Policy Evaluation and Value function Approximation

Wen Yu
Departamento de Control Automático
CINVESTAV-IPN

On-policy/Off-policy

ow to learn good policy

- On-policy: evaluate a policy from data obtained from **that** policy. Sampling policy is the same as learning policy. (s_1, a_1, s_2, a_2)
- Off-policy: evaluate a policy from data obtained from a **different** policy. Sampling policy is different with learning policy. (s_1, a_1, s_2, a_1), (s_1, a_2, s_2, a_2)

On-policy/Off-policy

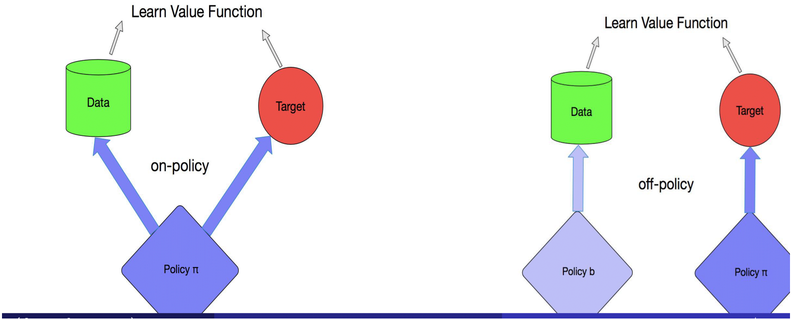
On-policy:

- It learns a better policy using the data from this policy. It is local optimal.
- It cannot assure both exploration and exploitation.

Off-policy approach used two policies:

- One policy generates behavior (exploration, go anywhere). It is called behavior policy, b
- Another policy is for exploitation (use optimal policy). It is called target policy, π

On-policy/Off-policy



Generalized Policy Interaction

Initialize policy π

Repeat

  Polity evaluation: compute Q^{π}

  Policy improvement: update π

$$\pi'(s) = \arg\max_a Q^{\pi}(s,a)$$
$$= \arg\max_a \left[r(s,a) + \gamma \sum_{s' \in S} p(s' \mid s,a) V^{\pi}(s')\right]$$

It is model-based, we need $p(s' \mid s, a)$

MC for Q evaluation (On Policy)

MC Policy Evaluation (Model free)

Initialize the counter $N\left(s,a\right)=0$, $G\left(s,a\right)=0$, $Q^{\pi}\left(s,a\right)=0$, $\forall s\in S$, $\forall a\in A$

Loop

Using policy π sample the episode i untill T_i, generate $s_{i,1}, a_{i,1}, r_{i,1}, \cdots s_{i,T_i}, a_{i,T_i}, r_{i,T_i},$

calculate the return from t to all path of the ith episode

$$G_{i,t}=r_{i,t}+\gamma r_{i,t+1}+\cdots+\gamma^{T_i-1}r_{i,T_i}$$

For each state-action (s,a) visited in episode i

For the **first** time t that the state (s,a) is visited in episode i

$$\begin{aligned} N\left(s,a\right) &= N\left(s,a\right)+1\\ G\left(s,a\right) &= G\left(s,a\right)+G_{i,t}\left(s,a\right)\\ Q^{\pi}\left(s,a\right) &= \frac{G\left(s,a\right)}{N\left(s,a\right)} \end{aligned}$$

Policy Improvement

Given an estimate $Q^{\pi}(s,a), \forall s, \forall a$
Update new policy

$$\pi_{t+1}(s) = \arg\max_{a} Q^{\pi_t}(s,a)$$

$$\begin{aligned} &\arg\max_a Q^{\pi_i}(s,a) \longrightarrow a \\ &\max_a Q^{\pi_i}(s,a) \longrightarrow Q \end{aligned}$$

Policy Evaluation with Exploration

- Need to try all (s, a) pairs, then follow π
- $Q^{\pi}(s, a)$ should be good enough so that policy improvement is a monotonic operator

