



Temporal Difference Methods for Control

Rupam Mahmood

March 2, 2020



Difference between prediction and control in pseudocode

tabular TD(0) for q_π :
$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

target policy determine what this estimate you hope that would converge to.
here V_π : you need the model.
here q_π : you can only have state and action pair.
i.e. the target policy has to be π_θ !!

For prediction. \wedge Which is different than the following

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

From Policy Evaluation to policy improvement.
estimating the optimal policy.

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

Initialize S

Choose A from S using policy derived from Q (e.g., ε -greedy)

Loop for each step of episode:

Take action A , observe R, S'

Choose A' from S' using policy derived from Q (e.g., ε -greedy)

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma Q(S', A') - Q(S, A)]$$

$S \leftarrow S'; A \leftarrow A'$;

until S is terminal

approximating qpi

can also called SARSA (for prediction!!)
⇒ ε -policy π_θ as π_θ .

What would you modify in the above to get the pseudo code for TD(0)?

Tabular TD(0) for estimating v_π ⇒ estimate π is fixed.

Input: the policy π to be evaluated

Algorithm parameter: step size $\alpha \in (0, 1]$

Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\text{terminal}) = 0$

Loop for each episode:

 Initialize S

 Loop for each step of episode:

$A \leftarrow$ action given by π for S

 Take action A , observe R, S'

$V(S) \leftarrow V(S) + \alpha [R + \gamma V(S') - V(S)]$

$S \leftarrow S'$

until S is terminal

Q-learning and on-policy vs. off-policy

SARSA: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$

Q-learning: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_b Q(S_{t+1}, b) - Q(S_t, A_t) \right]$

target: ∵ the target policy is greedy policy.
∴ it is not a fix policy. ∵ it is for control.
∴ greedy policy is deterministic !!

⇒ off policy: b-policy: ε-greedy. ⇒ does it have to be ε-greedy? or it can be other stochastic policy?
t-policy: greedy policy.

⇒ control.
(2 or 2 behavior policies) \Rightarrow both.
With deterministic, you will miss some state-action pairs !!
which means the whole update will not converge to what you want.
∴ b-policy can't be greedy !!

Notice what the target of the update represents and whether the underlying policy of that matches the behavior policy

On-policy constant- α MC:

$$V^{MC}(S_t) \leftarrow V^{MC}(S_t) + \alpha \left[\underbrace{G_t}_{\text{target}} - V^{MC}(S_t) \right]$$

return generated by the behavior policy

Off-policy constant- α MC:

$$V^{MC}(S_t) \leftarrow V^{MC}(S_t) + \alpha \left[\underbrace{\rho_{t:T-1} G_t}_{\text{target}} - V^{MC}(S_t) \right]$$

Difference between Q-learning and SARSA in pseudocode

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Loop for each step of episode:

 Take action A , observe R, S'

 Choose A' from S' using policy derived from Q (e.g., ε -greedy)

$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$

$S \leftarrow S'; A \leftarrow A'$;

 until S is terminal

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Loop for each step of episode:

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Take action A , observe R, S'

$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$

$S \leftarrow S'$

 until S is terminal

$t = 0$

choose action

Take action & observe

choose action

Take action & observe

choose action

Take action & observe

...

Time $t \rightarrow$

Where would you put SARSA updates?

Where would you put Q-learning updates?

What happens if we switch the time of update for both?

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Loop for each step of episode:

 Take action A , observe R, S'

 Choose A' from S' using policy derived from Q (e.g., ε -greedy)

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma Q(S', A') - Q(S, A)]$$

$S \leftarrow S'; A \leftarrow A'$;

 until S is terminal

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Loop for each step of episode:

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Take action A , observe R, S'

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$$

$S \leftarrow S'$

 until S is terminal

Modify the Q-learning Pseudocode minimally to get SARSA

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Loop for each step of episode:

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Take action A , observe R, S'

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$$

$$S \leftarrow S'$$

 until S is terminal

The equivalence of two SARSA algorithms breaks in the real world

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Loop for each step of episode:

 Take action A , observe R, S'

 Choose A' from S' using policy derived from Q (e.g., ε -greedy)

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \underbrace{Q(S', A')}_{\substack{\text{Sarsa: state, action pair for next state and action.} \\ \text{action value for}}} - Q(S, A)]$$

$S \leftarrow S'$; $A \leftarrow A'$;

 until S is terminal

6.6 Expected Sarsa

Consider the learning algorithm that is just like Q-learning except that instead of the maximum over next state-action pairs it uses the expected value, taking into account how likely each action is under the current policy. That is, consider the algorithm with the update rule

$$\begin{aligned}
 Q(S_t, A_t) &\leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \mathbb{E}_\pi [Q(S_{t+1}, A_{t+1}) \mid S_{t+1}] - Q(S_t, A_t) \right] \\
 &\leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \sum_a \pi(a|S_{t+1}) Q(S_{t+1}, a) - Q(S_t, A_t) \right], \tag{6.9}
 \end{aligned}$$

the target.
the target policy
expected Sarsa: bootstrapping from action values from all the next actions possible
drawn from behavior policy π_b .
behavior policy π_b have to specify!!

It could be on-policy or off-policy, and for prediction or control!

fix target policy.

changing target policy. (PT 和 TD 策略进行，最终收敛到最优策略)

(1): π_b - policy is also π_π . Expected Sarsa is on-policy. (2): π_π over $Q(S_t, A_t)$ \neq π_π A_t is drawn from π_π .
 也就是说 π_π 是 π_π 还是 $\pi_{\pi_{\pi}}$ 或者 $\pi_{\pi_{\pi_{\pi}}}$ 等等。>> Specify if expected Sarsa is control or prediction.

Question:

What will be an off-policy TD(0) update for q_π ?

TD Control

Wed



[Notes]

when we say a policy is fixed, we mean its state to action probability ^{map} is stationary, that is, does not change over time. In that sense, an ϵ -greedy policy with action values being learned is not a fixed policy. ^{Its} action values is always changing!!

[Identifying the target policy in TD methods]

The target policy can be identified through the target of the update:

★ New-estimate \doteq previous-estimate

$$+ \text{step size } \underbrace{\alpha \left[\text{target} - \text{previous-estimate} \right]}_{\text{error}}$$

A method described only using an update rule is under-specified. We need to describe the behavior and target policies.

For example, which method is the following?

$$Q(S_t, A_t) \doteq Q(S_t, A_t) + \alpha \left[\underbrace{R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)}_{\text{target}} \right]$$

It is known as SARSA. It is an on-policy method meaning that the target and the behavior policies are the same. But SARSA can be either a prediction or a control method. Which one is it?

Depends on the target policy! How is A_{t+1} drawn? According to fixed or ϵ -greedy policy?

[Temporal difference learning algorithms for action values] (for state values)

	Prediction	Control
on-policy	<ul style="list-style-type: none">- SARSA for prediction Also known as TD(0) for qpi- Expected SARSA- (TD(0))	<ul style="list-style-type: none">- SARSA for control- Expected SARSA
off-policy	<ul style="list-style-type: none">- Expected SARSA- (TD(0))	<ul style="list-style-type: none">- Expected SARSA- Q-learning

- Prediction vs. control is determined based on whether the target policy is fixed or ϵ -greedy.
- On-policy vs. off-policy is determined based on whether the behavior and the target policies are the same or different.

[Questions]

① Give the specification of the on-policy SARSA prediction method.

(Ans)

Update rule:

$$Q(S_t, A_t) \doteq Q(S_t, A_t) + \alpha [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

Target policy is fixed.

Behavior policy is the same as the target policy.

② Give the specification of the on-policy SARSA control method.

(Ans)

Update rule:

$$Q(S_t, A_t) \doteq Q(S_t, A_t) + \alpha [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

Target policy is ϵ -greedy.

Behavior policy is the same as the target policy.

③ Is the following expected SARSA method on-policy or off-policy and for prediction or for control?

