Reinforcement Learning

Temporal Difference Algorithms

Lecturer: Nicolò Cesa-Bianchi

This material is partially based on the book draft “Reinforcement Learning: Foundations” by Shie Mannor, Yishay Mansour, and Aviv Tamar.

We consider the discounted infinite horizon criterion and focus on MDP with finite state space $S$, finite action space $A$ such that $A(s) = A$ for all $s \in S$, transition kernel $\{p(\cdot \mid s, a) : s \in S, a \in A\}$, and time-independent reward function $r : S \times A \to [-1, 1]$.

Fix a stationary deterministic policy $\pi$ and consider the problem of estimating the state-value function $V^\pi$. If the MDP were known, we could simply use fixed-policy value iteration or linear programming. When the MDP is unknown, we must use samples from the trajectory generated by $\pi$. Recall the system of linear equations that $V^\pi$ satisfies,

$$V^\pi(s) = r(s, \pi(s)) + \gamma \mathbb{E}[V^\pi(s') \mid s] \quad s \in S$$

where $s' \sim p(\cdot \mid s, \pi(s))$. Now, similarly to what we did in $Q$-learning, we can obtain a sequence $V_0, V_1, \ldots$ of approximations to $V^\pi$ by running gradient descent on the square loss

$$\ell_t(x) = \frac{1}{2} \left( x - r(s_t, \pi(s_t)) - \gamma \mathbb{E}[V_t(s') \mid s_t] \right)^2$$

for $x = V_t(s_t)$, which amounts to the update

$$V_{t+1}(s_t) = (1 - \eta_t)V_t(s_t) + \eta_t \left( r(s_t, \pi(s_t)) + \gamma \mathbb{E}[V_t(s') \mid s_t] \right)$$

Since, however, $\mathbb{E}[V_t(s') \mid s]$ is not directly accessible, we run gradient descent on a perturbed gradient,

$$V_{t+1}(s_t) = (1 - \eta_t)V_t(s_t) + \eta_t \left( r(s_t, \pi(s_t)) + \gamma V_t(s_{t+1}) \right)$$

where $s_{t+1} \sim p(\cdot \mid s_t, \pi(s_t))$. We call temporal difference the quantity

$$\Delta_t = r(s_t, \pi(s_t)) + \gamma V_t(s_{t+1}) - V_t(s_t)$$

and write the above update equivalently as

$$V_{t+1}(s_t) = V_t(s_t) + \eta_t \Delta_t$$

The algorithm based on this update is known as TD(0). Similarly to what we did for $Q$-learning, we can prove the convergence of TD(0) when $\eta_t$ is a function $\eta_t : S \to [0, 1]$ of the states defined by

$$\eta_t(s) = \frac{\mathbb{I}\{s = s_t\}}{N_t(s)} \quad \text{where} \quad N_t(s) = \sum_{\tau=0}^t \mathbb{I}\{s_{\tau} = s\}$$

Because we focus on deterministic policies, the learning rate $\eta_t$ can depend only on states rather than on state-action pairs.
Theorem 1 Assume that TD(0) is run with a stationary deterministic policy \( \pi \) inducing an irreducible Markov chain on the underlying MDP. Then
\[
\lim_{t \to \infty} V_t(s) = V^\pi(s) \quad s \in S
\]
with probability 1.