Balance exploration and exploitation. Define $|\mathcal{A}|$ be the number of actions. ϵ-greedy policy is

$$\pi(a \mid s) = \begin{cases} \arg\max_a Q^{\pi}(s, a) & \text{probability } (1-\epsilon) \\ a & \text{probability } \frac{\epsilon}{|\mathcal{A}|} \end{cases}$$

where a is a random action with probability ϵ, but we take a $|\mathcal{A}|$ times, $\max_a Q^{\pi}(s, a)$ is greedy with probability $(1-\epsilon)$
It is also called $\epsilon-$soft, or soft policy

Convergence the soft-policy

Theorem

For any ϵ-greedy policy π, w.r.t. Q^{π}, π_{i+1} is a monotonic improvement

$$V^{\pi_{i+1}} \geq V^{\pi_i}$$

Proof.

Becasue

$$\begin{aligned} Q^{\pi_k}\left(s,a\right) = r\left(s,a\right) + \gamma \textstyle\sum_{s' \in S} p\left(s' \mid s,a\right) V^{\pi_k}\left(s'\right) \\ Q^{\pi}\left(s,a\right) = r\left(s,a\right) + \gamma \textstyle\sum_{s' \in S} \pi\left(s' \mid s,a\right) Q^{\pi}\left(s'\right) \end{aligned}$$

so

$$\begin{aligned} & V^{\pi_{i+1}} = V^{\pi}\left(s,\pi_{i+1}\right) = \mathbb{E}\left[Q^{\pi_{i+1}}\left(s,a\right)\right] = \textstyle\sum_{a \in A} \pi_{i+1}\left(a \mid s\right) Q^{\pi}\left(s,a\right) \\ & = \left(1-\epsilon\right) \max_a Q^{\pi}\left(s,a\right) + \frac{\epsilon}{|\mathcal{A}|} \textstyle\sum_{a \in A} Q^{\pi}\left(s,a\right) \end{aligned}$$

□

Convergence the soft-policy

We calculate

$$
\begin{aligned}
&(1-\epsilon)\max_a Q^{\pi}(s,a) = (1-\epsilon)\max_a Q^{\pi}(s,a)\frac{1-\epsilon}{1-\epsilon}\\
&= (1-\epsilon)\max_a Q^{\pi}(s,a)\frac{\left(\sum_a \pi_i(a|s)\right)-\epsilon}{1-\epsilon}\\
&\geq (1-\epsilon)\, Q^{\pi}(s,a)\frac{\sum_a\left[\pi_i(a|s)-\frac{1}{|\mathcal{A}|}\epsilon\right]}{1-\epsilon}\\
&= Q^{\pi}(s,a)\sum_a\left[\pi_i(a \mid s)-\frac{1}{|\mathcal{A}|}\epsilon\right]\\
&= \sum_a \pi_i(a \mid s)\, Q^{\pi}(s,a)-\frac{\epsilon}{|\mathcal{A}|}\sum_a Q^{\pi}(s,a)
\end{aligned}
$$

Convergence the soft-policy

So

$$V^{\pi}(s,\pi_{i+1}) \geq \sum_a \pi_i(a \mid s) Q^{\pi}(s,a) - \frac{\epsilon}{|\mathcal{A}|}\sum_a Q^{\pi}(s,a) + \frac{\epsilon}{|\mathcal{A}|}\sum_{a\in A} Q^{\pi}(s,$$
$$= \sum_a \pi_i(a \mid s) Q^{\pi}(s,a) = \mathbb{E}\left[Q^{\pi}(s,a)\right] = V^{\pi_i}$$

So the policy improvement

$$\pi(a \mid s) = \begin{cases} \arg\max_a Q^{\pi}(s,a) & \text{probability } (1-\epsilon) \\ a & \text{probability } \frac{\epsilon}{|\mathcal{A}|} \end{cases}$$

can assure

$$V^{\pi_{i+1}} \geq V^{\pi_i}, \text{ for all } s \in \mathcal{S}$$

$\varepsilon-$greedy policy improvement over policy π, for any $s \in \mathcal{S}$

Convergence the soft-policy

When the ε-soft policy π is no longer improved, then

$$V^{\pi_{i+1}} = V^{\pi_i}$$

Proof.

