No.

Date

Reinforcement Learning

State $S = \{s_1 \dots s_n\} = \{s_i\}_{i=1}^{n}$

Action $A = \{a_1 \dots a_m\} = \{a_i\}_{i=1}^{5}$ e.g. $s_1 \xrightarrow{a_2} s_2$

State transition:

deterministic: $p(s_2 | s_1, a_2) = 1$, $p(s_3 | s_1, a_2) = 0$

stochastic: $p(s_2 | s_1, a_2) = 0.5$, $p(s_3 | s_1, a_2) = 0.5$

Tabular representation of state transition can only represent deterministic

Policy: tells the agent what actions to take at state

$\pi(a_1 | s_1) = 0$

$\pi(a_2 | s_1) = 1$

Tabular representation works for both deterministic and stochastic

Reward: show encouragement and punishment

$p(r = -1 | s_1, a_1) = 1 \dots p(r \neq -1 | s_1, a_1) = 0$

Trajectory: state-action-reward chain

Return: Sum of rewards along trajectory

connect human intuition with math, evaluate the policy

Discounted return $\gamma \in [0, 1)$

$0 + \gamma 0 + \gamma^2 0 + \gamma^3 1 + \gamma^4 1 + \dots = \gamma^3 (1 + \gamma + \dots) = \gamma^3 \frac{1}{1-\gamma}$

Role: 1) sum becomes finite 2) balance the far and near rewards

No.

Date

Markov Decision Process (MDP)

Sets: S, $A(s)$, $R(s,a)$

Prob distribution: $p(s'|s,a)$, $p(r|s,a)$

Policy: $\pi(a|s)$

Markov Property: memory less

$$p(s_{t+1} \mid a_{t+1}, s_t, \dots a_1, s_0) = p(s_{t+1} \mid a_{t+1}, s_t)$$

$$p(r_{t+1} \mid a_{t+1}, s_t, \dots a_1, s_0) = p(r_{t+1} \mid a_{t+1}, s_t)$$

MDP becomes Markov Process once ~~policy~~ is given!

$$v_1 = r_1 + \gamma(r_2 + \gamma r_3 + \dots) = r_1 + \gamma v_2$$

$$v_2 = r_2 + \gamma(r_3 + \gamma r_4 + \dots) = r_2 + \gamma v_3$$

$$\dots \dots$$

$$v_4 = r_4 + \gamma(r_1 + \gamma r_2 + \dots) = r_4 + \gamma v_1$$

The returns rely on each other — Bootstrapping!

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} + \begin{bmatrix} \gamma v_2 \\ \gamma v_3 \\ \gamma v_4 \\ \gamma v_1 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} + \gamma \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}$$

Bellman equation for this specific deterministic problem

$$v = r + \gamma P v \quad \text{(matrix-vector form)}$$

$$r = (I - \gamma P) v$$

Core idea: The value of one state relies on the state of others.

Campus

$S_t \xrightarrow{A_t} S_{t+1}, R_{t+1}$

- $S_t \to A_t$ is governed by $\pi(A_t = a \mid S_t = s)$
- $S_t, A_t \longrightarrow R_{t+1}$ is governed by $p(R_{t+1} = r \mid S_t = s, A_t = a)$
- $S_t, A_t \longrightarrow S_{t+1}$ is governed by $p(S_{t+1} = s' \mid S_t = s, A_t = a)$

discounted Return:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \cdots$$

G_t is a random variable since $R_{t+1}, R_{t+2} \cdots$ are

State-value (function)

$$V_\pi(s) = E[G_t \mid S_t = s]$$

- A conditional expectation with the condition that state starts from s
- Mean of all possible returns that can be obtained starting from a state

Bellman equation:

$$V_\pi(s) = E[R_{t+1} \mid S_t = s] + \gamma E[G_{t+1} \mid S_t = s]$$

$$= \sum_a \pi(a|s) \sum_r p(r|s,a) r + \gamma \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) V_\pi(s')$$

mean of immediate rewards; mean of future rewards

$$= \sum_a \pi(a|s) \left[\sum_r p(r|s,a) r + \gamma \sum_{s'} p(s'|s,a) V_\pi(s') \right], \quad \forall s \in S$$

It describes the relationship among the values of all states

Every state has a equation like this!

No.

Date

Derivation:

Consider a random trajectory:

$$S_t \xrightarrow{A_t} R_{t+1}, S_{t+1} \xrightarrow{A_{t+1}} R_{t+2}, S_{t+2} \xrightarrow{A_{t+2}} R_{t+3}, \dots$$

$$G_t = R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \dots)$$
$$= R_{t+1} + \gamma G_{t+1}$$

Following the def. of state value:

$$V_\pi(s) = E[G_t \mid S_t = s]$$
$$= E[R_{t+1} \mid S_t = s] + \gamma E[G_{t+1} \mid S_t = s]$$
$$\text{①} \qquad \text{②}$$