The update of TD(0) is based on a 1-step lookahead \( R_t^{(1)}(s_t) = r(s_t, a_t) + \gamma V_t(s_{t+1}) \) so that \( \Delta_t = R_t^{(1)}(s_t) - V_t(s_t) \). Using the identity
\[
V^\pi(s) = r(s, \pi(s)) + \mathbb{E} \left[ \sum_{\tau=1}^{\infty} \gamma^\tau r(s_\tau, \pi(s_\tau)) \mid s \right] \quad s \in S
\]
where \( s_\tau \sim p(\cdot \mid s_{\tau-1}, \pi(s_{\tau-1})) \) and \( s_0 = s \), TD(0) can be easily generalized to a \( n \)-step lookahead
\[
R_t^{(n)}(s_t) = \sum_{\tau=0}^{n-1} \gamma^\tau r(s_{t+\tau}, \pi(s_{t+\tau})) + \gamma^n V_t(s_{t+n})
\]
The corresponding updates are \( V_{t+1}(s_t) = V_t(s_t) + \eta_t \Delta_t^{(n)} \) where \( \Delta_t^{(n)} = R_t^{(n)}(s_t) - V_t(s_t) \). Note that
\[
\Delta_t^{(n)} = \sum_{\tau=0}^{n-1} \gamma^\tau \Delta_{t+\tau}
\]
Indeed,
\[
\sum_{\tau=0}^{n-1} \gamma^\tau \Delta_{t+\tau} = \sum_{\tau=0}^{n-1} \gamma^\tau \left( r(s_{t+\tau}, \pi(s_{t+\tau})) + \gamma V_t(s_{t+\tau+1}) - V_t(s_{t+\tau}) \right)
\]
\[
= \sum_{\tau=0}^{n-1} \gamma^\tau r(s_{t+\tau}, \pi(s_{t+\tau})) + \sum_{\tau=0}^{n-1} \left( \gamma^{\tau+1} V_t(s_{t+\tau+1}) - \gamma^\tau V_t(s_{t+\tau}) \right)
\]
\[
= \sum_{\tau=0}^{n-1} \gamma^\tau r(s_{t+\tau}, \pi(s_{t+\tau})) + \gamma^n V_t(s_{t+n}) - V_t(s_t)
\]
\[
= R_t^{(n)}(s_t) - V_t(s_t) = \Delta_t^{(n)}
\]
It can be shown that if we run TD(0) with a \( n \)-step lookahead (for any given \( n \geq 1 \)), then \( V_t(s) \) converges to \( V^\pi(s) \) for all \( s \in S \).

In case of deterministic \( T \) (finite horizon), we can choose \( n = T \) and set \( \gamma = 1 \). The resulting algorithm is known as Monte-Carlo sampling (see Algorithm 1 below here).

Note that the estimate \( V_N \) of the state-value function \( V^\pi \) satisfies
\[
V_N(s) = \frac{1}{n(s)} \sum_{i=1}^{N} R_i(s)
\]
where \( n(s) \) is the number of episodes in which the state \( s \) has been visited at least once. For \( \alpha > 0 \), let \( S_\alpha \subseteq S \) the set of states such that the probability that \( \pi \) visits \( s \in S_\alpha \) in any given episode is at least \( \alpha \). Then we have the following result.
Algorithm 1 Monte-Carlo sampling for MDP with deterministic horizon $T$

**Input:** Stationary deterministic policy $\pi$, initial state $s_0 \in \mathcal{S}$, number $N$ of episodes

1: Set $V_0(s) = 0$ and $n(s) = 0$ for all $s \in \mathcal{S}$; set $s_1 = s_0$
2: for $i = 1, \ldots, N$ do
3: Use $\pi$ to generate $(s_1, a_1, r_1), \ldots, (s_T, a_T, r_T)$
4: for $t = T - 1, \ldots, 0$ do
5: if $s_t$ does not appear in $s_0, \ldots, s_{t-1}$ then
6: $R_i(s_t) = r_t + \cdots + r_T$
7: $n(s) \leftarrow n(s) + 1$
8: Update $V_i(s_t) = V_{i-1}(s_t) + R_i(s_t)$
9: end if
10: end for
11: $s_1 = s_T$
12: end for
13: $V_N(s) \leftarrow \frac{V_N(s)}{n(s)}$ for all $s \in \mathcal{S}$

