Deep Reinforcement Learning

Scratching the surface
Deep Reinforcement Learning: AI = RL + DL
Scenario of Reinforcement Learning

Observation → State → Agent → Action → Change the environment → Reward → Don’t do that → Environment
Scenario of Reinforcement Learning

Agent learns to take actions to maximize expected reward.

Observation → State → Action → Change the environment → Reward

https://yoast.com/how-to-clean-site-structure/
Learning to play Go

Environment

Observation

Action

Reward

Next Move

Environment
Learning to play Go

Agent learns to take actions to maximize expected reward.

Observation

Action

Environment

reward = 0 in most cases

If win, reward = 1

If loss, reward = -1
Learning to paly Go
- Supervised v.s. Reinforcement Learning

• Supervised: Learning from teacher

Next move: “5-5”

Next move: “3-3”

• Reinforcement Learning Learning from experience

First move ...... many moves ...... Win!
(Two agents play with each other.)

Alpha Go is supervised learning + reinforcement learning.
Learning a chat-bot

• Sequence-to-sequence learning
Learning a chat-bot - Supervised v.s. Reinforcement

• Supervised
  - Say “Hi”
  - Say “Good bye”

• Reinforcement
  - Bad
Learning a chat-bot
- Reinforcement Learning

• Let two agents talk to each other (sometimes generate good dialogue, sometimes bad)

How old are you?
See you.

See you.
See you.

How old are you?
I am 16.

I though you were 12.

What make you think so?

See you.
Learning a chat-bot
- Reinforcement Learning

• By this approach, we can generate a lot of dialogues.
• Use some pre-defined rules to evaluate the goodness of a dialogue

```
Machine learns from the evaluation
```

Deep Reinforcement Learning for Dialogue Generation
More applications

- Interactive retrieval

More precisely, please.

US President

Trump

Is it related to “Election”? Yes.

Here are what you are looking for.

I see!

[Wu & Lee, INTERSPEECH 16]
More applications

• Flying Helicopter
  • https://www.youtube.com/watch?v=0JL04JJjocc

• Driving
  • https://www.youtube.com/watch?v=0x01Ldx3L5Q

• Google Cuts Its Giant Electricity Bill With DeepMind-Powered AI

• Text generation
  • Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, Wojciech Zaremba, “Sequence Level Training with Recurrent Neural Networks”, ICLR, 2016
Example: Playing Video Game

- Widely studies:
  - Gym: https://gym.openai.com/
  - Universe: https://openai.com/blog/universe/

  Machine learns to play video games as human players

  - What machine observes is pixels
  - Machine learns to take proper action itself
Example: Playing Video Game

- Space invader

Termination: all the aliens are killed, or your spaceship is destroyed.
Example: Playing Video Game

• Space invader
  • Play yourself: http://www.2600online.com/spaceinvaders.htm
  • How about machine: https://gym.openai.com/evaluations/eval_Eduox4HRYqgTCVk9Itw
Example: Playing Video Game

Start with observation $s_1$

Observe $s_2$

Observe $s_3$

Example: Playing Video Game

Obtain reward $r_1 = 0$

Action $a_1$: “right”

Obtain reward $r_2 = 5$

Action $a_2$: “fire”

(kill an alien)

Usually there is some randomness in the environment
Example: Playing Video Game

Start with observation $s_1$

After many turns

Observation $s_2$

Game Over (spaceship destroyed)

Obtain reward $r_T$

Action $a_T$

Observation $s_3$

This is an episode.

Learn to maximize the expected cumulative reward per episode.
Difficulties of Reinforcement Learning

• Reward delay
  • In space invader, only “fire” obtains reward
    • Although the moving before “fire” is important
  • In Go playing, it may be better to sacrifice immediate reward to gain more long-term reward

• Agent’s actions affect the subsequent data it receives
  • E.g. Exploration
Outline

Alpha Go: policy-based + value-based + model-based

Policy-based

Learning an Actor

Actor + Critic

Value-based

Learning a Critic

Asynchronous Advantage Actor-Critic (A3C)
To learn deep reinforcement learning ......

