2020 Spring IERG 6130
Reinforcement Learning and Beyond

Lecture 2: Get Hand Dirty by Coding RL

Bolei Zhou

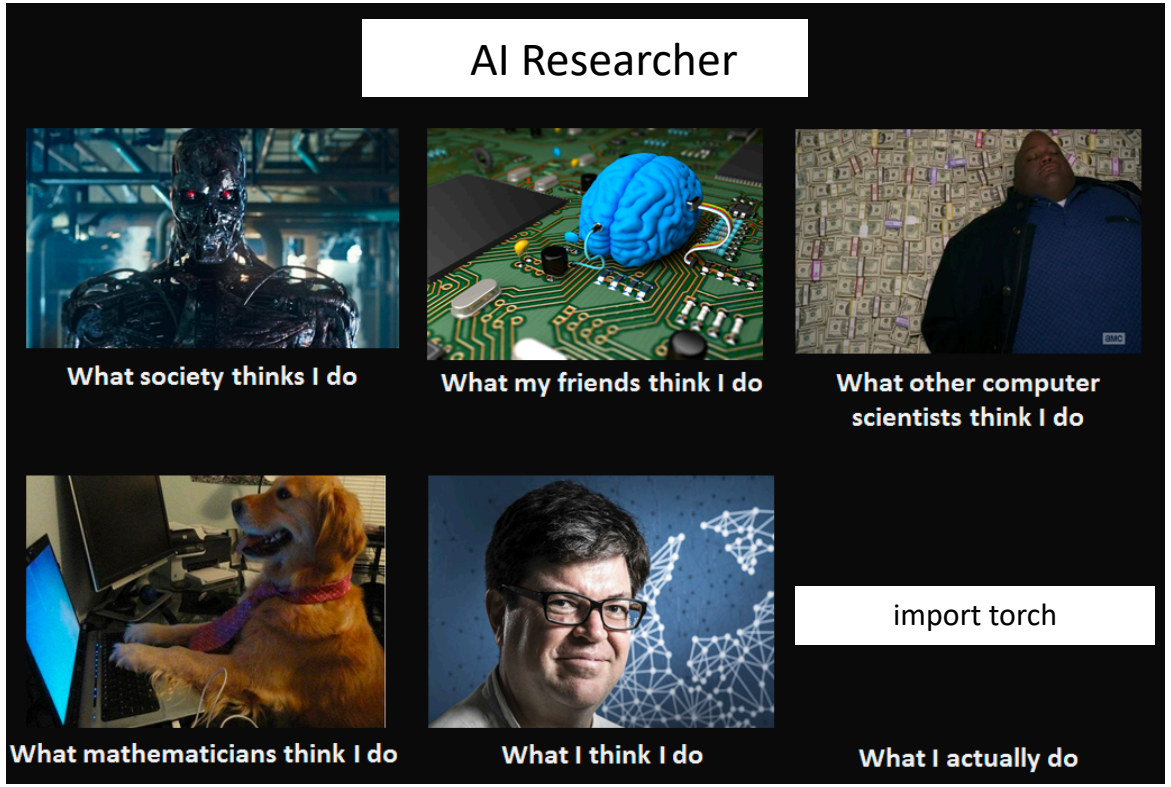
The Chinese University of Hong Kong

Experimenting with Reinforcement Learning

- Getting hand dirty on reinforcement learning is very important
- Deep learning and AI become more and more empirical
- Trial and error approach to learn reinforcement learning

Coding

- Python coding
- Deep learning libraries: PyTorch or TensorFlow
- https://github.com/metalbubble/RLexample