Wehn the policy π is no longer improved,

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

so

$$V^{\pi}(s, \pi_{i+1}) = \frac{\epsilon}{|\mathcal{A}|} \textstyle\sum_{a \in A} Q^{\pi}(s,a) + (1-\epsilon) \max_a Q^{\pi}(s,a)$$
$$= \textstyle\sum_a \pi_i (a \mid s) Q^{\pi}(s,a) = V^{\pi_i} = V^{\pi *}$$

$\square$

GLIE (Greedy in Limit of Infinite Exploration)

If all state-action pairs are visited an infinite number of times

$$\lim_{i \to \infty} N_i(s, a) \longrightarrow \infty$$

This means $\epsilon = 0$, the ϵ-greedy policy becomes

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

GLIE

$$\lim_{i \to \infty} \pi_i(a \mid s) \longrightarrow \arg \max_{\text{with prob } 1} Q(s, a)$$

So the behavior policy $\max Q(s, a)$ converges to greedy policy $\pi_i(a \mid s)$.

MC for Q function evaluation with e-greedy

Initialize the counter $N(s,a)=0,\ Q(s,a)=0,\ \forall s\in S,\ \forall a\in A$, set $\epsilon=1,\ i=1$
Create initial ϵ-greedy policy

$$\pi_i = \epsilon\text{-greedy}(Q)$$

Loop

sample the episode i with $\pi_i : s_{i,1}, a_{i,1}, r_{i,1}, \cdots s_{i,T_i}, a_{i,T_i}, r_{i,T_i}$
calculate the return from t to all path of the ith episode

$$G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \cdots + \gamma^{T_i-1} r_{i,T_i}$$

For $t = 1 \cdots T$

if first-visit (s,a) in episode i ,

$$\begin{aligned} &N(s,a) = N(s,a)+1 \\ &Q(s_t,a_t) = \frac{G(s,a)}{N(s,a)} \\ &= Q(s_t,a_t) + \frac{1}{N(s,a)}\left[G_{i,t} - Q(s_t,a_t)\right] \end{aligned}$$

$i = i+1$

TD for policy iteration (SARSA)

Initial ϵ-greedy policy π randomly ($\epsilon = 1$), initial state $s_t = s_0$
Sample action from policy, Take $a_t \sim \pi\left(s_t\right)$
Observer $\left(r_t, s_{t+1}\right)$
Loop

Take action $a_{t+1} \sim \pi\left(s_{t+1}\right)$
Observer $\left(r_{t+1}, s_{t+2}\right)$
Update Q with $\left(s_t, a_t, r_t, s_{t+1}, a_{t+1}\right)$

$$Q\left(s_t, a_t\right) = Q\left(s_t, a_t\right) + \alpha\left[r_t + \gamma Q\left(s_{t+1}, a_{t+1}\right) - Q\left(s_t, a_t\right)\right]$$

Perform policy improvement, ϵ-greedy(Q)
$t = t + 1$

Convergence properties of SARSA

Theorem

SARSA for finite-state and finite-action MDPs converges to the optimal action-value,

$$Q(s,a) \rightarrow Q^*(s,a)$$

if 1) The policy sequence $\pi_t(a \mid s)$ satisfied the condition GLIE (All state-action pairs are visited an infinite number of times), 2) The step α_t satisfied the Robbins-Muno sequence

$$\sum_{t=1}^{\infty} a_t = \infty$$
$$\sum_{t=1}^{\infty} a_t^2 < \infty$$

Empirically do not use it.

Robbins-Muno sequence

Check the condition

$$0 < a_t \leq 1, \sum_t a_t = \infty.$$

We usually select constant learning rate

$$a_t = \alpha, \quad 0 < \alpha \leq 1$$

When

$$a_t = \frac{\eta}{1 + \frac{1}{\beta}k}, \quad 0 < \eta \leq 1, \quad \beta >> 1$$

where η is a constant. Since β is very big, for finite time k, the learning rate $\alpha_k \approx \eta$, is constant.

