Reinforcement Learning

Some definations

Agent → The agent is the one who takes actions.

Action → $A = \{a_1, a_2, \ldots a_n\}$

Finite Set of all actions available to agent

Environment → The world through which agent moves. Everything outside agent

$E(S_t, a_t) = S_{t+1}, R_{t+1}$

- S_t: current state
- a_t: current action
- S_{t+1}: next state
- R_{t+1}: reward for being in this state

State (S) → A concrete and immediate situation in which the agent is in.

Generally represented as a vector $S \in \mathbb{R}^d$

Reward (R) → Feedback to the agent.

Reward can be immediate or delayed.

$R \in \mathbb{R}^1$ ↓ numerical value

S, R → Random Variables.

Discount Factor (γ) → How much important future rewards are to the agent.

$\gamma = 1$ → Future reward as important as immediate

$\gamma = 0$ → Don't care about future. Myopic Agent ↓ Far-Sighted Agent

Agent looks only one step ahead to maximize reward.

Example → Hill climbing Algorithm (local, greedy, can trap in local sub-optimal solution)

$0 \leq \gamma \leq 1$

Total Global Reward at time t

$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

k → no. of steps in future the agent see to maximize its reward.

$G_t \rightarrow$ Sum of infinite terms

→ finite if R_{t+R+1} is finite and $\gamma < 1$

$G_t = R_{t+1} + \gamma G_{t+1}$

Bellman Expectation Equation

Policy (π) → $\pi(s) = a$ $\pi(a|s)$

Strategy that the agent employs to determine the next action based on current state.

Start State (s_0)

Terminal State (s_T) → End state when only action is "Exit"

$T \rightarrow$ horizon

Episode → One trial run from start to terminal state

$$s_0, a_0, s_1, R_1, a_1, s_2, R_2, a_2, \ldots, s_T, R_T, \text{"Exit"}$$

Transition Function $P(s_{t+1} | s_t, a_t) \rightarrow$

State Transition Probability

$$p(s_{t+1} | s_t, a_t) = \sum_R p(s_{t+1}, R | s_t, a_t)$$

Optimal Policy (π^*) → The goal of RL is to find optimal policy.

$$\pi^* = \arg\max_{\pi} \mathbb{E}[G_T | \pi]$$

where G_T → Total global reward at terminating time T

$$G_T = \sum_{k=0}^{T-1} \gamma^k R_{k+1}(s_k, a_k, s_{k+1})$$

Value Function $V^{\pi}(s)$: value (expected return) of being in the state s.

Expected return when starting from state s and following policy π

$$V^{\pi}(s) = \mathbb{E}_{\pi}[G_t | s_t = s, \pi] = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, s_t = s, \pi\right]$$

more specifically if we start at $t=0$ and terminate at $t=H$ → Horizon.

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{H-1} \gamma^t R(s_t, a_t, s_{t+1}) \,\middle|\, \pi, s_0 = s\right]$$

Episode → $s_0, a_0, R_1, s_1, a_1, R_2 \ldots, R_H, s_H, a_H =$ "Exit"

$R_H \equiv R(s_{H-1}, a_{H-1}, s_H)$

Optimal state value function

$$V^{\pi^*}(s) = \max_{\pi} V^{\pi}(s) \qquad \forall\, s \in \mathcal{S}$$

Q-value or action-value (Q)
Long term return for state s, taking action a under policy π

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[G_t \mid s_t = s, a_t = a\right]$$

$$= \mathbb{E}_{\pi}\left[\sum_{t=0}^{H-1} \gamma^t R(s_t, a_t, s_{t+1}) \,\middle|\, s, a\right]$$