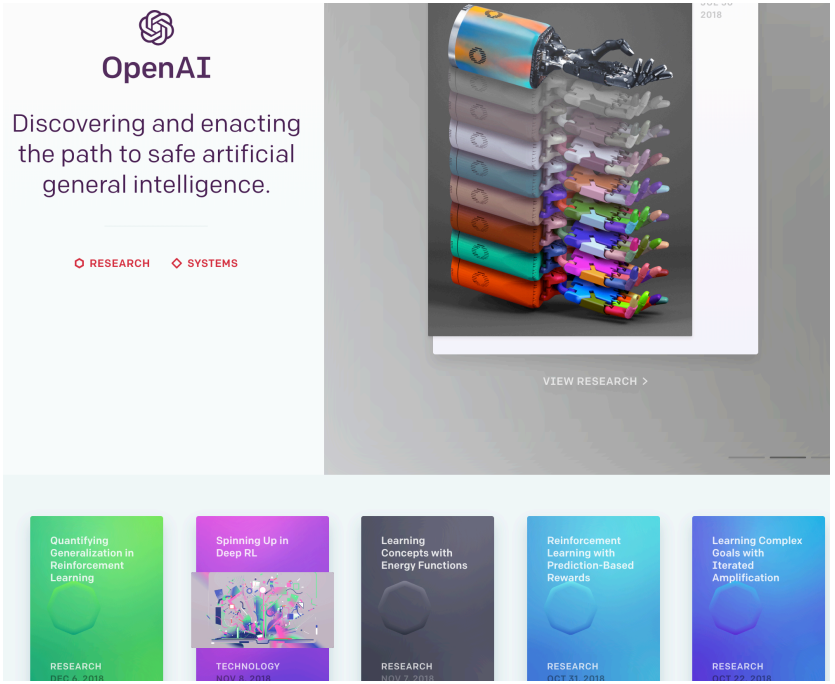


Reinventing Wheels? (造轮子？)

No. Start with existing libraries and pay more attentions to the specific algorithms

OpenAI: specialized in Reinforcement Learning

- https://openai.com/
- OpenAI is a non-profit AI research company, discovering and enacting the path to safe artificial general intelligence (AGI).



OpenAI gym library

https://gym.openai.com/

Gym

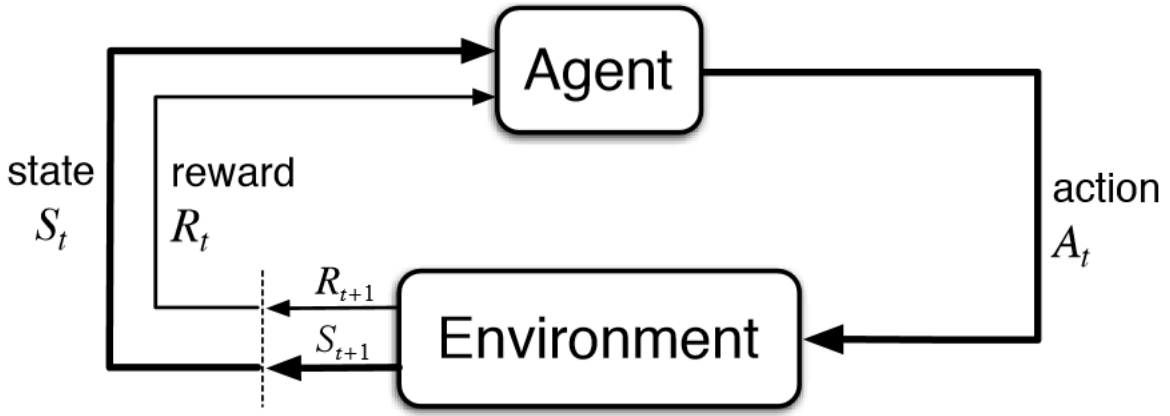
Gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents everything from walking to playing games like Pong or Pinball.

View documentation ›
View on GitHub ›

https://github.com/openai/retro

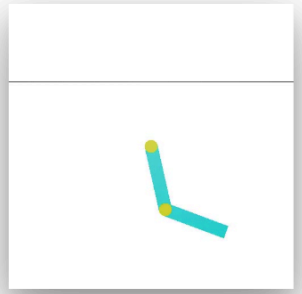
We're releasing the full version of Gym Retro, a platform for reinforcement learning research on games. This brings our publicly-released game count from around 70 Atari games and 30 Sega games to over 1,000 games across a variety of backing emulators. We're also releasing the tool we use to add new games to the platform.

Algorithmic interface of reinforcement learning

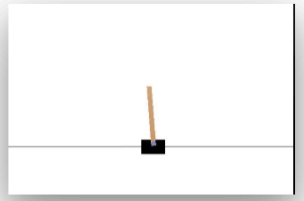



```
import gym
env = gym.make("Taxi-v2")
observation = env.reset()
agent = load_agent()
for step in range(100):
    action = agent(observation)
    observation, reward, done, info = env.step(action)
```

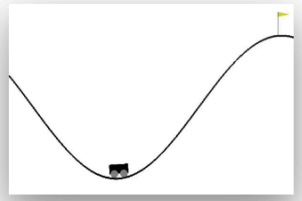
Classic Control Problems



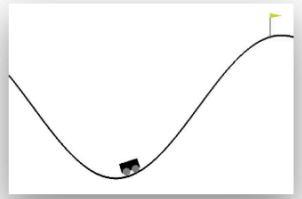
Acrobot-v1
Swing up a two-link robot.