Robbins-Muno sequence

Because

$$\sum_{k=1}^{\infty} \frac{\eta}{1+\frac{1}{\beta}k} = \infty$$

$$\sum_{k=1}^{\infty} \left(\frac{\eta}{1+\frac{1}{\beta}k}\right)^2 = \eta^2\beta^2\psi\left(\beta,1\right) - \eta^2 < \infty$$

where $\psi\left(\beta,1\right)$ is the Digamma function, it is bounded.
So Q_k converges to zero w.p.1, and hence, Q_k converges to Q^* with probability one.

Q-learning

Maintain state-action estimates and use bootstrapping, use the value of the best function action

$$Q\left(s_t, a_t\right) = Q\left(s_t, a_t\right) + \alpha\left[r_t + \gamma \max_{a'} Q\left(s_{t+1}, a'\right) - Q\left(s_t, a_t\right)\right]$$

where SARSA is

$$Q\left(s_t, a_t\right) = Q\left(s_t, a_t\right) + \alpha\left[r_t + \gamma Q\left(s_{t+1}, a_{t+1}\right) - Q\left(s_t, a_t\right)\right]$$

-greedy exploration

Initial $Q(s,a)$, $\forall s \in S$, initial state $s_t = s_0$
Set π_b to be ϵ-greedy w.r.t. Q
Loop

- Take action $a_t \sim \pi_b(s_t)$
- Observer (r_t, s_{t+1})
- Update Q with (s_t, a_t, r_t, s_{t+1})

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

- Perform policy improvement, ϵ-greedy(Q)
- $t = t + 1$

Double Q-learning

Initial $Q_1(s,a)$ and $Q_2(s,a)$, $\forall s \in S$, initial state $s_t = s_0$

Loop

Set a_t using ϵ-greedy

$$\pi(s) = \arg\max_a (Q_1 + Q_2)$$

Observer (r_t, s_{t+1})

If with c probability

$$Q_1(s_t, a_t) = Q_1(s_t, a_t) + \alpha \left[r_t + \gamma \max_a Q_2(s_{t+1}, a) - Q_1(s_t, a_t) \right]$$

e else

$$Q_2(s_t, a_t) = Q_2(s_t, a_t) + \alpha \left[r_t + \gamma \max_a Q_1(s_{t+1}, a) - Q_2(s_t, a_t) \right]$$

$t = t + 1$

Convergence of Q-learning

Lemma

Consider the stochastic process (ζ, Δ, F), *let* P *be a sequence of increasing,* ζ_0 *and* Δ_0 *are* P_0*-measurable, and* ζ_k, Δ_k *and* F_k *are* P_k*-measurable, if* Δ_k *satisfies*

$$\Delta_{k+1} = (1 - \zeta_k)\,\Delta_k + \zeta_k F_k \tag{1}$$

and

1. $0 < \zeta_k \leq 1$, $\sum_k \zeta_k = \infty$, $\sum_k \zeta_k^2 < \infty$
2. $\|E\{F_k|P_k\}\| \leq \kappa\|\Delta_k\| + c_k$, $\kappa \in (0, 1]$, *and* c_k *converges to zero*
3. $var\{F_k|P_k\} \leq K(1 + \kappa\|\Delta_k\|)^2$, K *is a positive constant,* $\|\cdot\|$ *denotes the maximum norm.*

then Δ_k *converges to zero with probability one.*

Convergence of Q-learning

Theorem

For the finite MDP (X, U, f, ρ), *the Q-learning algorithm*

$$Q_{k+1}(x_k, u_k) = Q_k(x_k, u_k)$$
$$+\alpha_k \left\{\rho(x_k, u_k) + \gamma \min_{u^*} \left[Q_k(x_{k+1}, u^*)\right] - Q_k(x_k, u_k)\right\}$$

