Reinforcement Learning

Marshal Sinaga - marshal.sinaga@aalto.fi

2024-09-17

This note aims to cover some materials on the reinforcement learning. The primary references are Reinforcement Learning: An Introduction (2nd edition) by Sutton & Barto and ELEC-E8125 by Joni Pajarinen.

1 Overview

- Reinforcement learning (RL) problem:
  - Denote that $\pi : O \to A$ is a policy that maps the observation to an action.
  - Determine a policy:

$$a = \pi(s) \tag{1}$$

  - s.t. the expected cumulative return is maximum, i.e.,

$$\pi^* = \arg\max_\pi \mathbb{E}[G] \tag{2}$$
$$G = \sum_t r_t \tag{3}$$

- Markov decision process (MDP):
  - We have an environment observable $z = s$, defined by a Markov dynamics defined as:

$$p(s_{t+1}|s_t, a_t) \tag{4}$$

    and a reward function

$$r_t = r(s_t, a_t) \tag{5}$$

  - The solution is formulated as follows:

$$a^*_{1,\dots,T} = \arg\max_{a_1,\dots,a_T} \sum_{t=1}^{T} r_t \tag{6}$$

    Represented as policy:

$$a = \pi(s) \tag{7}$$

- Connection between RL and MDP: RL is a MDP with unknown Markov dynamics $p(s_{t+1}|s_t, a_t)$, and unknown reward function r_t.
- Partially observable MDP (POMDP):
  - The environment is not directly observable.
  - Following MDP, POMDP is governed by a Markov dynamics $p(s_{t+1}|s_t, a_t)$ and reward function $r_t = r(s_t, a_t)$. In addition, we have an observation model $p(z_{t+1}|s_{t+1}, a_t)$.

2 Solving discrete MDP

- Markov property: future is independent of past conditioned on the present, i.e.,

$$p(s_{t+1}|s_t) = p(s_{t+1}|s_1, \dots, s_t) \tag{8}$$

- Markov process: a random process that generates a state sequences $\mathcal{S}$, following the Markov property. Markov process is defined as a tuple $(\mathcal{S}, T)$, where $T : \mathcal{S} \times \mathcal{S} \to [0, 1]$ denotes the state transition function.
- Markov reward process: defined by a tuple $(\mathcal{S}, T, r, \gamma)$:
  - $\mathcal{S}, T$ follows Markov process
  - $r : \mathcal{S} \to \mathcal{R}$ denotes the reward function
  - $\gamma \in [0, 1]$ denotes the discount factor
  - Accumulate reward in H horizon step (can be infinite):

$$G_t = \sum_{k=0}^{H} \gamma^k r_{t+k} \tag{9}$$

- State value function:

$$\begin{aligned} V(s) &= \mathbb{E}[G_t|s_t = s] && (10) \\ &= \mathbb{E}[r_t + \gamma V(s_{t+1})|s_t = s] && (11) \end{aligned}$$

- MDP: defined by a tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$
  - $\mathcal{S}, \gamma$ follows Markov reward process
  - $\mathcal{A}$ denotes set of actions
  - $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$
  - $R : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$ denotes the reward function
  - Goal: Find the policy $\pi(s)$ that maximizes $V(s)$
- Policy:
  - Deterministic: $\pi(s) : \mathcal{S} \to \mathcal{A}$
  - Stochastic: $\pi(a|s) \to [0, 1]$, i.e., distribution over actions.

- MDP value function:

$$
\begin{aligned}
V_\pi(s) &= \mathbb{E}_\pi[G_t|s_t = s] && (12)\\
&= \mathbb{E}_\pi[r_t + \gamma V_\pi(s_{t+1})|s_t = s] && (13)\\
&= r(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s') V_\pi(s') && (14)
\end{aligned}
$$

- Action-value function:

$$
\begin{aligned}
Q_\pi(s,a) &= \mathbb{E}_\pi[r_t + \gamma Q_\pi(s_{t+1}, a_{t+1}|s_t = s, a_t = a)] && (15)\\
&= r(s,a) + \gamma \sum_{s'} T(s,a,s') Q_\pi(s', \pi(s')) && (16)
\end{aligned}
$$

- Optimal value function:

$$
\begin{aligned}
V^*(s) &= \max_\pi V_\pi(s) && (17)\\
Q^*(s,a) &= \max_\pi Q_\pi(s,a) && (18)
\end{aligned}
$$

- Optimal policy:

$$
\begin{aligned}
\pi^*(s) &= \text{argmax}_a \mathbb{E}_{s'}[r(s,a) + \gamma V^*(s')] && (19)\\
&= \text{argmax}_a (r(s,a) + \gamma \sum_{s'} T(s,a,s') V^*(s')) && (20)
\end{aligned}
$$

- Iterative policy evaluation
  - Problem: Evaluate the value of policy π
  - Solution: Iterate Bellman expectation backs-up:

$$V_1 \to \cdots \to V_\pi$$

  - Apply synchronous back-ups:
    - For all s, update $V_{k+1}(s)$ from $V_k(s')$
    - Repeat

$$
\begin{aligned}
V_{k+1}(s) &= r(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s') V_k(s') && (21)\\
&= \sum_a \pi(a|s)(r(s,a) + \gamma \sum_{s'} T(s,a,s') V_k(s')) && (22)
\end{aligned}
$$

- Time complexity of value iteration:
  - Complexity $\mathcal{O}(|\mathcal{A}||\mathcal{S}|^2)$ per iteration.
  - Complexity when applied to action-value function: $\mathcal{O}(|\mathcal{A}|^2|\mathcal{S}|^2)$ per iteration.

3 RL in discrete domains

- Monte-Carlo policy evaluation
  - Complete episodes give samples of return G.
  - Learn the value of a particular policy from episodes under that policy.
  - Estimate value as an empirical mean return:

$$N(s) = N(s) + 1 \quad S(s) = S(s) + G_t \qquad V(s) \approx S(s)/N(s) \tag{23}$$

- Temporal difference: for each state transition, update a guess towards a guess:

$$V(s_t) = V(s_t) + \alpha(r_t + \gamma V(s_{t+1}) - V(s_t)) \tag{24}$$

- λ-return:
  - Combine returns in different horizons:

$$G_t^\lambda = (1 - \lambda) \sum_{k=0}^{\infty} \lambda^k G_t^k \tag{25}$$

  - State value function update (TD(λ)):

$$V(s_t) = V(s_t) + \alpha(G_t^\lambda - V(s_t)) \tag{26}$$

- Backward-TD(λ):
  - Extend TD time horizon with decay λ
  - After episode, update

$$V(s) = V(s) + \alpha E_t(s)(r_t + \gamma V(s_{t+1}) - V(s_t)) \tag{27}$$

$$E_t(s) = \gamma\lambda E_{t-1}(s) + 1(s_t = s) \tag{28}$$

- SARSA:
  - Apply TD to $Q(s, a)$

$$Q(s, a) = Q(s, a) + \alpha(r + \gamma Q(s', a') - Q(s, a)) \tag{29}$$

- SARSA(λ):
  - Apply TD(λ) to $Q(s, a)$
  - Backward SARSA(λ)

$$E_t(s, a) = \gamma\lambda E_{t-1}(s, a) + 1(s_t = s, a_t = a) \tag{30}$$

$$Q(s, a) = Q(s, a) + \alpha E_t(s, a)(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \tag{31}$$

- Q-learning:

$$Q(s, a) = Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a)) \tag{32}$$