- Textbook: Reinforcement Learning: An Introduction

- Lectures of David Silver
  - http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html (10 lectures, 1:30 each)
  - http://videolectures.net/rldm2015_silver_reinforcement_learning/ (Deep Reinforcement Learning )

- Lectures of John Schulman
  - https://youtu.be/aUrX-rP_ss4
Policy-based Approach

Learning an Actor
Machine Learning
≈ Looking for a Function

Observation
Function input

Actor/Policy

Action = \pi(\text{Observation})

Action
Function output

Used to pick the best function

Reward

Environment
Three Steps for Deep Learning

Step 1: Define a set of function

Step 2: Goodness of function

Step 3: Pick the best function

Deep Learning is so simple ......
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

What is the benefit of using network instead of lookup table?

Probability of taking the action:
- left: 0.7
- right: 0.2
- fire: 0.1

Generalization
Three Steps for Deep Learning

Step 1: Neural Network as Actor

Step 2: goodness of function

Step 3: pick the best function

Deep Learning is so simple ......
Goodness of Actor

• Review: Supervised learning

Training Example

Given a set of parameters $\theta$

Softmax

Loss $l$

As close as possible

Target

Total Loss:

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

Find the network parameters $\theta^*$ that minimize total loss $L$
Goodness of Actor

- Given an actor $\pi_\theta(s)$ with network parameter $\theta$
- 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 $r_T$
  - Total reward: $R_\theta = \sum_{t=1}^{T} r_t$

Even with the same actor, $R_\theta$ is different each time

Randomness in the actor and the game

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

$\bar{R}_\theta$ evaluates the goodness of an actor $\pi_\theta(s)$
Goodness of Actor

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

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

Sum over all possible trajectory

Use $\pi_\theta$ to play the game $N$ times, obtain $\{\tau^1, \tau^2, \ldots, \tau^N\}$

Sampling $\tau$ from $P(\tau|\theta)$ $N$ times
Three Steps for Deep Learning

Step 1: Neural Network as Actor

Step 2: Goodness of Function

Step 3: Pick the best function

Deep Learning is so simple ......
Gradient Ascent

• Problem statement

\[ \theta^* = \arg \max_{\theta} \bar{R}_{\theta} \]

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

• Gradient ascent

• Start with \( \theta^0 \)
• \( \theta^1 \leftarrow \theta^0 + \eta \nabla \bar{R}_{\theta^0} \)
• \( \theta^2 \leftarrow \theta^1 + \eta \nabla \bar{R}_{\theta^1} \)
• ......
Gradient Ascent

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

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

\( R(\tau) \) do not have to be differentiable
It can even be a black box.

\[ \approx \frac{1}{N} \sum_{n=1}^{N} R(\tau^n)\nabla \log P(\tau^n|\theta) \]

\[ \frac{d \log(f(x))}{dx} = \frac{1}{f(x)} \frac{df(x)}{dx} \]

Use \( \pi_\theta \) to play the game \( N \) times,
Obtain \( \{\tau^1, \tau^2, \ldots, \tau^N\} \)
Gradient Ascent

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

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

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

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

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

- Control by your actor $$\pi_{\theta}$$
- not related to your actor

Actor

$$\pi_{\theta}$$

- left
  - 0.1
- right
  - 0.2
- fire
  - 0.7

$$p(a_t = "fire" | s_t, \theta) = 0.7$$
Gradient Ascent

\[ \nabla \log P(\tau|\theta) = ? \]

- \( \tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_T, a_T, r_T\} \)

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

\[
\log P(\tau|\theta)
\]

\[
= \log p(s_1) + \sum_{t=1}^{T} \log p(a_t|s_t, \theta) + \log p(r_t, s_{t+1}|s_t, a_t)
\]

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

Ignore the terms not related to \( \theta \)
Gradient Ascent

\[ \theta^{\text{new}} \leftarrow \theta^{\text{old}} + \eta \nabla \bar{R}_{\theta}^{\text{old}} \]