If $V^{\pi^*}(s)$ is given starting from s, optimal policy is

$$\mathbb{E}_{s_{t+1} \sim T(s_{t+1} \mid s_t, a_t)}\left[V^{\pi^*}(s_{t+1})\right]$$

choose the ~~next~~ ~~st~~ current action a_t s.t the next state s_{t+1} has max. $V^{\pi^*}(s_{t+1})$

more computationally extensive

$$V^{\pi}(s) = \sum_a \pi(a|s)\, Q^{\pi}(s,a)$$

If $Q^{\pi}(s,a)$ is given optimal policy is

$$\arg\max_a Q^{\pi}(s,a)$$

$$V^{\pi}(s) = \max_a Q^{\pi}(s,a)$$

more memory extensive,

$$Q^{\pi^*}(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

$$Q^{\pi^*}(s,a) =$$

$$\mathbb{E}\left[R_{t+1} + \gamma V^{\pi^*}(s_{t+1}) \,\middle|\, s_t, a_t\right]$$

Bellman Equations

value iteration

$$V^{\pi}(s) = \mathop{\mathbb{E}}_{\pi}\left[G_t \mid S_t = s \right]$$

we know

$$G_t = R_{t+1} + \gamma G_{t+1}$$

current reward — future reward

$$V^{\pi}(s) = \mathop{\mathbb{E}}_{\pi}\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s \right]$$

$$\mathop{\mathbb{E}}_{\pi}\left[R_{t+1} \mid S_t = s \right] =$$

$$\sum_a \pi(a \mid S_t = s) \sum_{s'} p(S_{t+1} = s' \mid S_t = s, a_t = a)$$

$$R(s, a, s')$$

$$\mathop{\mathbb{E}}_{\pi}\left[\gamma G_{t+1} \mid S_t = s \right] =$$

$$\sum_a \pi(a \mid S_t = s) \sum_{s'} p(S_{t+1} = s' \mid s, a)$$

$$\gamma \mathop{\mathbb{E}}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \mid S_{t+1} = s' \right]$$

$$V^{\pi}(s) = \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a)$$

$$\left[R(s, a, s') + \gamma V^{\pi}(s') \right]$$

$$V_t^{\pi *}(s) = \max_a \sum_{s'} p(s' \mid s, a)$$

$$\left[R(s, a, s') + \gamma V_{t-1}^{\pi^*}(s') \right]$$

Q-value Iteration

~~Policy~~ Iteration

$$Q^{\pi}(s, a) = \mathop{\mathbb{E}}_{\pi}\left[G_t \mid S_t = s, a_t = a \right]$$

$$Q^{\pi}(s, a) = \mathop{\mathbb{E}}_{\pi}\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, a_t = a \right]$$

$$\mathop{\mathbb{E}}_{\pi}\left[R_{t+1} \mid s, a \right] = \sum_{s'} p(s' \mid s, a) R(s, a, s')$$

$$\mathop{\mathbb{E}}_{\pi}\left[\gamma G_{t+1} \mid s, a \right] =$$

$$\sum_{s'} p(s' \mid s, a) \gamma \mathop{\mathbb{E}}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \right]$$

$$= \sum_{s'} p(s' \mid s, a) \gamma \sum_{a'} \mathop{\mathbb{E}}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k R_{k+t+2} \mid s', a' \right]$$

$$Q^{\pi}(s, a) = \sum_{s'} p(s' \mid s, a)$$

$$\left[R(s, a, s') + \gamma \sum_{a'} \mathop{\mathbb{E}}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k R_{k+t+2} \mid s', a' \right] \right]$$

$$Q^{\pi}(s, a) = \sum_{s'} p(s' \mid s, a)$$

$$\left[R(s, a, s') + \gamma \sum_{a'} \pi(a' \mid s') Q^{\pi}(s', a') \right]$$

$$Q_{t+1}^{\pi *}(s, a) = \sum_{s'} p(s' \mid s, a)$$

$$\left[R(s, a, s') + \gamma \max_{a'} Q_t^*(s', a') \right]$$

Bellman Equations

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[R_{t+1}(s,a,s') + \gamma G_{t+1} \mid s,a \right]$$

$$V_k^{\pi}(s) = \sum_a \pi(a|s) \sum_{s'} p(s'|a,s) \left[R(s,a,s') + \gamma V_{k-1}^{\pi}(s') \right]$$

