Reinforcement Learning

Marshal Sinaga - marshal.sinaga@aalto.fi

2024-11-05

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

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

- 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:

$$
V^*(s) = \max_\pi V_\pi(s) \quad (17)
$$

$$
Q^*(s, a) = \max_\pi Q_\pi(s, a) \quad (18)
$$

- 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:

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

4 Optimal Control Problems

- Optimal control optimization (deterministic) objective:

$$\min_{a_1 \dots a_T} \sum_t c(s_t, a_t) \text{s.t.} s_{t+1} = f(s_t, a_t) \tag{33}$$

- Reparameterize:

$$\min_{a_1 \dots a_T} c(s_1, a_1) + c(f(s_1, a_1), a_2) + \cdots + c(f(f(\dots)), a_T) \tag{34}$$

- Linear quadratic regulator (LQR) problem definition:

$$f(s_t, a_t) = \begin{pmatrix} A_t & B_t \end{pmatrix} \begin{pmatrix} s_t \\ a_t \end{pmatrix} + f_t = F_t \begin{pmatrix} s_t \\ a_t \end{pmatrix} + f_t \tag{35}$$

$$c_t(s_t, a_t) = \frac{1}{2} \begin{pmatrix} s_t \\ a_t \end{pmatrix} C_t \begin{pmatrix} s_t \\ a_t \end{pmatrix} + \begin{pmatrix} s_t \\ a_t \end{pmatrix} c_t \tag{36}$$

where

$$C_t = \begin{pmatrix} C_{s_t,s_t} & C_{s_t,a_t} \\ C_{a_t,s_t} & C_{a_t,a_t} \end{pmatrix} \quad \text{and} \quad c_t = \begin{pmatrix} c_{s_t} \\ c_{a_t} \end{pmatrix} \tag{37}$$

- Action value function:

$$Q(s_T, a_T) = \text{const} + \frac{1}{2} \begin{pmatrix} s_T \\ a_T \end{pmatrix} C_T \begin{pmatrix} s_T \\ a_T \end{pmatrix} + \begin{pmatrix} s_T \\ a_T \end{pmatrix}^\top c_T \tag{38}$$

$$\nabla_{a_t} Q(s_T, a_T) = C_{s_T,a_T} = C_{a_T,s_T} + C_{a_T,a_T} a_t + c_{a_t} = 0 \tag{39}$$

$$a_T = -C^{-1}_{a_T,a_T}(C_{a_t,s_t} s_t +) \tag{40}$$

- Given the above action-value function, we find that the solution of Equation 34 can be written as follows:

$$a_T = K_T s_T + k_T \tag{41}$$

$$K_T = -C^{-1}_{a_T,a_T} C_{a_t,s_t} \tag{42}$$

$$k_T = -C_{a_T,a_T} c_{a_t} \tag{43}$$

- State-value function by substitution:

$$V(s_T) = \text{const} + \frac{1}{2} \begin{pmatrix} s_T \\ K_T s_T + k_T \end{pmatrix} C_T \begin{pmatrix} s_T \\ K_T s_T + k_T \end{pmatrix} + \begin{pmatrix} s_T \\ K_T s_T + k_T \end{pmatrix}^\top c_T \tag{44}$$

. It is quadratic in s_T

- Reparameterize:

$$Q_t = C_t + F_t^\top V_{t+1} F_t \tag{45}$$

$$q_t = c_t + F_t^\top V_{t+1} f_t + F_t^\top v_{t+1} \tag{46}$$

- The fact that $\nabla_{a_t} Q(s_t, a_t) = Q_{a_t,s_t} + Q_{a_t,a_t} a_t + q_t^\top = 0$ provides the following solutions:

$$a_t = K_t s_t + k_t \tag{47}$$
$$K_t = -Q_{a_t,a_t}^{-1} Q_{a_t,s_t} \tag{48}$$
$$k_t = -Q a_t, a_t^{-1} q_{a_t} \tag{49}$$

- LQR algorithm
  - Backward recursion:
    For t = T down to 1:
    * $Q_t = C_t + F_t^\top V_{t+1} F_t$
    * $q_t = c_t + F_t^\top V_{t+1} f_t + F_t^\top v_{t+1}$
    * $K_t = -Q_{a_t,a_t}^{-1} Q_{a_t,s_t}$
    * $k_t = -Q_{a_t,a_t}^{-1} q_{a_t}$
    * $V_t = Q_{s_t,s_t} + Q_{s_t,a_t} K_t + K_t^\top Q_{a_t,s_t} + K_t^\top Q_{a_t,a_t} K_t$
    * $v_t = q_{s_t} + Q_{s_t,a_t} k_t + K_t^\top q_{a_t} + K_t^\top Q_{a_t,a_t} k_t$
  - Forward recursion:
    For t = 1 to T:
    * $a_t = K_t s_t + k_t$
    * $s_{t+1} = f(s_t, a_t)$

