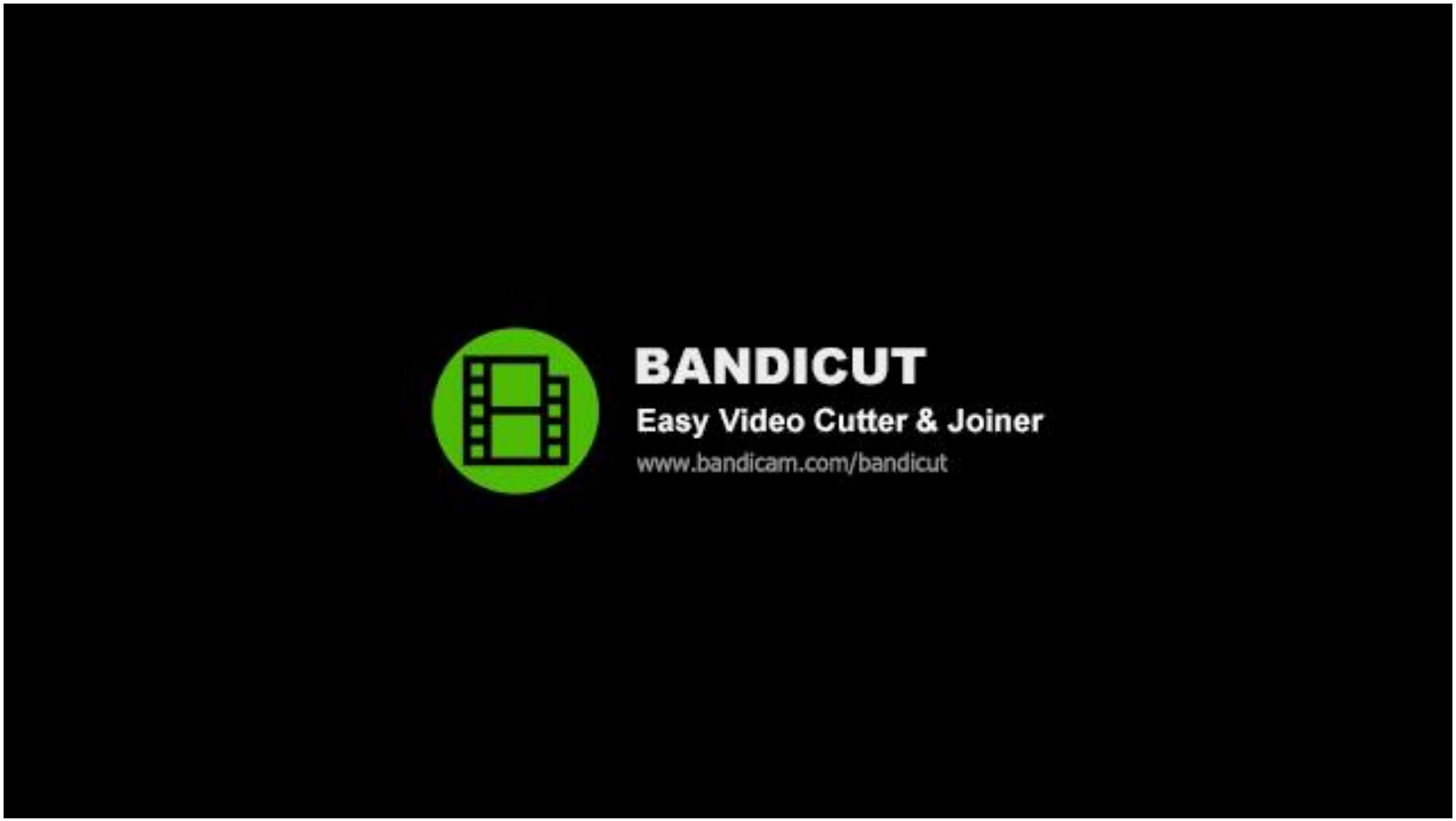
$s_i \rightarrow$  $\rightarrow a_i \longleftrightarrow \hat{a}_i$

Problem: The experts only sample limited observation.