↳ $\mathbb{E}_{\pi}[R_{t+1}(s,a,s')] = \sum_a \pi(a|s) \sum_{s'} p(s'|a,s) R(s,a,s')$

value iteration ⭐⭐

we can also have:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[R_{t+1}(s,a,s') + \gamma G_{t+1} \mid s,a \right]$$

$$V^{\pi}(s) = R_{t+1}(s,a,s') + \gamma \sum_{s'} p(s'|a,s) V^{\pi}(s)$$

$$V^{\pi}(s) = \left(I - \gamma T^{\pi} \right)^{-1} R_{t+1}(s,a,s')$$

$$V^{\pi}(s) = \left(I - \gamma T^{\pi} \right)^{-1} R^{\pi}$$

$1 \times d$ ↓ $d \rightarrow$ no. of states

$d \times d$ Transition Matrix

$1 \times d$ ↳ Linear system of equations

Value Iteration Algorithm

1. Initialize $V_0^{\pi}(s) = 0 \quad \forall s \in S$
2. For $t = 1 \dots H$ (one episode)
   For all $s \in S$
   For all a possible at state s

update value $V_t^{\pi}(s) \leftarrow \max_a \sum_{s'} p(s'|s,a) \left[R(s,a,s') + \gamma V_{t-1}^{\pi}(s') \right]$

update policy $\pi_t(s) \leftarrow$ that max action a

Can be solved using Dynamic Programming

note → for each time step, update values of all states, update policy. Repeat time steps till convergence.

v-value iteration

$$V_t^{\pi}(s) = \max_a \sum_{s'} p(s'|a,s) \left[R(s,a,s') + \gamma V_{t-1}^{\pi}(s) \right]$$

q-value iteration

$$Q_t^{\pi}(s,a) = \sum_{s'} p(s'|a,s) \left[R(s,a,s') + \gamma \max_{a'} Q_{t-1}^{\pi}(s',a') \right]$$

Policy Iteration

1. we start with an initial policy π_t
2. we find values until convergence.

value update $V_i^{\pi_t}(s) = \max_a \sum_{s'} p(s'|a,s) \left[R(s,a,s') + \gamma V_{i-1}^{\pi_t}(s') \right]$

↑ loop over $s \in S$, $a \in A$

3. update policy based on $V_k^{\pi_k}(s)$ using one step look ahead.

one policy update $\pi_{t+1}(s) \leftarrow \arg\max_a \sum_{s'} p(s'|s,a) \left[R(s,a,s') + \gamma V^{\pi_t}(s') \right]$

4. Repeat until policy converges ↓ converged.

* Dynamic Programming

1. If we know $p(s'|s,a)$, the transition function and $R(s,a,s')$, the reward system, we can use Dynamic Programming to compute $\pi^*(s)$ (optimal policy)

Policy and Value iterations are iterative hence, intermidiate outputs can be saved to reduce computational complexity.

Reinforcement Learning

Model Free Approach

Goal → Find optimal policy
By finding optimal value functions
$V^{\pi^*}(s)$, $Q^{\pi^*}(s,a)$

Condition → $p(s'|s,a)$ Transition Function
$R(s,a,s')$ Reward → UNKOWN

Learn by interacting with Environment.

