Introduction to Reinforcement learning

Tri Nguyen

Oregon State University

August 19, 2021

Reinforcement Learning (RL)

	With teacher	Without teacher
Passive	Supervised Learning	Self-(un)supervised Learning
Active	**Reinforcement Learning**	Intrinsic Motivation (Exploration)

Table: Tutorial - ICML2021

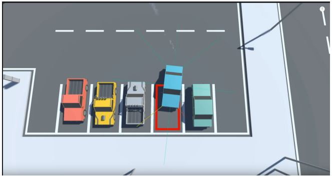
Learning **what-to-do** from **interaction** and optimizing **reward**.

Problems to cast to RL

(a) Board games

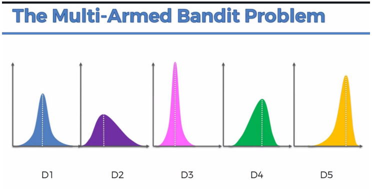
(b) Robot manipulation

(c) Real-time strategy games



(d) Self-driving car

and many more . . .



Given n bandit machines. You can pull the level of one of them and observer the result: either nothing or win a fixed amount of cash. Each bandit machine has it owns wining distribution. If you can play M times, what's your strategy to maximize total amount of cash?

- A_t as action picked at step t
- R_t as reward received at step t
- Action value: $q_*(a) = \mathbb{E}[R_t | A_t = a]$
- Estimate action value at step t: $Q_t(a)$

Balancing between exploitation and exploration!

Baseline: ϵ-greedy Algorithm

Estimate $q_*(a)$ by $Q_n(a) = \dfrac{R_1 + R_2 + \ldots R_{n-1}}{n-1}$

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Loop forever:

$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$

$R \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\big[R - Q(A)\big]$

Figure: From [Sutton&Barto]

Variant 1: Optimistic Initial Values

Init $Q(a)$ as a nonzero constant $C > 0$.

A simple bandit algorithm

Initialize, for $a = 1$ to k:
$Q(a) \leftarrow C$
$N(a) \leftarrow 0$

Loop forever:
$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1 - \varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$
$R \leftarrow bandit(A)$
$N(A) \leftarrow N(A) + 1$
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\big[R - Q(A)\big]$

Figure: From [Sutton&Barto]

Variant 2: Upper-Confidence-Bound Action Selection (UCB)

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Loop forever:

$\boxed{A_t \doteq \underset{a}{\arg\max}\left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}}\right]}$

$R \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

Figure: From [Sutton&Barto]

Evaluation

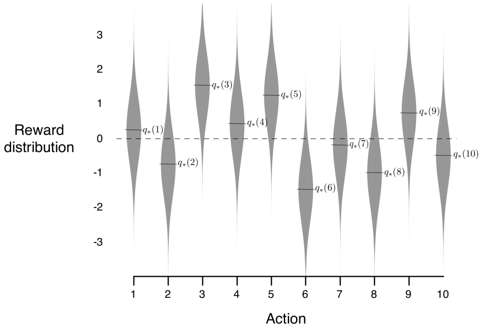
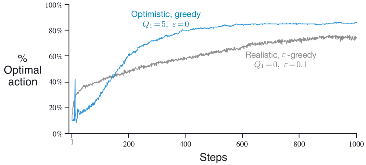


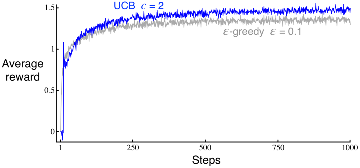
Figure: From [Sutton&Barto]

- 10-armed bandits
- Each $q_*(a) \sim \mathcal{N}(0, 1)$
- And then the actual rewards are drawn from $\mathcal{N}$ with a mean $q_*(a)$ and unit variance

The results are averages over 2000 trials.



(a) ϵ-greedy v.s Optimistic Initial Value



(b) ϵ-greedy v.s UCB

Note that in both figures, there is a jump around in early part of the curve.