**Output:** $V_N : \mathcal{S} \rightarrow \mathbb{R}$

**Theorem 2** Assume that we execute $N$ episodes using policy $\pi$ and each episode has length at most $T$. Then, with probability at least $1 - \delta$, for any $s \in \mathcal{S}$, we have $|V_N(s) - V^\pi(s)| \leq \varepsilon$ for

$$N \geq \frac{2m}{\alpha} \ln \frac{2|\mathcal{S}|}{\delta} \quad \text{and} \quad m = \frac{T^2}{\varepsilon} \ln \frac{2|\mathcal{S}|}{\delta}$$

In the discounted setting, the choice of $n$ may impact the quality of the policy evaluation process. Instead of choosing a single value for $n$, we may average over all positive integers. A simple way of implementing this idea is through exponential averaging with a parameter $\lambda \in (0, 1)$. This implies that the weight assigned to each parameter $n$ is $(1 - \lambda)^n$. This leads to the TD($\lambda$) algorithm.

Recall $\Delta_t^{(n)} = R_t^{(n)}(s_t) - V_t(s_t)$. The TD($\lambda$) update is defined by

$$V_{t+1}(s_t) = V_t(s_t) + (1 - \lambda)\eta_t \sum_{n=1}^{\infty} \lambda^{n-1} \Delta_t^{(n)}$$

The problem with this approach is that we have to compute an infinite sum to make a single update. Luckily, there is an equivalent formulation that avoids this problem. The trick is to use the notion of eligibility trace

$$e_t(s) = \sum_{k=0}^{t} (\lambda\gamma)^{t-k} \eta_k I\{s = s_k\}$$

Note that $e_t$ can be recursively computed from $e_0(s) = 0$ and $e_t(s) = (\lambda\gamma)e_{t-1}(s) + \eta_t I\{s = s_t\}$ for all $s \in \mathcal{S}$.

Now recall the definition of temporal difference,

$$\Delta_t = r(s_t, \pi(s_t)) + \gamma V_t(s_{t+1}) - V_t(s_t)$$
Algorithm 2 TD(λ)

Input: Stationary deterministic policy π, initial state s_0 ∈ S, parameter λ ∈ (0, 1)
1: Set V_0(s) = 0 and e_0(s) = 0 for all s ∈ S
2: for t = 0, 1, . . . do
3: Get a_t = π(s_t) and observe r(s_t, a_t), s_{t+1} ∼ p(· | s_t, a_t)
4: Compute ∆_t = r(s_t, a_t) + γ V_t(s_{t+1}) − V_t(s_t)
5: for s ∈ S do
6: Compute e_t(s) = (λγ)e_{t-1}(s) + η_t [s = s_t]
7: Update V_{t+1}(s) = V_t(s) + e_t(s)∆_t
8: end for
9: end for

The backward temporal difference is just the standard temporal difference multiplied by the eligibility trace, ∆_t^B(s) = ∆_t e_t(s). The resulting algorithm is described below here. Note that in the backward view all states s get updated at each time step t.

The following result shows that the forward and backward updates

\[ V_{t+1}^F(s_t) = V_t^F(s_t) + (1 − λ)η_t \sum_{n=1}^\infty λ^{n-1} \Delta_t^{(n)} \quad \text{and} \quad V_{t+1}^B(s) = V_t^B(s) + \Delta_t^B(s) \quad s ∈ S \]

converge to the same limit.

**Theorem 3** Assume

\[ \lim_{t→∞} [s_t = s] = ∞ \quad \text{and} \quad \lim_{t→∞} V_t^F(s) = V^π(s) \]

for all s ∈ S with probability 1. Let V_0^F(s) = 0 and V_0^B(s) = 0 for all s ∈ S. Then

\[ \lim_{t→∞} V_t^B(s) = V^π(s) \quad s ∈ S \]

**Proof.** Fix any s ∈ S. Since V_0^F(s) = 0, s_t = s occurs for infinitely many t with probability 1, and V_{t+1}^F(s) = V_t^F(s) when s_t ≠ s, we have that

\[ \lim_{t→∞} V_t^F(s) = \sum_{t=0}^\infty \left( V_{t+1}^F(s) − V_t^F(s) \right) [s_t = s] \]