2 Approaches.
1. Monte Carlo Approach
2. Temporal Difference Approach.

Monte Carlo Approach

Few concepts

1. Incremental mean.

$$\mu_k = \frac{1}{k}\sum_{j=1}^{k} x_j \quad \text{mean of } j \text{ terms}$$

$$= \frac{1}{k}\left(x_k + \sum_{j=1}^{k-1} x_j\right)$$

$$= \frac{1}{k}\left(x_k + (k-1)\mu_{k-1}\right)$$

$$= \frac{x_k}{k} + \mu_{k-1} - \frac{1}{k}\mu_{k-1}$$

$$\mu_k = \mu_{k-1} + \frac{1}{k}(x_k - \mu_{k-1})$$

new Mean = old mean + $\frac{1}{\text{Total no. of terms now}}$ (new value − old mean)

$$\mu_k = f(\mu_{k-1}, k, x_k)$$

$$V^{\pi}_{t+1}(s) = V^{\pi}_{t}(s) + \frac{1}{N(s_{t+1})+1}\left(G_t - V^{\pi}_{t}(s)\right)$$

G_t → total reward at step t

Monte carlo Approach

* $V^{\pi}(s) = \underset{\pi}{\mathbb{E}}[G_t \mid S_t = s]$

we learn $V^{\pi}(s)$ by taking average over G_t

* Evaluted over one each episode.

Algorithm Random initialize $V(s)$ $\forall s \in S$

1. loop forever (for each episode)
2. Generate an episode (E) from S_0 — to — S_T

   $G_t = 0$
3. For each t from ~~$T-1 \to 0$~~ $T-1 \to 0$

   $G_t = \gamma G_t + R_{t+1}$

   for S in E (state visited in current episode)

   update $N(S_t) \leftarrow N(S_t) + 1$

   $V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$

we can get reward for first occurence or every occurence of state S in E

or

$$V(S_t) \leftarrow V(S_t) + \alpha (G_t - V(S_t))$$

↓ For non-stationary problems.

↓ remember rate Learning Rate

Key Points

* Value is estimated as mean over G_t
* we loop over those S which are in E, and not over all possible S as in Dynamic Programming
* Above is value iteration, we can also do policy iteration or control tasks.
* For Q-value

  $N(S_t, a_t) \leftarrow N(S_t, a_t) + 1$

  $Q(S_t, a_t) \leftarrow Q(S_t, a_t) + \frac{1}{N(S_t, a_t)}(G_t - Q(S_t, a_t))$
* First generate E and then evaluate.

Temporal Difference Approach (T.D)

mc + DP ——— Dynamic Programming

(mc = monte carlo)

* Don't wait till end of episode (E) to update the expected future reward estimation (V)
* wait only till next step.

In mc

$$V^{\pi}(s) = \mathbb{E}_{\pi}[G_t \mid S_t = s]$$

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

G_t — Actual Return — computed at end of episode.

In TD & DP

$$V^{\pi}(s) = \mathbb{E}_{\pi}[G_t \mid S_t = s]$$

$$= \mathbb{E}_{\pi}[R_{t+1} + \gamma G_{t+1} \mid S_t = s]$$

$$= \mathbb{E}_{\pi}[R_{t+1} + \gamma V^{\pi}(S_{t+1}) \mid S_t = s]$$

So,

$$V^{\pi}(S_t) \leftarrow V^{\pi}(S_t) + \alpha\left(R_{t+1} + \gamma V^{\pi}(S_{t+1}) - V^{\pi}(S_t)\right)$$

- $V^{\pi}(S_t)$ ← Simplest TD(0) ↓ look ahead one-step
- α ← learning rate
- $R_{t+1} + \gamma V^{\pi}(S_{t+1})$ ← An estimate of return (in mc it was $G_t \rightarrow$ Actual Return)

$$Q^{\pi}(S_t, a_t) \leftarrow Q^{\pi}(S_t, a_t) + \alpha\left(R_{t+1} + \gamma \max_{a \in A} Q^{\pi}(S_{t+1}, a) - Q^{\pi}(S_t, a_t)\right)$$

we can do more steps look ahead

example ↓

$$\alpha\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 \max_{a \in A} Q(S_{t+3}, a) - Q^{\pi}(S_t, a_t)\right]$$