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

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

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

\[ R(\tau^n) \text{ is positive} \quad \Rightarrow \quad \text{Tuning } \theta \text{ to increase } p(a_t^n | s_t^n) \]

\[ R(\tau^n) \text{ is negative} \quad \Rightarrow \quad \text{Tuning } \theta \text{ to decrease } p(a_t^n | s_t^n) \]

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

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

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

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

\[ \frac{\nabla p(a_t^n | s_t^n, \theta)}{p(a_t^n | s_t^n, \theta)} \]

Why divided by \( p(a_t^n | s_t^n, \theta) \)?

e.g. in the sampling data ... s has been seen in \( \tau^{13}, \tau^{15}, \tau^{17}, \tau^{33} \)

<table>
<thead>
<tr>
<th>( \ln \tau^{13} ), take action a</th>
<th>( R(\tau^{13}) = 2 )</th>
<th>( \ln \tau^{15} ), take action b</th>
<th>( R(\tau^{15}) = 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \ln \tau^{17} ), take action b</td>
<td>( R(\tau^{17}) = 1 )</td>
<td>( \ln \tau^{33} ), take action b</td>
<td>( R(\tau^{33}) = 1 )</td>
</tr>
</tbody>
</table>
Add a Baseline

$$\theta_{\text{new}} \leftarrow \theta_{\text{old}} + \eta \nabla \bar{R}_{\theta_{\text{old}}}$$

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

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

Ideal case

Sampling

......

It is probability ...

The probability of the actions not sampled will decrease.
Value-based Approach

Learning a Critic
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.
e.g. Q-learning

(not today)

Three kinds of Critics

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

![Diagram](image.png)
Three kinds of Critics

- State-action value function $Q^\pi(s, a)$
- When using actor $\pi$, 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 $\pi$ 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.
How to estimate $V^\pi(s)$

• The critic has the following 8 episodes
  - $s_a, r = 0, s_b, r = 0, \text{END}$
  - $s_b, r = 1, \text{END}$
  - $s_b, r = 1, \text{END}$
  - $s_b, r = 1, \text{END}$
  - $s_b, r = 1, \text{END}$
  - $s_b, r = 1, \text{END}$
  - $s_b, r = 0, \text{END}$

$V^\pi(s_b) = \frac{3}{4}$

Monte-Carlo: $V^\pi(s_a) = ? \ 0? \ 3/4?$

Temporal-difference:

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

3/4 \ 0 \ 3/4

(The actions are ignored here.)
Deep Reinforcement Learning
Actor-Critic
Actor-Critic

\[ \theta^{\text{new}} \leftarrow \theta^{\text{old}} + \eta \nabla \bar{R}_{\theta^{\text{old}}} \]

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

Advantage Function: \[ r_t^n = (V_{\pi\theta}(s_t^n) - V_{\pi\theta}(s_{t+1})) \]

Evaluated by critic

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 \( \pi_{\theta} \)

Positive advantage function

Increasing the prob. of action \( a_t^n \)

Negative advantage function

decreasing the prob. of action \( a_t^n \)
Actor-Critic

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

  ![Diagram]

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

1. Copy global parameters
2. Sampling some data
3. Compute gradients
4. Update global models

\[ \theta^1 + \eta \Delta \theta \]

(other workers also update models)
Demo of A3C

• DeepMind
  https://www.youtube.com/watch?v=nMR5mjCFZCw
Demo of A3C

- DeepMind
  https://www.youtube.com/watch?v=0xo1Ldx3L5Q
Conclusion of This Semester
Learning Map

- Supervised Learning
  - Regression
    - Linear Model
      - Deep Learning
        - SVM, decision tree, K-NN ...
        - Non-linear Model
      - Classification
    - Structured Learning
      - Semi-supervised Learning
      - Transfer Learning
      - Unsupervised Learning
      - Reinforcement Learning
Acknowledgment

• 感謝 Larry Guo 同學發現投影片上的符號錯誤