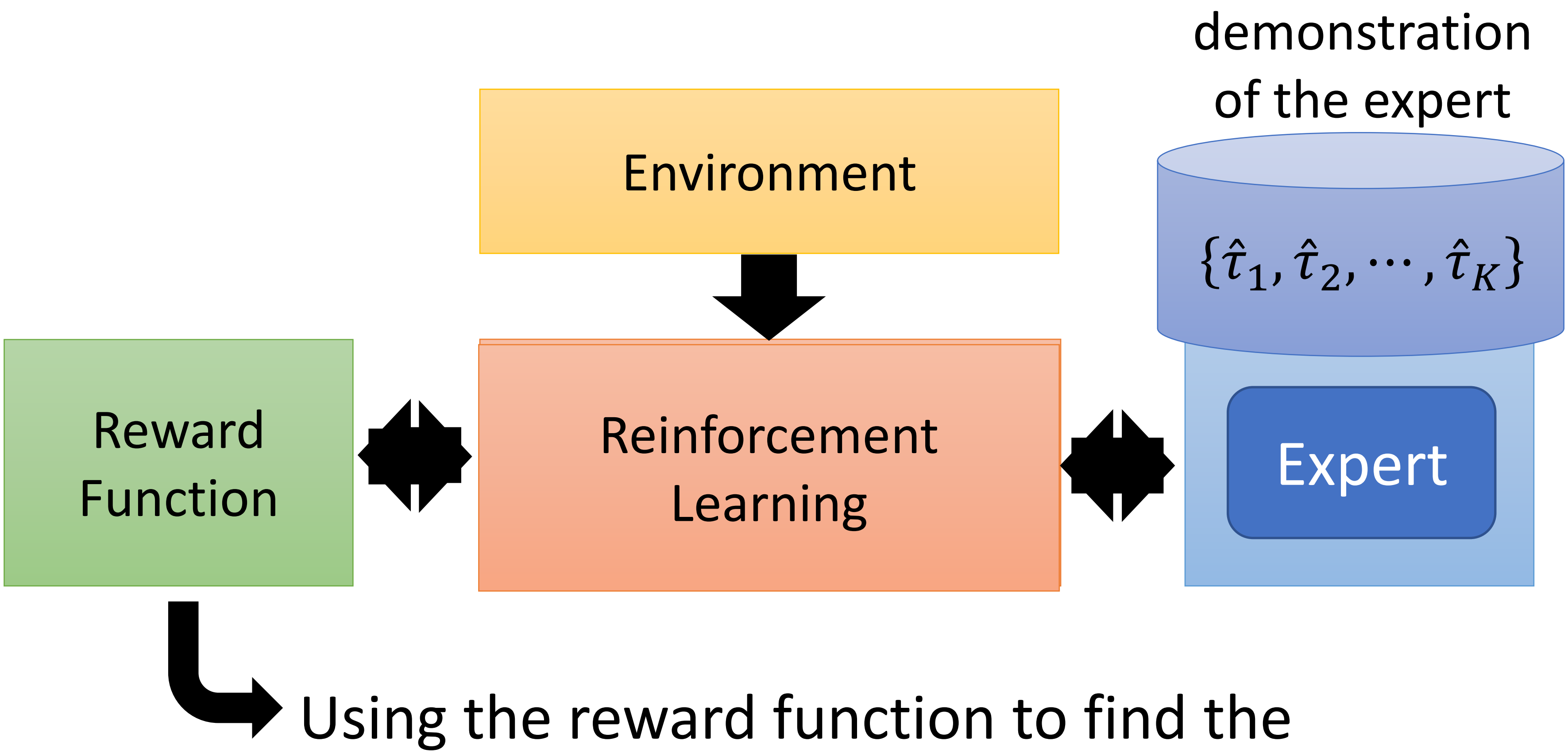
More problem ……

The agent will copy every behavior, even irrelevant actions.