Algorithm (On-Policy Method) SARSA

1. Random initialize $Q(s,a)$ $\forall s \in S$, $a \in A$

   $Q(S_T, \cdot) = 0$

   ↳ terminal state

2. Loop forever (for each episode)

3. for each step in episode

4. choose ~~Q~~ from S in E using policy derived from Q

   Take action 'a' observe $R(s,a,s')$

$$Q(s,a) \leftarrow Q(s,a) + \alpha [R(s,a,s') + \gamma \text{ ~~max~~ } Q(s',a') - Q(s,a)]$$

~~a~~

↓

choose action based on ~~epsilon~~ same ~~greedy method~~ policy. current $Q(s,a)$

$s \leftarrow s'$

Q-learning (Off-Policy method)

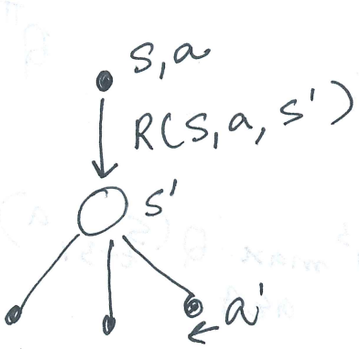
Agent may not follow the greedy policy.

Agent may not choose $\max_{a'} Q(s',a')$ as next action (Exploitation)

↳ Agent can choose random action (Exploration)

$$Q(s,a) \leftarrow Q(s,a) + \alpha [R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

choose action based on epsilon greedy method policy



Reinforcement Learning

Don't know $p(s'|a,s)$ (Transition Matrix) and/or $R(s,a,s')$ Rewards.

Q-Learning

Here, we learn Q-Table

Rows → state (s)

Columns → action (a)

$$Q^{\pi}(s_t, a_t) = \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \,\middle|\, s_t, a_t\right]$$

- $Q^{\pi}(s_t, a_t)$: Q-value for a given state & action pair
- $R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$: expected discounted cumulative award
- s_t, a_t: given current state and action

* we don't know policy (π)

In Q-Learning, we explore the environment to learn (update) Q-Table

Update Q-Table

$$Q(s,a) = Q(s,a) + \alpha\left[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$$

- $Q(s,a)$: new Q value
- $Q(s,a)$: current Q-value
- α: learning rate
- $R(s,a,s')$: reward for taking current action
- γ: discount factor
- $\max_{a'} Q(s',a')$: Highest Q value we can achieve in state s'

Deep Q-learning

$$Q(s,a) = (1-\alpha) Q(s,a) + \alpha\left[R(s,a,s') + \gamma \max_{a'} Q(s',a')\right]$$

$$\text{Target}^{T(s')} = R(s,a,s') + \gamma \max_{a'} Q(s',a')$$

$$Q(s,a) = (1-\alpha) Q(s,a) + \alpha \, \text{Target}$$

Q-Learning Algo

1. Initialize Q-Table $Q(s,a) = 0 \quad \forall s \in S \quad a \in A$
2. Sample an action a → How to sample action??
3. Perform action a
4. update $Q(s,a)$
5. Reached state s', repeat 2

How to Sample action -??

1. choose $\arg\max_a Q(s,a)$ → Greedy Exploitation
2. choose randomly → Exploration

Approximate Q-Learning

what if $Q(s,a)$ is a weighted sum of various features computed over s,a

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \cdots + w_n f_n(s,$$

$$Q(s,a) = \sum_{i=1}^{N} w_i f_i(s,a)$$

approximation of $Q(s,a)$

MSE error $(w) = \left(Q(s,a) - \sum_i w_i f_i(s,a) \right)^2$

$$\frac{\partial E(w)}{\partial w_m} = -\left(Q(s,a) - \sum_i w_i f_i(s,a) \right) f_m(s,a)$$

update weight

$$w_m = w_m + \alpha \left(Q(s,a) - \sum_i w_i f_i(s,a) \right) f_m(s,a)$$

0