① $$E[R_{t+1} \mid S_t = s] = \sum_a \pi(a|s)\, E[R_{t+1} \mid S_t = s, A_t = a]$$

$$= \sum_a \pi(a|s) \sum_r p(r|s,a)\, r$$ mean of immediate reward

② $$E[G_{t+1} \mid S_t = s] = \sum_{s'} E[G_{t+1} \mid S_t = s, S_{t+1} = s']\, p(s'|s)$$

memoryless

$$= \sum_{s'} E[G_{t+1} \mid S_{t+1} = s']\, p(s'|s)$$

$$= \sum_{s'} V_\pi(s') \sum_a p(s'|s,a)\, \pi(a|s)$$

mean of future rewards

Rewrite Bellman equation

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

as.

$$v_\pi(s) = r_\pi(s) + \gamma \sum_{s'} p_\pi(s'|s)v_\pi(s')$$

where

$$r_\pi(s) \triangleq \sum_a \pi(a|s) \sum_r p(r|s,a)r,$$

$$p_\pi(s'|s) \triangleq \sum_a \pi(a|s)p(s'|s,a)$$

in a matrix-vector form:

$$v_\pi = r_\pi + \gamma P_\pi v_\pi$$

where $v_\pi = [v_\pi(s_1) \dots v_\pi(s_n)]^T \in R^n$

$r_\pi = [r_\pi(s_1) \dots r_\pi(s_n)]^T \in R^n$

$P_\pi \in R^{n\times n}$, $[P_\pi]_{ij} = p_\pi(s_j|s_i)$ is state transition matrix

Closed-form solution:

$$v_\pi = (I - \gamma P_\pi)^{-1} r_\pi$$

matrix inverse is hard to calculate in high dim.

Iterative solution.

$$v_{k+1} = r_\pi + \gamma P_\pi v_k$$

we can show that $v_k \to v_\pi = (I - \gamma P)^{-1} r_\pi$, $k \to \infty$

State value : the average return the agent can get starting from a state

Action value : the average return starting from a state and taking an action

$$q_\pi(s,a) = E[G_t \mid S_t = s, A_t = a]$$

$$\because E[G_t \mid S_t = s] = \sum_a E[G_t \mid S_t = s, A_t = a]\, \pi(a \mid s)$$

$$\therefore V_\pi(s) = \sum_a \pi(a \mid s)\, q_\pi(s,a)$$

According to the def of Bellman

$$q_\pi(s,a) = \sum_r p(r \mid s,a)\, r + \gamma \sum_{s'} p(s' \mid s,a)\, V_\pi(s')$$

All the action can be computed, even if current policy won't adopt it

No.

Date

Optimal policy : if $v_{\pi_1}(s) \geq v_{\pi_2}(s) \quad \forall s \in S$

then π_1 is better than π_2

A policy π^* is optimal if $v_{\pi^*}(s) \geq v_{\pi}(s) \quad \forall s \in S, \pi$

Bellman optimality equation (elementwise form)

$$v(s) = \max_{\pi} \sum_{a} \pi(a|s) q(s,a) \quad s \in S$$

$$v = \max_{\pi} (r_\pi + \gamma P_\pi v) \quad \text{(matrix-vector form)}$$

$\|f(x_1) - f(x_2)\| \leq \gamma \|x_1 - x_2\|$ where $\gamma \in [0,1)$

Theorem: Contraction Mapping Theorem.

$\forall$ equation s.t. $x = f(x)$, if f is a contractive mapping, then

① Existence : $\exists$ fixed point x^* s.t. $f(x^*) = x^*$

② Uniqueness : x^* is unique

③ Algorithm : Consider a sequence $\{x_k\}$ where $x_{k+1} = f(x_k)$, then $x_k \to x^*$ as $k \to \infty$. The convergence rate is exponentially fast

$$v = f(v) = \max_{\pi} (r_\pi + \gamma P_\pi v)$$

Theorem: Contraction Property

$f(v)$ is a contraction mapping satisfying

$$\|f(v_1) - f(v_2)\| \leq \gamma \|v_1 - v_2\| \quad \text{(same for } q_\pi(s,a))$$

where γ is the discount rate

Theorem : Optimal Policy

suppose v^* is the unique solution to $v = \max_{\pi} (r_\pi + \gamma P_\pi v)$

$v^* \geq v_\pi \quad \forall \pi$ (unique)

$\pi^* = \arg\max_{\pi} (r_\pi + \gamma P_\pi v^*)$ is the optimal policy

(not necessarily unique)

KOKUYO

Theorem: Greedy Optimal Policy

$\forall s \in S$ the deterministic greedy policy

$$\pi^*(a|s) = \begin{cases} 1 & a = a^*(s) \\ 0 & a \neq a^*(s) \end{cases}$$

is an optimal policy solving BOE. here

$$a^*(s) = \arg\max_a q^*(a,s)$$

where $q^*(s,a) := \sum_r p(r|s,a)\, r + \gamma \sum_{s'} p(s'|s,a)\, v^*(s')$

Value iteration

$$u_0 \xrightarrow{PU} \pi_1' \xrightarrow{VU} u_1 \xrightarrow{PU} \pi_2' \longrightarrow \cdots$$

Policy iteration

$$\pi_0 \xrightarrow{PE} v_{\pi_0} \xrightarrow{PI} \pi_1 \xrightarrow{PE} v_{\pi_1} \xrightarrow{PI} \pi_2 \cdots$$

PE = Policy Evaluation find best v_{π_k}

PI = policy improvement find best π_{k+1}

Q1. In PE, how to get v_{π_k}?