- LQR with stochastic dynamics:

$$f(s_t, a_t) = F_t \begin{pmatrix} s_t \\ a_t \end{pmatrix} + f_t + w_t \quad w_t \sim \mathcal{N}(0, \Sigma_t) \tag{50}$$
$$p(s_{t+1}|s_t, a_t) \sim \mathcal{N}\left(F_t \begin{pmatrix} s_t \\ a_t \end{pmatrix} + f_t, \Sigma_t\right) \tag{51}$$

- Solving non-linear systems with LQR:
  Approximate a non-linear system as linear-quadratic:

$$f(s_t, a_t) \approx f(\hat{s}_t, \hat{a}_t) + \nabla_{s_t,a_t} f(\hat{s}_t, \hat{a}_t) \begin{pmatrix} s_t - \hat{s}_t \\ a_t - \hat{a}_t \end{pmatrix} \tag{52}$$
$$c_t(s_t, a_t) \approx c(\hat{s}_t, \hat{a}_t) + \frac{1}{2} \begin{pmatrix} s_t - \hat{s}_t \\ a_t - \hat{a}_t \end{pmatrix} \nabla^2_{s_t,a_t} c(\hat{s}_t, \hat{a}_t) \begin{pmatrix} s_t - \hat{s}_t \\ a_t - \hat{a}_t \end{pmatrix} \tag{53}$$
$$+ \nabla_{s_t,a_t} c(\hat{s}_t, \hat{a}_t) \begin{pmatrix} s_t - \hat{s}_t \\ a_t - \hat{a}_t \end{pmatrix} \tag{54}$$

5 Policy Gradient

- Update policy parameters:

$$\theta_{m+1} = \theta_m + \alpha_m \nabla_\theta R|_{\theta=\theta_m} \quad s.t. \sum_{m=0}^{\infty} \alpha_m = \infty \quad \sum_{m=0}^{\infty} \alpha_m^2 < \infty \tag{55}$$

  where

$$R(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r_t\right] \tag{56}$$

- Likelihood-ratio approach:
  Assume each trajectory τ is generated by a roll-out, thus

$$\tau \sim p_\theta(\tau) = p(\tau|\theta) \quad R(\tau) = \sum_{t=0}^{H} \gamma^t r_t \tag{57}$$

  Expected return:

$$R(\theta) = \mathbb{E}_\tau[R(\tau)] = \int p_\theta(\tau) R(\tau) d\tau \tag{58}$$

  Gradient:

$$\nabla_\theta R(\theta) = \int \nabla_\theta p_\theta(\tau) R(\tau) d\tau \tag{59}$$

$$= \int p_\theta(\tau) \nabla_\theta \log p_\theta(\tau) R(\tau) d\tau \tag{60}$$

$$= \mathbb{E}_\tau[\nabla_\theta \log p_\theta(\tau) R(\tau)] \tag{61}$$

- Monte Carlo policy gradient $\rightarrow$ REINFORCE
  1. Perform J episodes $i = 1, \ldots, J$
  2. Estimate gradient $g_{\text{REINFORCE}} = \mathbb{E}_\tau \left[(\nabla_\theta \log \pi_\theta(a_t|s_t)) \, R(i) \right]$
     $\approx \frac{1}{J} \sum_{i=1}^{J} \left[\left(\nabla_\theta \log \pi_\theta(a_t^{[i]}|s_t^{[i]}) \right) \left(\sum_t \gamma^t r_{t,i} \right) \right]$
  3. Update policy and repeat with new trials until convergence.
- Decreasing variance by adding baseline:

$$\nabla_\theta R(\theta) = \mathbb{E}_\tau[\nabla_\theta \log p_\theta(\tau)(R(\tau) - b)] = \mathbb{E}_\tau[\nabla_\theta \log p_\theta(\tau) R(\tau)] \tag{62}$$

- It does not cause bias since

$$\mathbb{E}_\tau[\nabla_\theta \log p_\theta(\tau) b] = \int p_\theta(\tau) \nabla_\theta \log p_\theta(\tau) b d\tau \tag{63}$$

$$\int \nabla_\theta p_\theta(\tau) b d\tau = b \nabla_\theta \int p_\theta(\tau) d\tau = b \nabla_\theta 1 = 0 \tag{64}$$