converges to the optimal value function Q^* *almost surely, if*

$$\sum_k \alpha_k = \infty, \quad \sum_k \alpha_k^2 < \infty \tag{2}$$

Convergence of Q-learning

The Q-learning algorithm is

$$\begin{aligned} Q_{k+1}(x_k, u_k) &= Q_k(x_k, u_k) + \alpha_k \left[r_{k+1} + \gamma \min_{u^*} Q_k(x_{k+1}, u^*) - Q_k(x_k, u_k)\right] \\ &= (1-\alpha_k) Q_k(x_k, u_k) + \alpha_k \left[r_{k+1} + \gamma \min_{u^*} Q_k(x_{k+1}, u^*)\right] \end{aligned} \tag{3}$$

where $r_{k+1} = \rho(x_k, u_k)$.Define $\Delta_k(x_k, u_k) = Q_k(x_k, u_k) - Q^*$,

$$\Delta_{k+1}(x_k, u_k) = (1-\alpha_k)\Delta_k(x_k, u_k) + \alpha_k \left(r_{k+1} + \gamma \min_{u^*} Q_k\left(x_{k+1}, u^*\right) - Q^*\right)$$

Convergence of Q-learning

Define

$$F_k(x_k, u_k) = r_{k+1} + \gamma \min_{u^*} Q_k\left(x_{k+1}, u^*\right) - Q^*$$

Check the condition

$$\|E\{F_k|P_k\}\| \leq \kappa\|\Delta_k\| + c_k, \kappa \in (0,1]$$

Use the value iteration mapping $\mathcal{H}$ for P_k,

$$E\{F_k(x_k, u_k)|P_k\} = \mathcal{H}\left[Q_k(x_k, u_k)\right] - Q^* = \mathcal{H}\left[Q_k(x_k, u_k)\right] - \mathcal{H}(Q^*)$$

Since $\mathcal{H}$ is a contraction, from Lemma

$$\mathcal{H}\left[Q_k(x_k, u_k)\right] - \mathcal{H}(Q^*) \leq \gamma\|Q_k(x_k, u_k) - Q^*(x_k, u_k)\| = \gamma\|\Delta_k(x_k, u_k)\|$$

So

$$\|E\{F_k(x_k, u_k)|P_k\}\| \leq \gamma\|\Delta_k(x_k, u_k)\|$$

Check the condition

$$var\{F_k|P_k\} \leq K(1+\kappa\|\Delta_k\|)^2$$

$$var\{F_k(x_k,u_k)|P_k\} = var\left[\left(r_{k+1}+\gamma\min_{u^*}Q_k(x_{k+1},u^*)-Q^*\right)^2\right]$$

Because r_{k+1} is bounded, and $\Delta_k(x_k,u_k) = Q_k(x_k,u_k) - Q^*$,

$$var\{F_k(x_k,u_k)|P_k\} \leq K\left(1+\gamma\|\Delta_k(x_k,u_k)\|\right)^2$$

where K is a constant.

Convergence of Q-learning

Check the condition

$$0 < \zeta_k \leq 1, \sum_k \zeta_k = \infty.$$

For Q-learning we usually select constant learning rate

$$\zeta_k = \alpha, \quad 0 < \alpha \leq 1$$

But if

$$\alpha_k = \frac{\eta}{1 + \frac{1}{\beta} k}, \quad 0 < \eta \leq 1, \quad \beta >> 1$$

where η is a constant. Since β is very big, for finite time k, the learning rate $\alpha_k \approx \eta$, is constant.

Convergence of Q-learning

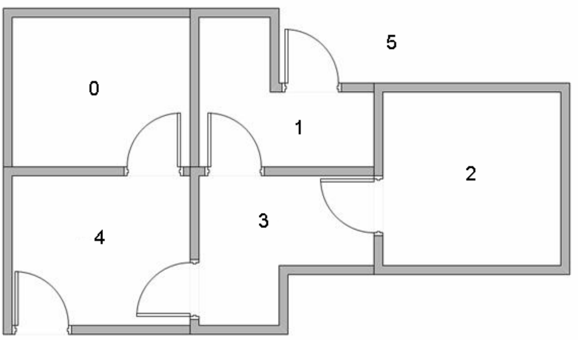
Because

$$\sum_{k=1}^{\infty} \frac{\eta}{1+\frac{1}{\beta}k} = \infty$$

$$\sum_{k=1}^{\infty} \left(\frac{\eta}{1+\frac{1}{\beta}k}\right)^2 = \eta^2\beta^2\psi\left(\beta,1\right) - \eta^2 < \infty$$