A1: By Solving Bellman equation in iteration

It's a small iteration in a big (policy) iteration

Q2: why π_{k+1} better than π_k,

Lemma: Policy Improvement

If $\pi_{k+1} = \arg\max_\pi (r_\pi + \gamma P_\pi v_{\pi_k})$, then $v_{\pi_{k+1}} \geq v_{\pi_k} \quad \forall k$

Q3. why iterative algorithm reach optimal policy?

Theorem: Convergence of Policy iteration.

$\{v_{\pi_k}\}_{k=0}^{\infty}$ converges to optimal state value v^*, hence

$\{\pi_k\}_{k=0}^{\infty}$ converges to π^*

	Policy iteration	Value iteration
1) Policy	π_0	N/A
2) Value	$V_{\pi_0} = r_{\pi_0} + \gamma P_{\pi_0} V_{\pi_0}$	$V_0 := V_{\pi_0}$
3) Policy	$\pi_1 = \arg\max_{\pi} (r_\pi + \gamma P_\pi V_{\pi_0})$	$\pi_1 = \arg\max_{\pi} (r_\pi + \gamma P_\pi V_0)$ same so far
4) Value	$V_{\pi_1} = r_{\pi_1} + \gamma P_{\pi_1} V_{\pi_1}$ iteration inside (∞ step)	$V_1 = r_{\pi_1} + \gamma P_{\pi_1} V_0$ one step
5) Policy	$\pi_2 = \arg\max_{\pi} (r_\pi + \gamma P_\pi V_{\pi_1})$	$\pi_2' = \arg\max_{\pi} (r_\pi + \gamma P_\pi V_1)$

For solving $r_{\pi_1} = r_{\pi_1} + \gamma P_{\pi_1} V_{\pi_1}$

$V_{\pi_0} = V_0$

value iteration V_1 ← $V_{\pi_1}^{(1)} = r_{\pi_1} + \gamma P_{\pi_1} V_{\pi_1}^{(0)}$

⋮

truncated policy iteration $\bar{V}_1$ ← $V_{\pi_1}^{(j)} = r_{\pi_1} + \gamma P_{\pi_1} V_{\pi_1}^{(j-1)}$

⋮] truncated

policy iteration V_{π_1} ← $V_{\pi_1}^{(\infty)} = r_{\pi_1} + \gamma P_{\pi_1} V_{\pi_1}^{(\infty)}$ idealistic

Monte Carlo Method: .

Model free, estimate expectation base on DATA

Generalized policy iteration: switching frequently between PI and PE

ε-greedy policy

$$\pi(a|s) = \begin{cases} 1 - \frac{\varepsilon}{|A(s)|}(|A(s)|-1), & \text{for greedy action} \\ \frac{\varepsilon}{|A(s)|} & \text{for other actions} \end{cases}$$

Balance between exploration and exploitation

$$\pi_{k+1}(s) = \arg\max_{\pi \in \Pi_\varepsilon} \sum_a \pi(a|s) q_{\pi_k}(s,a)$$

similar to dp, reuse state-action pairs in episode for data efficiency

MC Exploring Starts	MC ε-greedy
first visit	every visit
deterministic greedy	ε-greedy

No.

Date

Stochastic approximation (SA)

it doesn't require to know the expression of objective function nor its derivativ

Robbins-Monro (RM)

$$w_{k+1} = w_k - a_k \tilde{g}(w_k, \eta_k)$$

w - $E[x]$ noise · $E[x] - x$ ·

where $\tilde{g}(w_k, \eta_k) = g(w_k) + \eta_k$

normally we want to find $g(w) = \nabla_w J(w) = 0$, but now we don't have J.

so we extract the information purely from data.

input sequence : $\{w_k\}$

output sequence : $\{\tilde{g}(w_k, \eta_k)\}$

Theorem: RM Theorem.

If 1) $0 < c_1 \leq \nabla_w g(w) \leq c_2$ for all w;

2) $\sum_{k=1}^{\infty} a_k = \infty$ and $\sum_{k=1}^{\infty} a_k^2 < \infty$;

3) $E[\eta_k | \mathcal{H}_k] = 0$ and $E[\eta_k^2 | \mathcal{H}_k] < \infty$

where $\mathcal{H}_k = \{w_k, w_{k-1} \dots\}$, then w_k converges with probability 1 (w.p.1) to the root w^* satisfying $g(w^*) = 0$

Explanation : ① : g is monotonically increasing, which ensures root of $g(w)=0$ exists and unique.
The gradient is bounded from above.

② . $\sum_{k=1}^{\infty} a_k^2 < \infty$: $a_k \xrightarrow{k\to\infty} 0$.