- In the first 10 iterations, all actions are seleted once regardless of actual rewards.
- After that, $Q_{11}(a)$ might estimate $q_*(a)$ relatively correct, i.e., $\arg\max_a Q_{11}(a) = \arg\max_a q_*(a)$
- Then it is likely (40%) that an optimal action is picked.

Reinforcement Learning Framework

General question: how to train an agent to archive a goal by interacting with the environment, e.g., train a robot to escape a maze.
RL solves it by using the following framework:

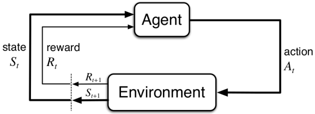


Figure: [Sutton&Barto]

Input of the framework

Environment - Agent interation:

- At time step t, *agent* at state S_t performs an action $A_t \in \mathcal{A}(S_t)$
- The *environment* acts accordingly change to state S_{t+1}, emits a reward R_{t+1}
- *Return*: $G_t = \sum_{k=1}^{\infty} \gamma^k R_{t+k+1}$

Output of the framework

An agent that acts on the environment so that it maximizes the expectation of *return*, i.e., $\mathbb{E}[G_t]$.

Examples of using framework

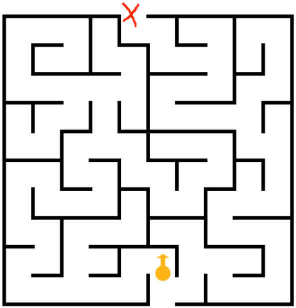


Figure: Given a position, find a way to reach the red X

Goal: Given a position, find a way to reach the red X

- At time t, state S_t is current position
- Agent at state S_t can move to any valid directions (direction not break wall)
- Environment dynamic:
  - Agent being at the desired position by 1 unit
  - Emits a reward of 10 when it reach X, otherwise 0.
- Return $G_t = \sum_{k=1}^{\infty} R_{t+k+1}$.
  Not that we choose $\gamma = 1$, and T is number of time steps until the agent reaches red X

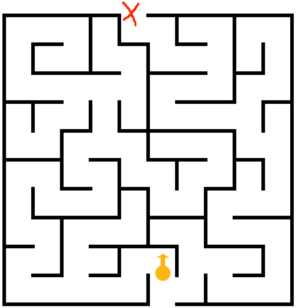


Figure: Given a position, find a way to reach the red X as fast as possible

Goal: Given a position, find a way to reach the red X as fast as possible!

- At time t, state S_t is current position
- Agent at state S_t can move to any valid directions (direction not break wall)
- Environment dynamic:
  - Agent being at the desired position by 1 unit
  - Emits a reward of 10 when it reach X, otherwise -1.
- Return $G_t = \sum_{k=1}^{\infty} R_{t+k+1}$.
  Not that we choose $\gamma = 1$, and T is number of time steps until the agent reaches red X

Cart-Pole Problem

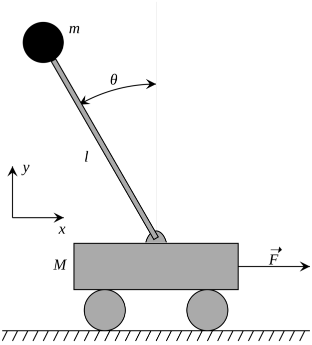


Figure: Keep the pole from falling by moving the card horizontally (From Lecture [1])

Goal: Keep the pole from falling by moving the card horizontally

- At time t, state S_t is a collection of *angle, angular speed, position, horizontal vertical*
- Agent at state S_t can apply a force horizontally to the cart.
- Environment dynamic:
  - The pole acts accordingly to physical laws.
  - Emits a reward of 1 when the pole is upright, otherwise -10.
  - Once the pole hits ground, there's no way to make it be upright.
- Return $G_t = \sum_{k=1}^{\infty} R_{t+k+1}$.
  Not that we choose $\gamma = 1$, and T is number of time steps until the pole is dropped.