where $\psi\left(\beta,1\right)$ is the Digamma function, it is bounded.
So Δ_k converges to zero w.p.1, and hence, Q_k converges to Q^* with probability one.

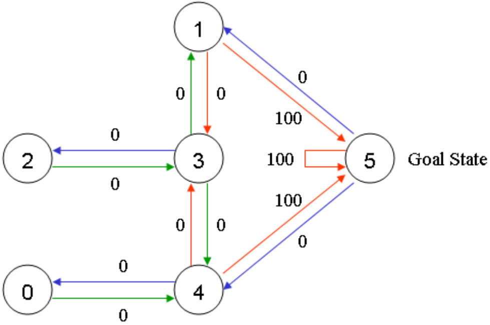
Example



- There are 5 rooms connected by doors.
- The outside of the building is number 5.
- An agent start from any room, and go outside (to Room-5)

Reward R

t



Action

$$R = \begin{array}{c c} & \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} \text{State} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{array}{c} \\ \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{bmatrix} \end{array} \end{array}$$

"-1" represents no link between nodes

Q-learning

Q learning

$$Q_{k+1}\left(s_t, a_t\right) = Q_k\left(s_t, a_t\right) + \alpha\left[r_{t+1} + \gamma \max_{a_t}\left[Q_k\left(s_{t+1}, a_t\right)\right] - Q_k\left(s_t, a_t\right)\right]$$

- Each exploration is an episode
- Each episode consists of the agent moving from the initial state to the goal state

Q-learning

The initial state is Room-1, $s_0 = 1$
The initialize matrix is $Q_1(s_0, a) = 0$
let $\gamma = 0.8$, $\alpha = 1$,

$$\begin{aligned} Q_{k+1}(s_t, a_t) &= Q_k(s_t, a_t) + r_{t+1} + \gamma \max_{a_t}\left[Q_k(s_{t+1}, a_t)\right] - Q_k(s_t, a_t) \\ &= r_{t+1} + \gamma \max_{a_t}\left[Q_k(s_{t+1}, a_t)\right] \end{aligned}$$

Let $Q(1,5)$ is the value of Q-table as $Q_{1,5}$. It is means: Room 1 $(s_t)\longrightarrow$ Room 5 (s_{t+1}).
$Q_k(s_{t+1}, a)$: from Room 5 there are three routes, we use greedy policy $\max_a\{q(5,1), q(5,4), q(5,5)\}$

$$Q(1,5) = r(1,5) + 0.8 \max_a \{Q(5,1), Q(5,4), Q(5,5)\} = 100 + 0.8 * 0 =$$

Q-learning

After one episode

$$Q = \begin{array}{c c} & \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Q-learning

Next episode:
The initial state is Room-3, $s_0 = 3$
There are three possible actions: $3 \longrightarrow 1$, $3 \longrightarrow 2$, $3 \longrightarrow 4$
For $3 \longrightarrow 1$, there are 2 possible actions: $1 \longrightarrow 3$, $1 \longrightarrow 5$.

$$Q(3,1) = r(3,1) + 0.8 \max_a \{Q(1,3), Q(1,5)\} = 0 + 0.8 * 100 = 80$$

$$Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 80 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Then from Room-1, got to Room-5 to finish this episode