$\sum_{k=1}^{\infty} a_k = \infty$: a_k doesn't converge too fast

③ : let $\{\eta_k\}$ be iid with $E[\eta_k]=0$ and $E[\eta_k^2] < \infty$
η_k doesn't need to be Gaussian

2.1 $\sum_{k=1}^{\infty} a_k^2 < \infty \longrightarrow (a_k \xrightarrow{k\to\infty} 0)$

$\therefore w_{k+1} - w_k = -a_k(\tilde{g}(w_k, \eta_k)) \to 0$ we need this for w_k converge

2.2. $\sum_{k=1}^{\infty} a_k = \infty$

$$w_\infty - w_1 = \sum_{k=1}^{\infty} a_k \tilde{g}(w_k, \eta_k)$$

suppose $w_\infty = w^*$, if $\sum_{k=1}^{\infty} a_k < \infty$, $w_\infty - w_1$ may be bounded, then we can't arbitrarily select w_1 far from w^*

$\boxed{a_k = \frac{1}{k}}$ satisfy 2.1 and 2.2.

- $\lim_{n\to\infty} \left(\sum_{k=1}^{\infty} \frac{1}{k} - \ln \right) = \kappa \approx 0.577$ (Euler-Mascheroni constant)

- $\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6} < \infty$ (Basel problem)

In practise we select a_k as a very small constant (which breaks ②) but still effective.

No.

Date

Mean estimation: compute $E[X]$ using $\{x_k\} \in X$ (iterative)

$$w_{k+1} = w_k + \frac{1}{k}(w_k - x_k)$$ don't need to wait for all sampling

RM algorithm: solve $g(w) = 0$ using noisy $\{\tilde{g}(w_k, \eta_k)\}$

without expression of $g(w)$

$$w_{k+1} = w_k - a_k \tilde{g}(w_k, \eta_k)$$

SGD algorithm: minimize $J(w) = E[f(w, X)]$ using $\{\nabla_w f(w_k, x_k)\}$ sampling of gradient

$$w_{k+1} = w_k - a_k \nabla_w f(w_k, x_k)$$

Question: Is the convergence of SGD slow or random?

relative error $\delta_k = \dfrac{|\nabla_w f(w_k, x_k) - E[\nabla_w f(w_k, X)]|}{|E[\nabla_w f(w_k, X)]|}$ stochastic gradient, batch (true) gradient

$$= \frac{|\nabla_w f(w_k, x_k) - E[\nabla_w f(w_k, X)]|}{|E[\nabla_w f(w_k, X)] - E[\nabla_w f(w^*, X)]|}$$

Based on Lagrange Mean Value theorem.

$$= \frac{|\nabla_w f(w_k, x_k) - E[\nabla_w f(w_k, X)]|}{|E[\nabla_w^2 f(\hat{w}_k, X)(w_k - w^*)]|} \quad \hat{w}_k \in [w_k, w^*]$$

Suppose $\nabla_w^2 f \geq c > 0$ (strictly convex)

$$|E[\nabla_w^2 f(\hat{w}_k, X)(w_k - w^*)]| = |E[\nabla_w^2 f(\hat{w}_k, X)]||(w_k - w^*)| \geq c|w_k - w^*|$$

$$\therefore \quad \delta_k \leq \frac{|\nabla_w f(w_k, x_k) - E[\nabla_w f(w_k, X)]|}{c|w_k - w^*|}$$

when $|w_k - w^*|$ is large, δ_k is small: SGD behaves like GD

small: more randomness in the neighborhood of w^*

KOKUYO

We may encounter a deterministic formulation of SGD.

$$\min_w J(w) = \frac{1}{n}\sum_{i=1}^{n} f(w, x_i),$$

where x comes from a set of real numbers $\{x_i\}_{i=1}^n$

here GD: $w_{k+1} = w_k - a_k \frac{1}{n}\sum_{i=1}^{n} \nabla_w f(w_k, x_i)$,

SGD: $w_{k+1} = w_k - a_k \nabla_w f(w_k, x_k)$

We can introduce a random variable manually
to convert the deterministic formulation to a stochastic formulation

- Suppose X is a random variable defined on $\{x_i\}_{i=1}^n$, and its prob distribution is uniform: $p(X = x_i) = \frac{1}{n}$

then $\frac{1}{n}\sum_{i=1}^{n} f(w, x_i) = E[f(w, X)]$

☆ mini-batch GD
- $n=1$ → SGD
- $m=n$ → ≒ BGD. but MBGD may use a value multiple times

For $J(w) = \frac{1}{2n}\sum_{i=1}^{n} \|w - x_i\|^2$

$w_{k+1} = w_k - a_k(w_k - \bar{x})$ BGD

$w_{k+1} = w_k - a_k(w_k - \bar{x}_k^{(m)})$ MBGD

$w_{k+1} = w_k - a_k(w_k - x_k)$ SGD

No.