[1] Lecture CS231 Stanford - Fei-Fei Li & Justin Johnson & Serena Yeung

Playing chess

Figure: Playing chess

Goal: Win as much as possible!

- At time t, state S_t is a position of all chess piecies.
- Agent at state S_t can apply a valid move of one of its chess pieces.
- Environment dynamic:
  - The opponent will move 1 of its piece.
  - Emits a reward of 0 when the game is not terminated, otherwise 1 for winning, -1 for losing, 0 for drawing.
- Return $G_t = \sum_{k=1}^{\infty} R_{t+k+1}$.

RL Framework

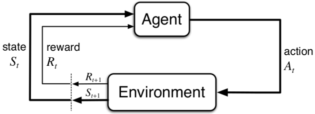


Figure: (ref. Book)

Input of the framework

Environment - Agent interation:

- At time step t, *agent* at state S_t performs an action $A_t \in \mathcal{A}_t$
- (2) Environment's dynamics. The *environment* acts accordingly change to state S_{t+1}, emits a reward R_{t+1}
- (1) Episodic/Continuing task. *Return*: $G_t = \sum_{k=1}^{\infty} \gamma^k R_{t+k+1}$

Output of the framework

(3) How. An agent that acts on the environment so that it maximizes the expectation of *return*, i.e., $\mathbb{E}[G_t]$.

(1) Episodic/Continuing tasks

The interation agent-environment produces $S_1, A_1, R_2, S_2, A_2, R_3, \ldots$

Episodic tasks

- The sequence can break natually into subsequences
- Example: playing chess
- Return

$$G_t = \sum_{k=1}^{T} \gamma^k R_{t+k+1}, 0 \leq \gamma \leq 1$$

- There exits a termination state

Continuing tasks

- Otherwise
- Example: Robot with long life span
- Return

$$G_t = \sum_{k=1}^{\infty} \gamma^k R_{t+k+1}, 0 \leq \gamma < 1$$

- There is no notion of termination state

In both cases: $G_t = R_{t+1} + \gamma G_{t+1}$.

(2) Environment's dynamics

Markov decision process (MDP) descibes environment's dynamics

- Finite sets of states, action, rewards $\mathcal{S}, \mathcal{A}, \mathcal{R}$
- Random variables $S_t \in \mathcal{S}, R_t \in \mathcal{R}$ are only dependent on preceding state and action, i.e.,
  $p(s', r|s, a) := \Pr(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a)$
- **Markov property.** State must include all information of the past that makes a difference for the future

Example: A recycling robot Describe more Some justication Diff on agent and Env

s	a	s'	$p(s' \mid s, a)$	$r(s, a, s')$
high	search	high	α	$r_{\texttt{search}}$
high	search	low	$1 - \alpha$	$r_{\texttt{search}}$
low	search	high	$1 - \beta$	-3
low	search	low	β	$r_{\texttt{search}}$
high	wait	high	1	$r_{\texttt{wait}}$
high	wait	low	0	-
low	wait	high	0	-
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	-

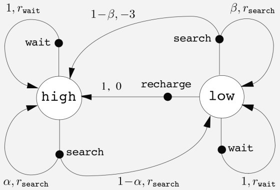


Figure: Example of an MPD

(3) How. Part1: Evaluation — Policy and Value function

- Policy π is a mapping from $\mathcal{S} \longrightarrow \mathcal{D}(a)$ where $\mathcal{D}(a)$ is some probability distribution over action space. If the agent follows policy π,

$$\pi(a|s) = \Pr(A_t = a|S_t = s)$$

- *Value function* of a state under π, denoted $v_\pi(s)$, is the expected return when starting in s and following π thereafter, i.e.,

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1}|S_t = s\right]$$

- We call v_π the *state-value function for policy* π
- Based on that, define the *action-value function* for policy π as

$$\begin{aligned} q_\pi(s,a) &= \mathbb{E}_\pi[R_{t+1} + \gamma G_t|S_t = s, A_t = a] \\ &= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a\right] \end{aligned}$$