CartPole-v1
Balance a pole on a cart.



MountainCar-v0
Drive up a big hill.

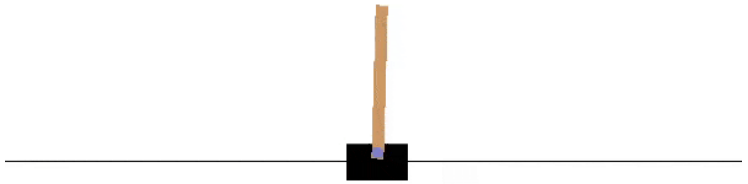


MountainCarContinuous-v0
Drive up a big hill with continuous control.

Pendulum-v0
Swing up a pendulum.

https://gym.openai.com/envs/#classic_control

Example of CartPole-v0



Actions

Type: Discrete(2)

Num	Action
0	Push cart to the left
1	Push cart to the right

Observation

Type: Box(4)

Num	Observation	Min	Max
0	Cart Position	-2.4	2.4
1	Cart Velocity	-Inf	Inf
2	Pole Angle	~ -41.8°	~ 41.8°
3	Pole Velocity At Tip	-Inf	Inf

Reward

Reward is 1 for every step taken, including the termination step

Episode Termination

1. Pole Angle is more than ±12°
2. Cart Position is more than ±2.4 (center of the cart reaches the edge of the display)
3. Episode length is greater than 200

https://github.com/openai/gym/blob/master/gym/envs/classic_control/cartpole.py

Example code

```
import gym
env = gym.make('CartPole-v0’)
env.reset()
env.render() # display the rendered scene
action = env.action_space.sample()
observation, reward, done, info = env.step(action)
```

Example code: Random Agent

python my_random_agent.py CartPole-v0

python my_random_agent.py Pong-ram-v0

python my_random_agent.py Breakout-v0

What is the difference in the format of the observations?

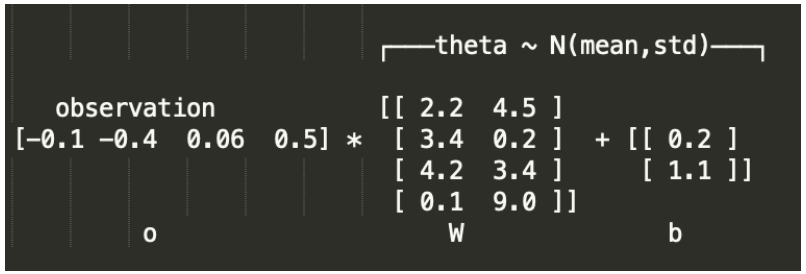
Example code: Naïve learnable RL agent

python my_random_agent.py CartPole-v0

python my_random_agent.py Acrobot-v1

python my_learning_agent.py CartPole-v0

python my_learning_agent.py Acrobot-v1



P_a = oW+b

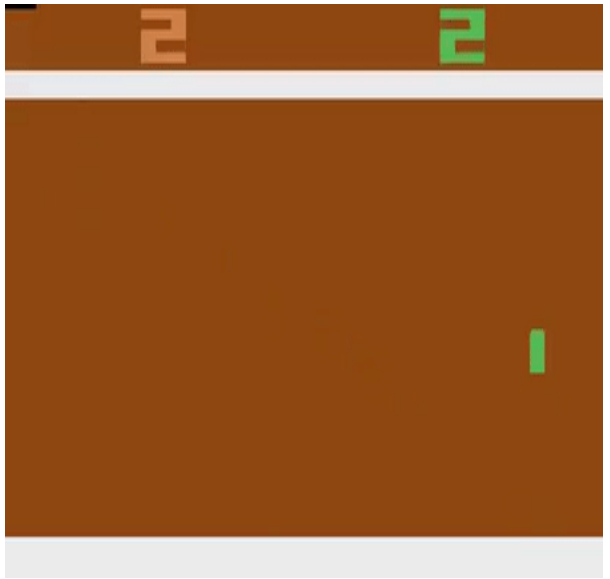
What is the algorithm?