Date

Temporal-Difference (TD) Algorithm

TD error δ_t

$$\begin{cases} V_{t+1}(s_t) = V_t(s_t) - \alpha_t(s_t)\left[V_t(s_t) - \underbrace{[r_{t+1} + \gamma V_t(s_{t+1})]}_{\text{TD target } \bar{v}_t}\right] \\ V_{t+1}(s) = V_t(s), \quad \forall s \neq s_t \end{cases}$$

$V_t(s_t)$ is the estimated state value of $v_\pi(s_t)$

To algorithm drives $V(s_t)$ towards $\bar{v}_t$

$$V_{t+1}(s_t) \overset{-\bar{v}_t}{=} V_t(s_t) - \alpha_t(s_t)[V_t(s_t) - \bar{v}_t] - \bar{v}_t$$

$$|V_{t+1}(s_t) - \bar{v}_t| = |1 - \alpha_t(s_t)|\,|V_t(s_t) - \bar{v}_t|$$

Since $0 < 1 - \alpha_t(s_t) < 1$

therefore $|V_{t+1}(s_t) - \bar{v}_t| < |V_t(s_t) - \bar{v}_t|$

TD error:

$$\delta_t = V(s_t) - [r_{t+1} + \gamma V(s_{t+1})]$$

Difference between two consequent time steps

It reflects the deficiency between v_t and v_π:

$$E[\delta_{\pi,t} \mid S_t = s_t] = v_\pi(s_t) - E[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s_t] = 0$$

If $v_t = v_\pi$, $\delta = 0$

Can be interpreted at new information. from (s_t, r_{t+1}, s_{t+1})

based on new experience we update state value v

No.

Date

Bellman expectation equation:

$$V_\pi(s) = E[R + \gamma G \mid S = s], \quad s \in S$$

where G is discounted return

Since:

$$E[G \mid S=s] = \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) V_\pi(s') = E[V_\pi(S') \mid S=s]$$

where S' is the next state.

$$V_\pi(s) = E[R + \gamma V_\pi(S') \mid S=s], \quad s \in S$$

With RM, we define

$$g(v(s)) = v(s) - E[R + \gamma V_\pi(s') \mid s],$$

$$\tilde{g}(v(s)) = v(s) - [r + \gamma v_\pi(s')]$$

$$= g(v(s)) + \eta$$

With RM

$$v_{k+1}(s) = v_k(s) - a_k(v_k(s) - [r_k + \gamma V_\pi(s'_k)]), \quad k \in \mathbb{N}$$

Note that ① we must have experience set $\{(s, r, s')\}$ for $k \in \mathbb{N}$

we can replace it with trajectory $\{(s_t, r_{t+1}, s_{t+1})\}$, when visit we update s

② we don't know $V_\pi(s'_k)$

we first use an estimate $V_k(s'_k)$

Theorem: TD convergence

$v_t(s)$ converges to $V_\pi(s)$ $\forall s \in S$ as $t \to \infty$ if ① $\sum_t a_t(s) = \infty$

② $\sum_t a_t^2(s) < \infty$

No.

Date

TD / Sarsa learning	MC learning
Online: Update state / action value immediately after receiving a reward	Wait until an episode has been completely collected
Continuing tasks and Episodic tasks	Only episodic tasks that has terminate
Bootstrapping: require initial guess (may cause high bias)	Non-bootstrapping: directly estimate
Low estimation variance: less random variable	High estimation variance: we need samples of $R_{t+1} + \gamma R_{t+2} + \cdots$, $\lvert A \rvert^L$ possible episodes

Sarsa: (State - Action - Reward - State - Action) (s_t a_t r_{t+1} s_{t+1} a_{t+1})

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - a_t(s_t, a_t)\left[q_t(s_t, a_t) - [r_{t+1} + \gamma q_t(s_{t+1}, a_{t+1})]\right]$$

$$q_{t+1}(s, a) = q_t(s, a) \quad \forall (s, a) \neq (s_t, a_t)$$

$q_t(s_t, a_t)$ is an estimate of $q_\pi(s_t, a_t)$

Another Bellman equation expressed in terms of action values:

$$q_\pi(s, a) = \mathbb{E}[R + \gamma q_\pi(S', A') \mid s, a], \forall s, a$$

Sarsa has convergence with similar condition to TD

No.

Date

Expected Sarsa:

$$\begin{cases} q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t)\left[q_t(s_t, a_t) - (r_{t+1} + \gamma E[q_t(s_{t+1}, A)])\right] \\ q_{t+1}(s, a) = q_t(s, a) \end{cases}$$

where

$$E[q_t(s_{t+1}, A)] = \sum_a \pi_t(a|s_{t+1}) q_t(s_{t+1}, a) = v_t(s_{t+1})$$

is expected value of $q_t(s_{t+1}, a)$ under policy π

Need more computation, but reduce estimation variance

It solves another expression for Bellman equation

$$q_\pi(s, a) = E[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = a],$$

Sarsa $\longrightarrow$ $G_t^{(1)} = R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1})$

(low variance, high bias) $G_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 q_\pi(S_{t+2}, A_{t+2})$.

.....

n-step Sarsa: (unifies MC and Sarsa)

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^n q_\pi(S_{t+n}, A_{t+n}),$$

.....

MC $\longrightarrow$ $G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \cdots\cdots$

(large variance, low bias) $G_t^{(k)}$ are all the same, difference lies in how to decompose it.

- n-step Sarsa is:

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t)\left[q_t(s_t, a_t) - [r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^n q_t(s_{t+n}, a_{t+n})]\right]$$

not online nor offline

Campus

No.