Bellman Equation of Value Function

$$
\begin{aligned}
& v_\pi(s) \\
& = \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1}|S_t = s\right] \\
& = \sum_a \pi(a|s)\mathbb{E}\left[R_{t+1} + \gamma G_{t+1}|S_t = s\right] \\
& = \sum_a \pi(a|s)\sum_{s',r} p(s', r|S_t = s, A_t = a)\left[r + \gamma G_{t+1}\right] \\
& = \sum_a \pi(a|s)\sum_{s',r} p(s', r|S_t = s, A_t = a)\left(r + \gamma\mathbb{E}_\pi\left[R_{t+2} + \gamma G_{t+2}|S_{t+1} = s'\right]\right) \\
& = \sum_a \pi(a|s)\sum_{s',r} p(s', r|S_t = s, A_t = a)\left(r + \gamma v_\pi(s')\right)
\end{aligned}
$$

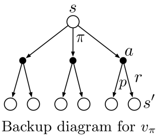


Figure: Backup operation for state-value function

(3) How. Part 2: Optimal Policies

Definition

A poicy π is defined to be better than or equal to a policy π' if its expected return is greater than or equal to that of π' for **all states**. In other words,

$$\pi \geq \pi' \Leftrightarrow v_\pi(s) \geq v_{\pi'}(s) \text{ for all } s \in \mathcal{S}$$

Definition

π_* is the *optimal policy* if and only if $\pi_* \geq \pi$ for any π. Value of optimal policy is called *optimal state-value function*, denoted v_* and defined as

$$v_*(s) := \max_\pi v_\pi(s), \quad \text{for all } s \in \mathcal{S}$$

Similarly, $q_*(s, a)$ is *optimal action-value function* and defined as

$$q_*(s, a) := \max_\pi q_\pi(s, a), \quad \text{for all } s \in \mathcal{S}, a \in \mathcal{A}_t$$

Bellman Optimality Equation

Assume π_* exists,

$$\begin{aligned} v_*(s) &= v_{\pi_*}(s) \quad \text{(by definition)} \\ &= \max_{a \in \mathcal{A}(s)} \; q_{\pi_*}(a, s) \\ &= \max_{a \in \mathcal{A}(s)} \; \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a\right] \\ &= \max_{a \in \mathcal{A}(s)} \; \mathbb{E}\left[R_{t+1} + \gamma v_*(s') | S_t = s, A_t = a\right] \\ \Rightarrow v_*(s) &= \max_{a \in \mathcal{A}(s)} \; \sum_{s', r} p(s', r | S_t = s, A_t = a)\left(r + \gamma v_*(s')\right) \qquad (1) \end{aligned}$$

Compare to Bellman equation

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s', r} p(s', r | S_t = s, A_t = a)\left(r + \gamma v_\pi(s')\right)$$

the equation (1) doesn't depend on any particular policy

Properties regarding Bellman Equations

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|S_t = s, A_t = a)\,(r + \gamma v_\pi(s')) \qquad (2)$$

$$v_*(s) = \max_{a \in \mathcal{A}(s)} \sum_{s',r} p(s', r|S_t = s, A_t = a)\,(r + \gamma v_*(s')) \qquad (3)$$

Remark

- Bellman equation (2) has an unique solution, which is $v_\pi(s)$.
- Bellman optimality equation (3) has an unique solution, which is $v_*(s)$

Implication:

- Given an optimal value function, *greedy policy* is the optimal policy, (actions satisfies (3))
- Given an optimal action-value function $q_*(s, a)$, the optimal policy is $\arg\max_a q_*(s, a)$

Sketch proof of Remark 0.1. Define

- A fixed point of a function f is x such that $x = f(x)$
- An function $f : \mathbb{R}^N \to \mathbb{R}^N$ is called a contraction if there exists $0 < \alpha < 1$ such that

$$\left\| f(\boldsymbol{s}) - f(\boldsymbol{s}') \right\|_p \leq \alpha \left\| \boldsymbol{s} - \boldsymbol{s}' \right\|_p, \qquad \text{for some } p \geq 1$$