Cross Entropy method (CEM)
https://gist.github.com/kashif/5dfa12d80402c559e060d567ea352c06

Deep Reinforcement Learning Example

- Pong example

```
import gym
env = gym.make('Pong-v0')
env.reset()
env.render() # display the rendered scene
```



```
python my_random_agent.py Pong-v0
```

Deep Reinforcement Learning Example

- Pong example

python pg-pong.py

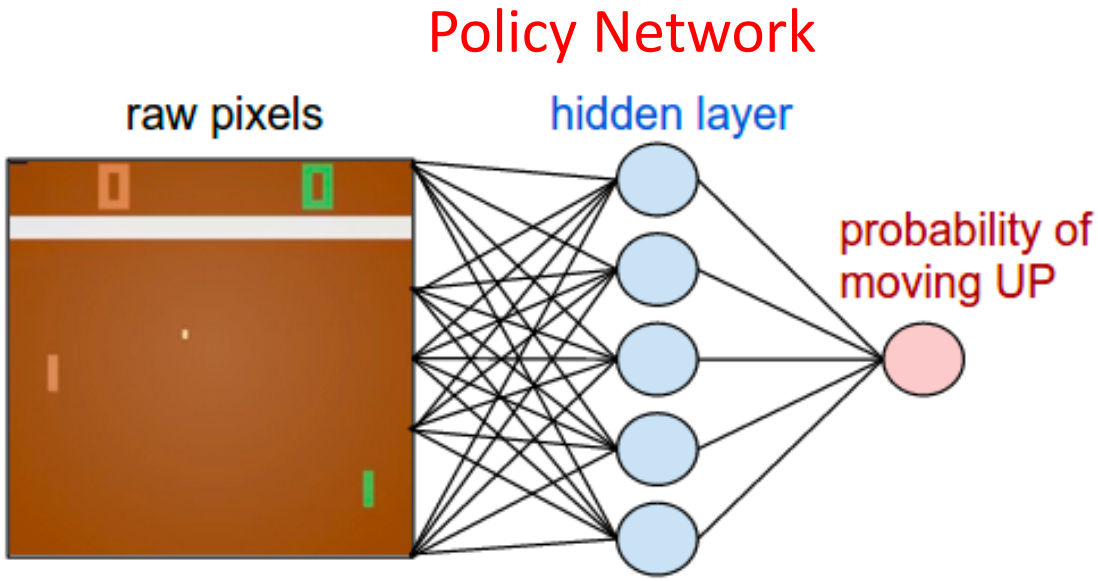
Loading weight: pong_bolei.p (model trained over night)

Deep Reinforcement Learning Example

- Look deeper into the code

```
observation = env.reset()

cur_x = prepro(observation)
x = cur_x - prev_x
prev_x = cur_x
aprob, h = policy_forward(x)
```



Randomized action:

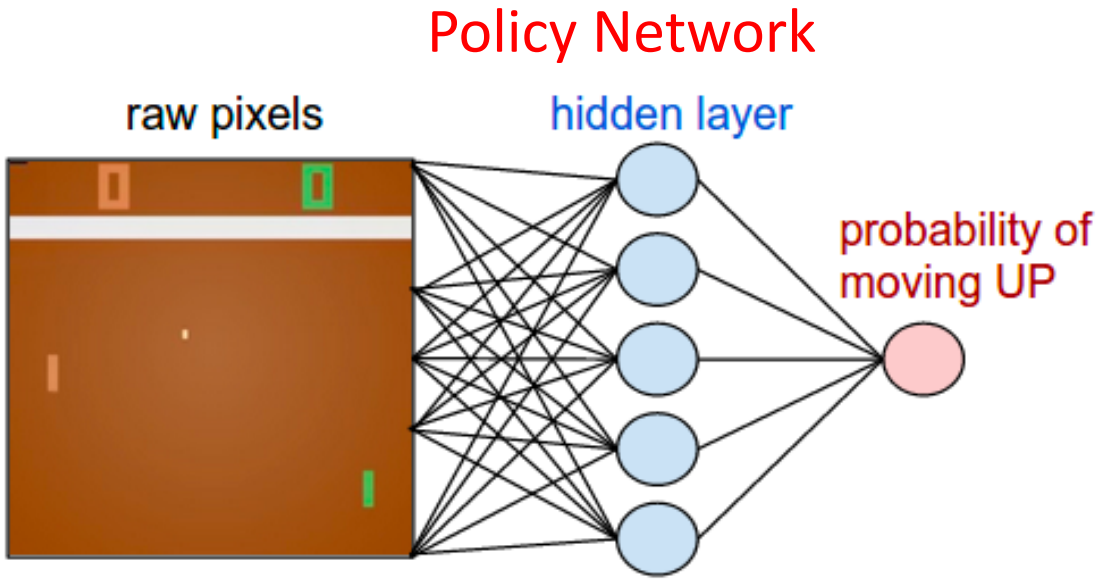
```
action = 2 if np.random.uniform() < aprob else 3 # roll the dice!
```