Date

Q-learning

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t)\left[q_t(s_t, a_t) - \left[r_{t+1} + \gamma \max_{a \in A} q_t(s_{t+1}, a)\right]\right],$$

$$q_{t+1}(s, a) = q_t(s, a), \quad \forall (s, a) \neq (s_t, a_t)$$

Aims to solve a Bellman optimality equation

$$q(s, a) = E\left[R_{t+1} + \gamma \max_{a} q(S_{t+1}, a) \mid S_t = s, A_t = a\right], \quad \forall s, a$$

Behavior Policy : generate experience samples

Target Policy : update toward optimal policy

On-Policy : behavior policy is the same as target policy

off-Policy : they're different

Advantage of off-policy:

Search for optimal policy based on experience samples generated by other polic

behavior policy can be selected to be exploratory, which is better

Check for on-policy or off-policy

1. check the math
2. check what is required to implement the algorithm.

No.

Date

1. Sarsa is on-policy

① It solves Bellman equation with given policy π:

$$q_\pi(s,a) = E[R + \gamma q_\pi(S', A') \mid s, a], \quad \forall s, a.$$

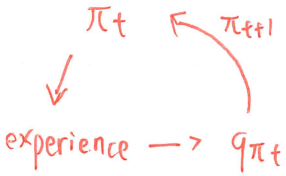
where $R \sim p(R|s,a)$, $S' \sim p(S'|s,a)$, $A' \sim \pi(A'|S')$

② the algorithm is

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t)[q_t(s_t, a_t) - [r_{t+1} + \gamma q_t(s_{t+1}, a_{t+1})]]$$

which requires $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$

If (s_t, a_t), r_{t+1} and s_{t+1} doesn't depend on any policy



π is both target and behavior policy

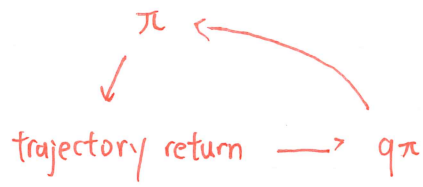
2. Monte Carlo is on-policy.

① It estimates action value with given policy π

$$q_\pi(s,a) = E[R_{t+1} + \gamma R_{t+2} + \cdots \mid S_t = s, A_t = a], \quad \forall s, a.$$

② Its implementation is:

$$q(s,a) = r_{t+1} + \gamma r_{t+2} + \cdots$$



π is both target and behavior policy

3. Q-learning is off-policy no policy π involved

① It aims to solve Bellman optimality equation

$$q(s,a) = E[R_{t+1} + \gamma \max_a q(S_{t+1}, a) \mid S_t = s, A_t = a], \forall s, a$$

② the algorithm is

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t)[q_t(s_t, a_t) - [r_{t+1} + \gamma \max_{a \in A} q_t(s_{t+1}, a)]]$$

which requires $(s_t, a_t, r_{t+1}, s_{t+1})$

Behavior policy generate a_t from s_t can be anything

Target policy is optimal policy

Unified expression TD target

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t)[q_t(s_t, a_t) - \bar{q}_t]$$

Sarsa. $\bar{q}_t = r_{t+1} + \gamma q_t(s_{t+1}, a_{t+1})$

n-step Sarsa $\bar{q}_t = r_{t+1} + \gamma r_{t+2} + \dots + \gamma^n q_t(S_{t+n}, a_{t+n})$

Expected Sarsa $\bar{q}_t = r_{t+1} + \gamma \sum_a \pi_t(a|s_{t+1}) q_t(s_{t+1}, a)$

Q-learing $\bar{q}_t = r_{t+1} + \gamma \max_a q_t(s_{t+1}, a)$

Monte Carlo $\bar{q}_t = r_{t+1} + \gamma r_{t+2} + \dots$

For MC, if $\alpha = 1$, $q_{t+1}(s_t, a_t) = \bar{q}_t$

No.

Date

All those algorithms can be viewed as stochastic approximation solving Bellman equation or Bellman optimality equation

Sarsa $\quad BE: q_\pi(s,a) = E[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$

n-step Sarsa $\quad BE: q_\pi(s,a) = E[R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^n q_\pi(S_{t+n}, a_{t+n}) \mid S_t = s, A_t = a)$

Expected Sarsa $\quad BE: q_\pi(s,a) = E[R_{t+1} + \gamma E_{A_{t+1}}[q_\pi(S_{t+1}, A_{t+1})] \mid S_t = s, A_t = a]$

Q-learning $\quad BOE: q(s,a) = E[R_{t+1} + \max_a q(S_{t+1}, a) \mid S_t = s, A_t = a]$

Monte Carlo $\quad BE: q_\pi(s,a) = E[R_{t+1} + \gamma R_{t+2} + \cdots \mid S_t = s, A_t = a]$

No.

Date

Value function approximation

Objective function

I. Uniform distribution

Treat all states equally, set prob of each state as $\frac{1}{|S|}$

$$J(w) = E[(v_\pi(s) - \hat{v}(s,w))^2] = \frac{1}{|S|}\sum_{s\in S}(v_\pi(s) - \hat{v}(s,w))^2$$

II Stationary distribution.