Update rule:

$$Q(S_t, A_t) \doteq Q(S_t, A_t) + \alpha [R_{t+1} + \gamma \sum_c \pi(c|S_{t+1}) Q(S_{t+1}, c) - Q(S_t, A_t)]$$

off-policy
Target policy π is fixed.

for prediction since the target policy is fixed and can't have target policy improvement

Behavior policy is different than target policy.

⇒ give the update and describe the policy.

④ Why Q-learning is off-policy? target policy: π ; behavior policy: μ

⑤ Why Q-learning is a control method?

⑥ Why off-policy expected SARSA or Q-learning does not require importance sampling?

(Ans) Expected SARSA in special case becomes Q learning!!

First of all, Q-learning can be subsumed within off-policy [^] expected SARSA in the following way:

$$Q(S_t, A_t) \doteq Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \sum_c \pi(c|S_{t+1}) Q(S_{t+1}, c) - Q(S_t, A_t) \right],$$

greedy based on Q . to the greedy action. Since exploring one max. all probability is assigned to c . behavior policy is ϵ -greedy.

where the target policy π is different than the behavior policy and is greedy.

Then it suffices to argue why ^{off-policy} [^] expected SARSA does not require importance sampling.

In off-policy methods, importance sampling is used to account for the discrepancy of (sampled) action probabilities between the target and the behavior policies. → For expected SARSA, no sampling action is accounted for.

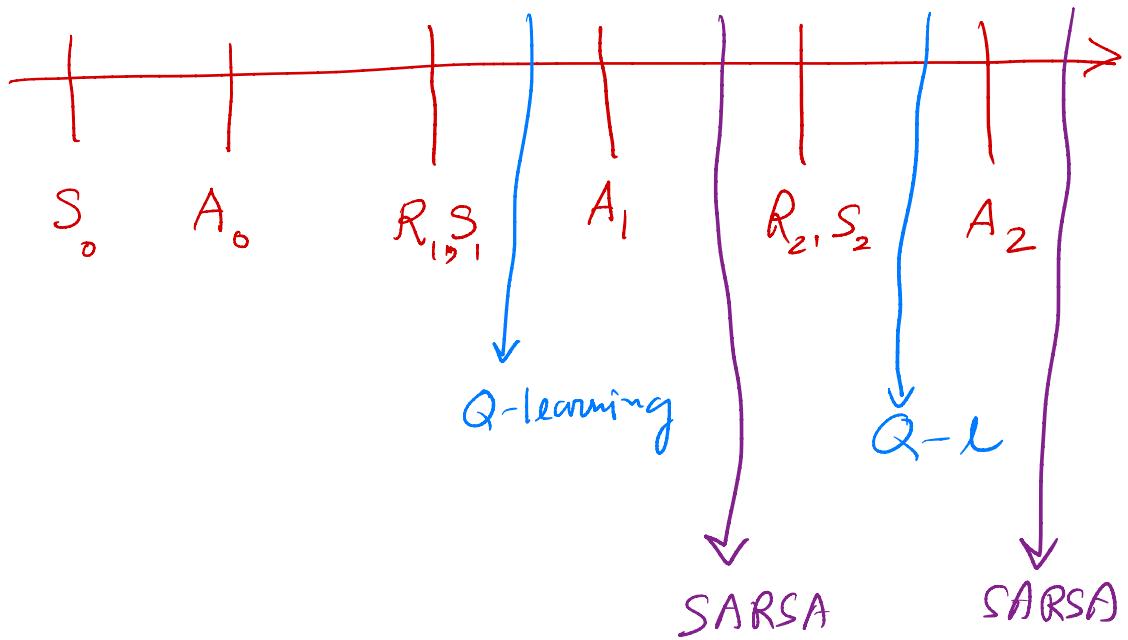
(See next page)

In action value methods, current action
 A_t is given. And the next action in
expected SARSA^{target} is never sampled.

Rather, the expectation is used
directly:

$$\sum_c \pi(c | S_{t+1}) Q(S_{t+1}, c) = E_{\pi} [Q(S_{t+1}, A_{t+1}) | S_{t+1}]$$

So no sampling discrepancy to account for.