- Episodic REINFORCE with optimal baseline
  Optimal baseline for episodic REINFORCE (minimize variance of estimator):

$$b_h = \frac{\mathbb{E}_\tau \left[\left(\sum_{t=0}^{H} \nabla_{\theta_h} \log \pi_\theta(a_t|s_t)^2 R_\tau \right) \right]}{\mathbb{E}_\tau [\sum_{t=0}^{H} (\nabla_{\theta_h} \log \pi_\theta(a_t|s_t))^2]} \tag{65}$$

- 1. Perform J trials $i = 1, \ldots, J$:
  2. For each gradient element h:
     Estimate optimal baseline b_h
     Estimate gradient $g_h = \frac{1}{J} \sum_{i=1}^{J} \left[(\sum_{t=0}^{H} \nabla_{\theta_h} \log \pi_\theta(a_t^{[i]}|s_t^{[i]}))(R(i) - b_h^{[i]}) \right]$

3. Repeat until convergence

- Off-policy policy gradient: optimize $\mathbb{E}_{\tau\sim\pi_\theta(\tau)}[R(\tau)]$ using samples from $\pi'(\tau)$.
- Importance sampling: $\mathbb{E}_{\tau\sim\pi_\theta(\tau)}[R(\tau)] = \mathbb{E}_{\tau\sim pi'(\tau)}[\frac{\pi_\theta(\tau)}{\pi'(\tau)}R(\tau)]$

$$\frac{\pi_\theta(\tau)}{\pi'(\tau)} = \frac{p(s_0)\prod_{t=0}^{H} p(s_{t+1}|s_t,a_t)\pi_\theta(a_t|s_t)}{p(s_0)\prod_{t=0}^{H} p(s_{t+1}|s_t,a_t)\pi'(a_t|s_t)} = \frac{\prod_{t=0}^{H}\pi_\theta(a_t|s_t)}{\prod_{t=0}^{H} pi'(a_t|s_t)} \tag{66}$$

- The gradient:

$$\nabla_\theta \mathbb{E}_{\tau\sim\pi'(\tau)}\left[\frac{\pi_\theta(\tau)}{\pi'(\tau)R(\tau)}\right] = \mathbb{E}_{\tau\sim\pi'(\tau)}[\frac{\nabla_\theta\pi_\theta(\tau)}{\pi'(\tau)}R(\tau)] \tag{67}$$

$$= \mathbb{E}_{\tau\sim\pi'(\tau)}\left[\left(\prod_t \frac{\pi_\theta(a_t|s_t)}{\pi'(a_t|s_t)}\right)\left(\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t)\right)\left(\sum_t \gamma^t r_t\right)\right] \tag{68}$$

6 Exploration and Exploitation

- Greedy approach in multi-armed bandit:

$$a^* = \text{argmax}_{a\in\mathcal{A}} Q(a) \tag{69}$$

$$Q(a) = \frac{1}{N(a)}\sum_{n=1}^{N(a)} r_n(a) \tag{70}$$

- Hoeffding's inequality: Given random variables $X_1,\ldots,X_M \in [0,1]$, where $\bar{X}_M = \frac{1}{M}\sum_{m=1}^{M} X_m$, it holds that $\mathbb{P}(\mathbb{E}[X] > \bar{X}_M + u) \leq \exp(-2Mu^2)$
- Applying Hoeffding's inequality on a bandit action a:

$$\mathbb{P}(\mathbb{E}[Q(a)] > Q(a) + U(a)) \leq \exp(-2N(a)U^2(a)) \tag{71}$$

- Limit probability of true value to exceed upper bound:

$$\begin{aligned}&\mathbb{P}(\mathbb{E}[Q(a)] > Q(a) + U(a)) \leq \exp(-2N(a)U^2(a)) = p\\ &\to U(a) = \sqrt{-1/2\log p/N(a)}\end{aligned} \tag{72}$$

- Setting $p = N^{-4}$ yields:

$$\hat{Q}(a) = Q(a) + \sqrt{2\log N/N(a)} \tag{73}$$

- Thompson sampling: taking action $a^* \in \mathcal{A}$ according to the probability that it maximizes the expected reward; a^* is chosen with a probability

$$\int \mathbb{I}[\mathbb{E}[r|a^*,x,\theta] = \max_{a^*}\mathbb{E}[r|a',x,\theta]]p(\theta|\mathcal{D})d\theta, \tag{74}$$

where θ denotes the reward's parameters.