http://karpathy.github.io/2016/05/31/rl/

Deep Reinforcement Learning Example

- Look deeper into the code

```
h = np.dot(W1, x)
h[h<0] = 0 # ReLU nonlinearity: threshold at zero logp = np.dot(W2, h) # compute log probability of going up
p = 1.0 / (1.0 + np.exp(-logp)) # sigmoid function (gives probability of going up)
```



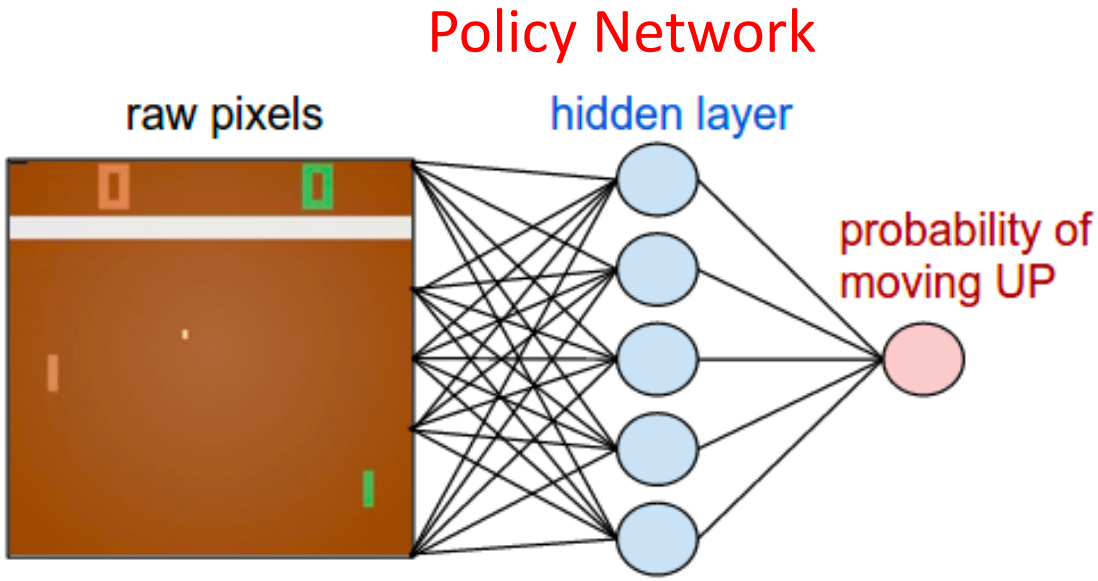
http://karpathy.github.io/2016/05/31/rl/

Deep Reinforcement Learning Example

- Look deeper into the code

How to optimize the W1 and W2?

Policy Gradient! (To be introduced in future lecture)



http://karpathy.github.io/2016/05/31/rl/

What could be the potential problems?

```
import gym
env = gym.make("Taxi-v2")
observation = env.reset()
agent = load_agent()
for step in range(100):
    action = agent(observation)
    observation, reward, done, info = env.step(action)
```

Speed, multiple agents, structure of agent?

Homework and What's Next

- Play with OpenAI gym and the example code

https://github.com/metalbubble/RLexample

- Try to understand my_learning_agent.py
- Go through this blog in detail to understand pg-pong.py

http://karpathy.github.io/2016/05/31/rl/

- Next week: Markov Decision Process

Please read **Sutton and Barton: Chapter 1 and Chapter 3**

In case you need it

- Python tutorial: http://cs231n.github.io/python-numpy-tutorial/
- Tensorflow tutorial: https://www.tensorflow.org/tutorials/
- PyTorch tutorial: https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html