1. If f is an α-contraction, then f has an unique fixed point
2. Let $\boldsymbol{v} \in \mathbb{R}^N$ be a vector of all value states,

$$T_\pi(\boldsymbol{v})[s] := \sum_a \pi(a|s) \sum_{s',r} p(s', r | S_t = s, A_t = a) \left(r + \gamma v_\pi(s') \right)$$

$$T_*(\boldsymbol{v})[s] := \max_{a \in \mathcal{A}(s)} \sum_{s',r} p(s', r | S_t = s, A_t = a) \left(r + \gamma v_*(s') \right)$$

T_π, T_* are both contraction.

Step 1

If f is a α-contraction, then f has an unique fixed point.

Proof.

Uniqueness Let $\boldsymbol{s}, \boldsymbol{s}'$ are 2 fixed points of f.

$$\alpha \left\| \boldsymbol{s} - \boldsymbol{s}' \right\|_p \geq \left\| f(\boldsymbol{s}) - f(\boldsymbol{s}') \right\|_p = \left\| \boldsymbol{s} - \boldsymbol{s}' \right\|_p$$

Since $0 < \alpha < 1$, this contraction leads to $\boldsymbol{s} = \boldsymbol{s}'$.
Existence

- Define a sequence of $\boldsymbol{s}_k$ such that $\boldsymbol{s}_{k+1} = f(\boldsymbol{s}_k)$

$$\begin{aligned}
\left\| \boldsymbol{s}_{k+1} - \boldsymbol{s}_k \right\|_p &= \left\| f(\boldsymbol{s}_k) - f(\boldsymbol{s}_{k-1}) \right\|_p \\
&\leq \alpha \left\| \boldsymbol{s}_k - \boldsymbol{s}_{k-1} \right\|_p = \alpha \left\| f(\boldsymbol{s}_{k-1}) - f(\boldsymbol{s}_{k-2}) \right\|_p \\
&\leq \alpha^2 \left\| \boldsymbol{s}_{k-1} - \boldsymbol{s}_{k-2} \right\|_p \leq \ldots \leq \alpha^k \left\| \boldsymbol{s}_1 - \boldsymbol{s}_0 \right\|_p
\end{aligned}$$

- Intuitively, we can say that $\boldsymbol{s}_* = \lim_{k \to \infty} \boldsymbol{s}_k$ for some $\boldsymbol{s}_*$, hence $\boldsymbol{s}_* = f(\boldsymbol{s}_*)$. Technically, it involes of showing domain of f is complete and sequence $\boldsymbol{s}_k$ is a Cauchy sequence.

$\square$

Step 2

T_π is a γ-contraction with $p = \infty$.

Proof.

$$
\begin{aligned}
|T_\pi(\boldsymbol{v})[s] - T_\pi(\boldsymbol{v}')[s]| &= |\sum_{a \in \mathcal{A}(s)} \sum_{s',r} \gamma \pi(a|s) p(s', r|s, a)(\boldsymbol{v}[s'] - \boldsymbol{v}'[s'])| \\
&\leq |\sum_{a \in \mathcal{A}(s)} \sum_{s',r} \gamma \pi(a|s) p(s', r|s, a) \max_s(\boldsymbol{v}[s] - \boldsymbol{v}'[s])| \\
&\leq |\gamma \max_s(\boldsymbol{v}[s] - \boldsymbol{v}'[s])| \\
&= \gamma \left\| \boldsymbol{v} - \boldsymbol{v}' \right\|_\infty
\end{aligned}
$$

$\square$

Similar proof for T_*.
Since T_π, T_* are contraction and there exists unique points v_π, v_*, they are unique.

(3) How. Part 3: Find optial policy

Next session.

Reference

- Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.
- The notes[2]by Dr Daniel Murfet

[1]http://therisingsea.org/notes/mast30026/lecture14.pdf