Likewise, using V_0^B(s) = 0,

\[ \lim_{t→∞} V_t^B(s) = \sum_{t=0}^\infty \left( V_{t+1}^B(s) − V_t^B(s) \right) \]

Therefore, we are left to prove that

\[ \sum_{t=0}^\infty \left( V_{t+1}^F(s) − V_t^F(s) \right) [s_t = s] = \sum_{t=0}^\infty \left( V_{t+1}^B(s) − V_t^B(s) \right) \]
We have the following chain of equalities

$$
\sum_{t=0}^{\infty} \left( V^F_{t+1}(s) - V^F_t(s) \right) \mathbb{I}\{s_t = s\} = (1 - \lambda) \sum_{t=0}^{\infty} \eta_t \mathbb{I}\{s_t = s\} \sum_{n=1}^{\infty} \lambda^{n-1} \Delta_t^{(n)}
$$

$$
= (1 - \lambda) \sum_{t=0}^{\infty} \eta_t \mathbb{I}\{s_t = s\} \sum_{n=1}^{\infty} \lambda^{n-1} \sum_{\tau=0}^{n-1} \gamma^\tau \Delta_{t+\tau}
$$

$$
= (1 - \lambda) \sum_{t=0}^{\infty} \eta_t \mathbb{I}\{s_t = s\} \sum_{n=0}^{\infty} \lambda^n \sum_{\tau=0}^{n} \gamma^\tau \Delta_{t+\tau}
$$

$$
= (1 - \lambda) \sum_{t=0}^{\infty} \eta_t \mathbb{I}\{s_t = s\} \sum_{\tau=0}^{\infty} \sum_{n=\tau}^{\infty} \lambda^{n} \gamma^{n-\tau} \Delta_{t+\tau}
$$

$$
= (1 - \lambda) \sum_{t=0}^{\infty} \eta_t \mathbb{I}\{s_t = s\} \sum_{\tau=0}^{\infty} \sum_{n=\tau}^{\infty} \lambda^{n} \gamma^{n-\tau} \Delta_{k}
$$

$$
= (1 - \lambda) \sum_{t=0}^{\infty} \eta_t \mathbb{I}\{s_t = s\} \sum_{\tau=0}^{\infty} \sum_{n=\tau}^{\infty} \sum_{k=t}^{\infty} \lambda^{n} \gamma^{n-\tau} \Delta_{k}
$$

$$
= (1 - \lambda) \sum_{t=0}^{\infty} \eta_t \mathbb{I}\{s_t = s\} \sum_{\tau=0}^{\infty} \sum_{n=\tau}^{\infty} \sum_{k=t}^{\infty} \lambda^{n} \gamma^{n-\tau} \Delta_{k}
$$

Since we chose $s$ arbitrarily, the proof is concluded. \qed

Recall that the actor-critic approach is a method for performing policy iteration without knowing the MDP.

1. Policy evaluation: Run $\pi_t$ to evaluate $V_t = V^\pi_t$

2. Policy improvement: Perform the update $\pi_t \rightarrow \pi_{t+1}$

While we can use TD($\lambda$) for the policy evaluation step, the policy improvement step is more easily performed using $Q^\pi$ instead of $V^\pi$. Indeed, for a known MDP, the policy improvement step can be written as

$$
\pi_{t+1}(s) \in \arg\max_{a \in A} \left( r(s, a) + \gamma \mathbb{E}[V^\pi_t(s') \mid s] \right) = \arg\max_{a \in A} Q^\pi_t(s, a)
$$