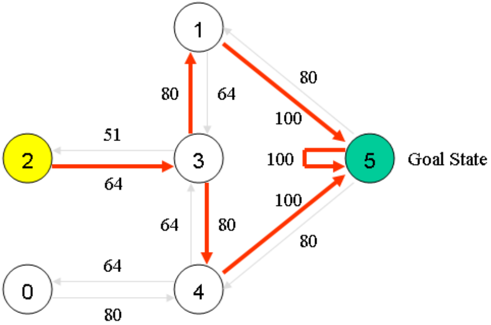
Q-learning

After many episodes, Q-table reaches

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 400 & 0 \\
0 & 0 & 0 & 320 & 0 & 500 \\
0 & 0 & 0 & 320 & 0 & 0 \\
0 & 400 & 256 & 0 & 400 & 0 \\
320 & 0 & 0 & 320 & 0 & 500 \\
0 & 400 & 0 & 0 & 400 & 500
\end{bmatrix}
\end{array}
$$

Control policy

$$\pi\left(s_t\right) = \arg\max_a \left\{Q_t\right\}$$



If the initial is Room-2, the optimal trajectory is $2 \longrightarrow 3 \longrightarrow 1 \longrightarrow 5$.

Value function Approximation (VFA)

Generalization: Tabular representation is insufficient
Update the estimator after each episod (MC) or each step (TD): teacher

Value function Approximation (VFA)

Approximating v_π from experience using a known policy π.
Approximate state value,

$$\hat{V}(s, w) \approx V^{\pi}(s)(s)$$

Approximate control methods: approximation of the action-state value of policy π,

$$\hat{Q}(s, a, w) \approx Q^{\pi}(s)(s, a)$$

There are many ways to approximate the value function. For example, in the simplest linear representation, using $\phi(s)$ to represent the eigenvector of the state s, the state value function can be approximately expressed as:

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

Approximators

1. Linear feature representation
2. Neural Networks
3. Decision trees: highly interpretable
4. Nearest neighbors
5. Fourier/Wavelet bases

Differential function approximators (smooth optimation property)

- Linear feature representation
- Neural Networks

Linear function approximation

We use a feature vetror to represent the feature of the state s,

$$X\left(s\right)=\left[x_1\left(s\right)\cdots x_n\left(s\right)\right]$$

where the features are basis function, which can be defined in many different ways.Value function

$$\hat{V}\left(s,w\right)=\sum_{i=1}^{n}W_iX_i\left(s\right)=X^T\left(s\right)W$$

Linear function approximation

The object function is

$$J_w = \mathbb{E}_\pi \left\{ \left[V^\pi (s) - \hat{V}(s, w) \right]^2 \right\}$$

The real value function $V^\pi (s)$ uses TD (SARSA) or Q-learning.
The stochastic gradient descent is

$$\begin{aligned} \Delta W &= -\alpha \bigtriangledown J(W) = -\alpha \frac{\partial J}{\partial W} \\ &= -2\alpha \left[\hat{V}(s, w) - V^\pi (s) \right] \frac{\partial}{\partial W} \hat{V}(s, w) \\ &= -2\alpha \left[\hat{V}(s, w) - V^\pi (s) \right] X(s) \end{aligned}$$

where α is leraning rate, $\hat{V}(s, w) - V^\pi (s)$ is prediction error, $X(s)$ is feature value

From tabular learning to function approximation

Becasue we do not have real $V^{\pi}(s)$ as the target in supervised learning, we use estimated values from MC or TD to replace $V^{\pi}(s)$
MC: the state value is the expected value of the return, we can substitue $V^{\pi}(s)$ the the return

$$G_t = \sum_i \gamma^{i-1} R_{i,t}$$

after each episod

$$W_{t+1} - W_t = \alpha \left[G_t - X^T(s_t) W_t \right] X(s_t)$$

Bath MC for value function approximation

Let $G\left(s_i\right)$ be an unbiased sample of true expected return $V^{\pi}\left(s_i\right)$

$$J = \arg\min_{W} \sum_{i=1}^{N} \left[G\left(s_i\right) - W_i X_i\left(s\right)\right]$$