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

Inverse Reinforcement Learning



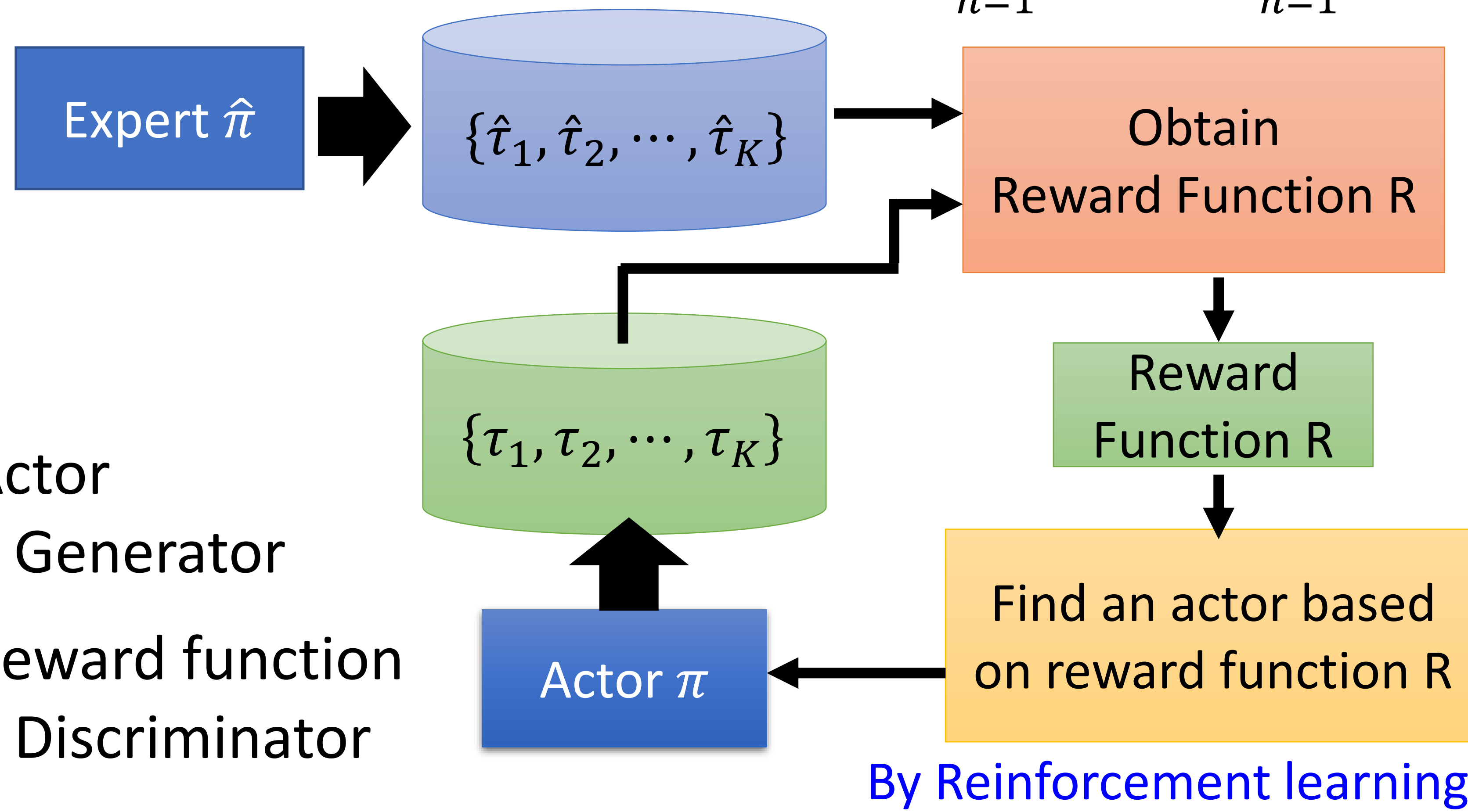
Using the reward function to find the ***optimal actor***.