Describe long-run behaviour

let $\{d_\pi(s)\}_{s\in S}$ denote the stationary distribution of Markov process under policy π. $d_\pi(s) \geq 0$ and $\sum_{s\in S} d_\pi(s) = 1$

$$J(w) = E[(v_\pi(s) - \hat{v}(s,w))^2] = \sum_{s\in S} d_\pi(s)(v_\pi(s) - \hat{v}(s,w))^2$$

↑ the frequency of visiting when agent reach stationary

$$d_\pi(s) \approx \frac{n_\pi(s)}{\sum_{s'\in S} v_\pi(s')}$$

from Bellman equation $v_\pi = r_\pi + \gamma P_\pi v_\pi$

$$d_\pi^T = d_\pi^T P_\pi$$

It can be calculated when d_π is the eigenvector for eigenvalue 1

SGD:

$$w_{t+1} = w_t + \alpha_t (v_\pi(s_t) - \hat{v}(s_t, w_t)) \nabla_w \hat{v}(s_t, w_t).$$

v_π is to be estimated, we replace v_π with approximation

1. MC — discounted return

$$w_{t+1} = w_t + \alpha_t (g_t - \hat{v}(s_t, w_t)) \nabla_w \hat{v}(s_t, w_t)$$

2. TD — TD target

$$w_{t+1} = w_t + \alpha_t [r_{t+1} + \gamma \hat{v}(s_{t+1}, w_t) - \hat{v}(s_t, w_t)] \nabla_w \hat{v}(s_t, w_t)$$

To select $\hat{v}(s, w)$

1. linear function

$$\hat{v}(s, w) = \phi^T(s) w,$$

where $\phi(s)$ is a feature vector, can be polynomial basis, Fourier basis...

2. nonlinear

$$s \rightarrow \boxed{w} \longrightarrow \hat{v}(s, w)$$

In linear case, we have TD-linear

$$w_{t+1} = w_t + \alpha_t [r_{t+1} + \gamma \phi^T(s_{t+1}) w_t - \phi^T(s_t) w_t] \phi(s_t)$$

Objective functions

1. True value error: $J_E(w) = E[(v_\pi(s) - \hat{v}(s, w))^2] = \|\hat{v}(w) - v_\pi\|_D^2$
2. Bellman error:

$$J_{BE}(w) = \|\hat{v}(w) - (r_\pi + \gamma P_\pi \hat{v}(w))\|_D^2 \doteq \|\hat{v}(w) - T_\pi(\hat{v}(w))\|_D^2$$

3. Projected Bellman error

$$J_{PBE}(w) = \|\hat{v}(w) - M T_\pi(\hat{v}(w))\|_D^2$$

Sarsa:

$$w_{t+1} = w_t + \alpha_t [r_{t+1} + \gamma \hat{q}(s_{t+1}, a_{t+1}, w_t) - \hat{q}(s_t, a_t, w_t)] \nabla_w \hat{q}(s_t, a_t, w_t)$$

with linear function approximation, $\hat{q} = \phi^T w$

Q-learning (on-policy)

$$w_{t+1} = w_t + \alpha_t [r_{t+1} + \gamma \max_{a \in A(s_{t+1})} \hat{q}(s_{t+1}, a, w_t) - \hat{q}(s_t, a_t, w_t)] \nabla_w \hat{q}(s_t, a_t, w_t)$$

Deep Q-learning

TD target ↗ $\nabla \hat{q}$

$$J(w) = E[(R + \gamma \max_{a \in A(S')} \hat{q}(S', a, \underline{w}) - \hat{q}(S, A, \underline{w}))^2],$$

where S, A, R, S' are random variables

This is Bellman optimality error, since

$$q(s,a) = E[R_{t+1} + \gamma \max_{a \in A(S_{t+1})} q(S_{t+1}, a) \mid S_t = s, A_t = a], \quad \forall s, a$$

We introduce two network

1. main network $\hat{q}(S, a, w)$ and 2. target network $\hat{q}(S, a, w_T)$

when w_T is fixed, the gradient can be obtained as

$$\nabla_w J = E[(R + \gamma \max_{a \in A(S')} \hat{q}(S', a, w_T) - \hat{q}(S, A, w)) \nabla_w \hat{q}(S, A, w)]$$

we only update w every iteration and treat w_T as constant, then set $w_T = w$ every C iterations

No.

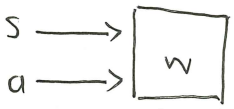
Date

Technique:

① Two networks:

w and w_T are set to be same initially

In every iteration, we draw a mini-batch of samples $\{(s,a,r,s')\}$ from the replay buffer



$$\longrightarrow y_T = r + \gamma \max_{a' \in A(s')} \hat{q}(s', a', w_T)$$

then we minimize TD error $(y_T - \hat{q}(s,a,w))^2$

② Experience replay.

we don't use the collected experience samples in the original order

we store them in a set replay buffer $B = \{(s,a,r,s')\}$

When training, we draw a mini-batch from B in uniform distribution

Break the correlation between consequent samples

Why Deep Q-learning involves distribution

The objective function in the deep case is a scalar average over all (S,A). While in tabular case we aims to solve a set of equations for all (s,a) (Bellman Optimality)

No.