正在发言: Rupam Mahmood

Deriving TD methods

$$v_{\pi}(s) \doteq E_{\pi} \left[G_t \right] \underset{\text{target of next}}{\text{S}_t = s}$$

$$= E_{\pi_L} \left[R_{t+1} + \gamma u_{\pi} \left(S_{t+1} \right) \middle| S_t = s \right]$$

Derive V_{th}, T_0 is for the current action. target of T_0

is desirous to fit for the next action.

Then the target for TD(0) for π^* is

is $R_{t+1} + \dots$

$$V(S_t) \leftarrow V(S_t)$$

$$+ \alpha \left[R_{t+1} + \gamma V(S_{t+1}) - V(S_t) \right]$$

target of ID for update

i. For MC_i , we make the same update. 是指 target 为 MC_i 的话, $\text{At}_i: V(S_t) \leftarrow V(S_t) + \alpha [G_t - V(S_t)]$.



正在发言: Rupam Mahmood

How to derive off-policy MC for v_{π}
 where the behavior policy is μ . \rightarrow off policy.

$$v_{\pi} \doteq E_{\pi} [G_t | S_t = s] = E_{\mu} \left[\frac{p_{t:T}}{p_{t:T}} G_t | S_t = s \right]$$

importance sampling ratio for whole trajectory.

the update for off-policy MC.

\therefore The target for the update is $p_{t:T} G_t$.

\therefore update for off-policy MC for v_{π} is

$$V(S_t) \leftarrow V(S_t) + \alpha \left[p_{t:T} G_t - V(S_t) \right]$$

target.

Derive off-policy TD for v_{π}

when $A \sim \pi$: action is drawn from behavior policy μ .

$$v_{\pi} = E_{\mu} \left[\sum_{t=0}^T (R_{t+1} + \gamma v_{\pi}(S_{t+1})) | S_t = s \right]$$

$$p_{t:t} = \frac{\pi(A_t | S_t)}{\mu(A_t | S_t)}$$

update target of TD.

importance sampling ratio for single timestep.

\therefore the update is

$$V(S_t) \leftarrow V(S_t) + \alpha \left[\sum_{t=0}^T (R_{t+1} + \gamma V(S_{t+1})) - V(S_t) \right]$$

where p here is the state-transition probability function defined by (3.4). Thus, the relative probability of the trajectory under the target and behavior policies (the importance-sampling ratio) is

$$\rho_{t:T-1} \doteq \frac{\prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k | S_k) p(S_{k+1} | S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k | S_k)}{b(A_k | S_k)}.$$

single timestep of the importance sampling ratio.

→ multiplying together give importance sampling ratio of the whole trajectory. (5.3)

Derive offpolicy TD(0) for π or π
or equivalently offpolicy SARSA
for prediction, when $A \sim \mu$.

$$\begin{aligned}
 q_{\pi}(s, a) & \stackrel{\text{def}}{=} E_{\pi} [A_t \mid S_t = s, A_t = a] \\
 & = E_{\pi} \left[R_{t+1} + \gamma q_{\pi}(S_{t+1}, A_{t+1}) \mid \begin{array}{l} S_t = s, \\ A_t = a \end{array} \right] \\
 & \quad \text{define } q_{\pi} : ? \text{ for the next action!} \\
 & = E_{\mu} \left[\underbrace{\rho_{t+1:t+1} (R_{t+1} + \gamma q_{\pi}(S_{t+1}, A_{t+1}))}_{\text{target}} \mid \begin{array}{l} S_t = s, \\ A_t = a \end{array} \right] \\
 & \quad \text{random variable here.}
 \end{aligned}$$

∴ update

$$\begin{aligned}
 Q(S_t, A_t) & \leftarrow Q(S_t, A_t) \\
 & + \alpha \left[\underbrace{\rho_{t+1:t+1} (R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})) - Q(S_t, A_t)}_{\text{target}} \right]
 \end{aligned}$$