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

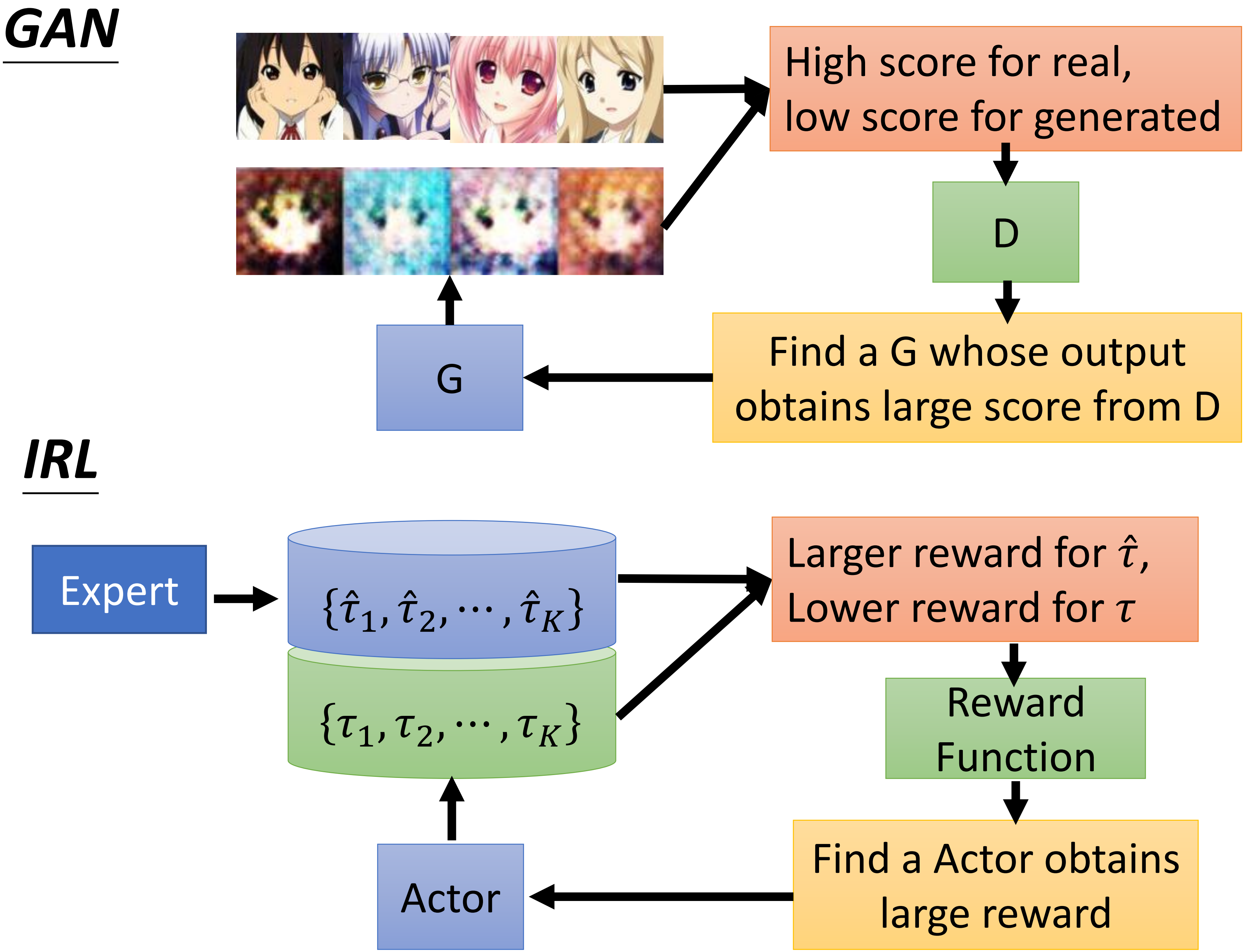
Framework of IRL

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



Actor
= Generator

Reward function
= Discriminator



GAN
High score for real,
low score for generated
D
Find a G whose output
obtains large score from D
G
IRL
Expert
$\{\hat{\tau}_1, \hat{\tau}_2, \cdots, \hat{\tau}_K\}$
$\{\tau_1, \tau_2, \cdots, \tau_K\}$
Larger reward for $\hat{\tau}$,
Lower reward for τ
Reward
Function
Find a Actor obtains
large reward
Actor

Robot

Chelsea Finn, Sergey Levine, Pieter Abbeel, Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization, ICML, 2016

http://rll.berkeley.edu/gcl/

Concluding Remarks

- What is RL? (Three steps in ML)
- Policy Gradient
- Actor-Critic
- Sparse Reward
- No Reward: Learning from Demonstration