Date

Policy Function Approximation (value-based to policy-based)

policy: $\pi(a|s;\theta)$ (from tabular to NN)

how to update policies?

when represented by table: directly change the entries

when represented by parameterized function: change parameter θ

$$\bar{v}_\pi = \sum_{s\in S} d(s) v_\pi(s)$$

$$= d^T v_\pi$$

d_0 ① states are equally important, $d_0(s) = \frac{1}{|S|}$

② only interested in specific state s_0

$$d_0(s_0) = 1 \quad d_0(s \neq s_0) = 0$$

③ if d depends on π, then

d can be selected as $d_\pi(s)$ (stationary distribution)

If one state is frequently visited in the long Run,
it's more important and deserves more weight

GD: $$\nabla_\theta J(\theta) = \sum_{s\in S} \eta(s) \sum_{a\in A} \nabla_\theta \pi(a|s,\theta) q_\pi(s,a)$$

- $J(\theta)$ can be $\bar{v}_\pi$, $\bar{r}_\pi$ or $\bar{v}_\pi^0$
- "=" may be strict equally, approximation or proportional to
- η is a distribution or weight of states.

No.

Date

$$\nabla_\theta J(\theta) = E[\nabla_\theta \ln \pi(A|S,\theta) q_\pi(S,A)]$$

$\propto$ with SGD

$$\nabla_\theta \ln \pi(a|s,\theta) = \frac{\nabla_\theta \pi(a|s,\theta)}{\pi(a|s,\theta)}$$

$$\therefore \nabla_\theta \pi(a|s,\theta) = \pi(a|s,\theta) \nabla_\theta \ln \pi(a|s,\theta)$$

$$\nabla_\theta J = E_{s\sim d}\left[\sum_a \pi(a|s,\theta) \nabla_\theta \ln \pi(a|s,\theta) q_\pi(s,a)\right]$$

$$= E_{s\sim d, A\sim\pi}\left[\nabla_\theta \ln \pi(A|s,\theta) q_\pi(s,A)\right]$$

with softmax, we guarantee $\pi(a|s,\theta) > 0$.
the parameterized policy is stochastic and exploratory

How to sample S, A?

- $S \sim d$. d is the long-run behavior under π
- $A \sim \pi(A|s,\theta)$, a_t should be sampled following $\pi(\theta_t)$ at s_t

on-policy

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \ln \pi(a_t|s_t,\theta_t) q_t(s_t,a_t)$$

$$\theta_{t+1} = \theta_t + \alpha \beta_t \nabla_\theta \pi(a_t|s_t,\theta_t)$$

we are optimizing $\pi(a_t|s_t,\theta_t)$ here!

Campus

when $\alpha \cdot \beta_t$ is sufficiently small.

① If $\beta_t > 0$, $\pi(a_t|s_t,\theta_{t+1}) > \pi(a_t|s_t,\theta_t)$

② If $\beta_t < 0$, $\pi(a_t|s_t,\theta_{t+1}) < \pi(a_t|s_t,\theta_t)$

$$\pi(a_t|s_t,\theta_{t+1}) \approx \pi(a_t|s_t,\theta_t) + \epsilon \nabla_\theta \pi(a_t|s_t,\theta_t)^T(\theta_{t+1}-\theta_t)$$

$$= \pi(a_t|s_t,\theta_t) + \underbrace{\alpha\beta_t \|\nabla_\theta \pi(a_t|s_t,\theta_t)\|^2}_{>0}$$

$$\beta_t = \frac{q_t(s_t,a_t)}{\pi(a_t|s_t,\theta_t)}$$

β_t balance exploration and exploitation.

① $\beta_t \propto q_t(s_t,a_t)$

enhance actions with greater value

② β_t inversely $\propto \pi(a_t|s_t,\theta_t)$

explore actions with low probabilities.

If $q_\pi(s_t,a_t)$ is approximated by MC estimation,
we call it REINFORCE

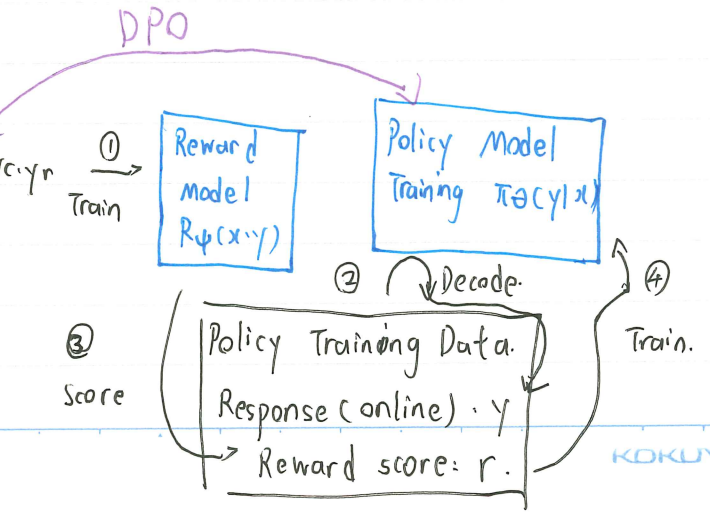
* DPO and PPO.

Preference Data:

PPO:

Responses (offline): y_c, y_r

Ranking: $y_c > y_r | x$



No.

Date