Hence, we replace the TD(0) evaluation step

$$
V_{t+1}(s_t) = (1 - \eta_t)V_t(s_t) + \eta_t \left( r(s_t, \pi_t(s_t)) + \gamma V_t(s_{t+1}) \right)
$$

with a SARSA evaluation step

$$
Q_{t+1}(s_t, a_t) = (1 - \eta_t)Q_t(s_t, a_t) + \eta_t \left( r(s_t, a_t) + \gamma Q_t(s_{t+1}, a_{t+1}) \right)
$$

(1)
where \( a_t \sim \pi_t(\cdot \mid s_t), \) \( s_{t+1} \sim p(\cdot \mid s_t, a_t) \) and \( a_{t+1} \sim \pi_{t+1}(\cdot \mid s_{t+1}) \). The corresponding improvement step would be

\[
\pi_{t+1}(s_t) = \text{argmax}_a Q_t(s_t, a)
\]

however, to ensure convergence, we must use randomize the improvement step much like we did in SARSA. Hence, we let

\[
\pi_{t+1}(s_t) = \begin{cases} 
\text{argmax}_a Q_t(s_t, a) & \text{with probability } 1 - \varepsilon_t(s) \\
\text{a random action} & \text{with probability } \varepsilon_t(s)
\end{cases}
\]

In the temporal difference setting, the SARSA evaluation step (1) can be also called SARSA(0) because we use a 1-step lookahead. In other words, we can rewrite (1) as \( Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \eta_t \Delta_t \) where we redefined \( \Delta_t = r(s_t, a_t) + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \).

\[
R^{(n)}_t(s_t, a_t) = \sum_{\tau=0}^{n-1} \gamma^\tau r(s_{t+\tau}, a_{t+\tau}) + \gamma^n Q_t(s_{t+n}, a_{t+n})
\]

The corresponding updates are

\[
Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \eta_t \Delta^{(n)}_t \\
\Delta^{(n)}_t = R^{(n)}_t(s_t, a_t) - Q_t(s_t, a_t)
\]

We can now define SARSA(\( \lambda \)) using exponential averaging with parameter \( \lambda \),

\[
Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + (1 - \lambda) \sum_{\tau=0}^{n-1} \lambda^\tau \Delta^{(n)}_{t+\tau}
\]

Now, similarly to TD(\( \lambda \)), we can define a backward view using eligibility traces

\[
e_0(s, a) = 0 \quad \text{and} \quad e_t(s) = (\lambda \gamma) e_{t-1}(s, a) + \eta_t \mathbb{I}\{s = s_t, a = a_t\} \quad \text{for all } (s, a) \in \mathcal{S} \times \mathcal{A}
\]

The resulting algorithm is described below here. Note that in the backward view all state and action pairs \((s, a)\) get updated at each time step \(t\).

---

**Algorithm 3** SARSA(\( \lambda \))

**Input:** Initial random policy \( \pi_0 \), initial state \( s_0 \in \mathcal{S} \), parameter \( \lambda \in (0, 1) \)

1. Set \( Q_0(s, a) = 0 \) and \( e_0(s, a) = 0 \) for all \((s, a) \in \mathcal{S} \times \mathcal{A}\)
2. Draw \( a_0 \sim \pi_0(\cdot \mid s_0) \)
3. for \( t = 0, 1, \ldots\) do
   4. Play \( a_t \) and observe \( r(s_t, a_t), s_{t+1} \sim p(\cdot \mid s_t, a_t) \)
   5. Draw \( a_{t+1} \sim \pi_{t+1}(\cdot \mid s_{t+1}, Q_t) \)
   6. Compute \( \Delta_t = r(s_t, a_t) + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \)
   7. for \((s, a) \in \mathcal{S} \times \mathcal{A}\) do
      8. Compute \( e_t(s, a) = (\lambda \gamma) e_{t-1}(s, a) + \eta_t \mathbb{I}\{s = s_t, a = a_t\} \)
      9. Update \( Q_{t+1}(s, a) = Q_t(s, a) + e_t(s, a) \Delta_t \)
   10. end for
5. end for

---