Least square

$$W = \left(X^T X\right)^{-1} XG$$

where X is a matrix of the features of each of N states $X_i\left(s\right)$

No any Markov assumption

TD learning

$$V_{t+1}^{\pi}(s) = V_t^{\pi}(s) + \alpha\left[\left\{r_t + \gamma V_t^{\pi}(s_{t+1})\right\} - V_t^{\pi}(s)\right]$$

or

$$V^{\pi}(s) = V^{\pi}(s) + \alpha\left[r + \gamma V^{\pi}(s') - V^{\pi}(s)\right]$$

The traget is

$$r + \gamma V^{\pi}(s')$$

find weights to minimize the mean squared erroe return $V^{\pi}(s_i)$

$$J_w = \mathbb{E}_{\pi}\left\{\left[r + \gamma \hat{V}^{\pi}(s', W) - \hat{V}^{\pi}(s, W)\right]^2\right\}$$

TD learning for value function approximation

Linear TD(0)

$$\Delta W = \alpha \left[r_t + \gamma \hat{V}^{\pi} \left(s', W \right) - \hat{V}^{\pi} \left(s, W \right) \right] X \left(s \right)$$
$$= \alpha \left[r + \gamma X^T \left(s' \right) W - X^T \left(s \right) W \right] X \left(s \right)$$

TD(0) linear value function approximation: Policy Evaluation

Initial $t = 1$, $W = \mathrm{rand}$

Loop

Sample (s_t, a_t, r_t, s_{t+1})

Update weights

$$W = W + \alpha \left[r + \gamma X^T (s') W - X^T (s) W \right] X(s)$$

$t = t + 1$

Action-Value approximation

$$\hat{Q}(s, a, W) \approx Q^{\pi}(s, a)$$

$$J_w = \mathbb{E}_{\pi}\left\{\left[Q^{\pi}(s, a) - \hat{Q}(s, a, W)\right]^2\right\}$$

Use stochastic gradient descent to fid a local minimum

$$\Delta W = -\alpha \frac{\partial J}{\partial W}$$

$$X(s, a) = \left[x_1(s, a) \cdots x_n(s, a)\right]$$

$$\hat{Q}(s, a, W) = X^T(s, a)\, W$$

Action-Value approximation

MC

$$W_{t+1} - W_t = \alpha \left[G_t - X^T \left(s_t, a_t \right) W_t \right] X \left(s_t, a_t \right)$$

SARSA

$$W = W + \alpha \left[r + \gamma X^T \left(s', a' \right) W - X^T \left(s, a \right) W \right] X \left(s, a \right)$$

Q-learning

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

or

$$W = W + \alpha \left[r + \gamma \max \left(X^T \left(s', a' \right) W \right) - X^T \left(s, a \right) W \right] X \left(s, a \right)$$

Deep Reinforcement Learning

Deep Neural Networks to approximate

- Value function
- Policy
- Model

Deep Reinforcement Learning

Neural network

$$\hat{V} = Wz = W\phi\left(s, a\right)$$
$$\hat{Q} = Wz = W\phi\left[V\left(s, a\right) \ldots\right]$$

Stochastic gradient decent (SGD)

$$\Delta W = -\alpha \nabla_W J\left(W\right)$$

MC method

$$\Delta W = \alpha \left[G_t - \hat{Q}\left(s, a, W\right) \right] \nabla_W \hat{Q}\left(s, a, W\right)$$

SARSAR

$$\Delta W = \alpha \left[r + \gamma \hat{Q}\left(s', a', W\right) - \hat{Q}\left(s, a, W\right) \right] \nabla_W \hat{Q}\left(s, a, W\right)$$

Q-leanming

$$\Delta W = \alpha \left[r + \gamma \max_a \hat{Q}\left(s', a, W\right) - \hat{Q}\left(s, a, W\right) \right] \nabla_W \hat{Q}\left(s, a, W\right)$$

DQN

- Deep Q-Network (DQN) that is the first deep reinforcement learning method proposed by DeepMind. Paper was published on Nature in 2015
- NN is CNN
- Naive DQN has 3 convolutional layers and 2 fully connected layers to estimate Q values directly from images.
- DNN is easily overfitting in online reinforcement learning.