CS 188 Spring 2019 | Introduction to Artificial Intelligence | Written HW 4 Sol.

Self-assessment due: Monday 3/4/2019 at 11:59pm (submit via Gradescope)

For the self assessment, **fill in the self assessment boxes in your original submission** (you can download a PDF copy of your submission from Gradescope). For each subpart where your original answer was correct, write "correct." Otherwise, write and explain the correct answer.

Q1. Reinforcement Learning

Imagine an unknown game which has only two states $\{A, B\}$ and in each state the agent has two actions to choose from: {Up, Down}. Suppose a game agent chooses actions according to some policy π and generates the following sequence of actions and rewards in the unknown game:

t	s_t	a_t	s_{t+1}	r_t
0	A	Down	B	2
1	B	Down	B	-4
2	B	Up	B	0
3	B	Up	A	3
4	A	Up	A	-1

Unless specified otherwise, assume a discount factor $\gamma = 0.5$ and a learning rate $\alpha = 0.5$

(a) Recall the update function of Q-learning is:

$$Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a'} Q(s_{t+1}, a'))$$

Assume that all Q-values initialized as 0. What are the following Q-values learned by running Q-learning with the above experience sequence?

$Q(\text{A}, \text{Down}) =$ __1__, $Q(\text{B}, \text{Up}) =$ __$\frac{7}{4}$__

Perform Q-learning update 4 times, once for each of the first 4 observations.

(b) In model-based reinforcement learning, we first estimate the transition function $T(s, a, s')$ and the reward function $R(s, a, s')$. Fill in the following estimates of T and R, estimated from the experience above. Write "n/a" if not applicable or undefined.

$\hat{T}(\text{A}, \text{Up}, \text{A}) =$ __1__, $\hat{T}(\text{A}, \text{Up}, \text{B}) =$ __0__, $\hat{T}(\text{B}, \text{Up}, \text{A}) =$ __$\frac{1}{2}$__, $\hat{T}(\text{B}, \text{Up}, \text{B}) =$ __$\frac{1}{2}$__

$\hat{R}(\text{A}, \text{Up}, \text{A}) =$ __-1__, $\hat{R}(\text{A}, \text{Up}, \text{B}) =$ __n/a__, $\hat{R}(\text{B}, \text{Up}, \text{A}) =$ __3__, $\hat{R}(\text{B}, \text{Up}, \text{B}) =$ __0__

Count transitions above and calculate frequencies. Rewards are observed rewards.

(c) To decouple this question from the previous one, assume we had **a different experience** and ended up with the following estimates of the transition and reward functions:

s	a	s'	$\hat{T}(s, a, s')$	$\hat{R}(s, a, s')$
A	Up	A	1	10
A	Down	A	0.5	2
A	Down	B	0.5	2
B	Up	A	1	-5
B	Down	B	1	8

(i) Give the optimal policy $\hat{\pi}^*(s)$ and $\hat{V}^*(s)$ for the MDP with transition function $\hat{T}$ and reward function $\hat{R}$.
Hint: for any $x \in \mathbb{R}$, $|x| < 1$, we have $1 + x + x^2 + x^3 + x^4 + \cdots = 1/(1-x)$.

$\hat{\pi}^*(A) =$ __*Up*__, $\hat{\pi}^*(B) =$ __*Down*__, $\hat{V}^*(A) =$ __20__, $\hat{V}^*(B) =$ __16__.

Find the optimal policy first, and then use optimal policy to calculate the value function using a Bellman equation.

(ii) If we repeatedly feed this new experience sequence through our Q-learning algorithm, what values will it converge to? Assume the learning rate α_t is properly chosen so that convergence is guaranteed.

● the values found above, $\hat{V}^*$

- ◯ the optimal values, V^*
- ◯ neither $\hat{V}^*$ nor V^*
- ◯ not enough information to determine

The Q-learning algorithm will not converge to the optimal values V^* for the MDP because the experience sequence and transition frequencies replayed are not necessarily representative of the underlying MDP. (For example, the true $T(A, Down, A)$ might be equal to 0.75, in which case, repeatedly feeding in the above experience would not provide an accurate sampling of the MDP.) However, for the MDP with transition function $\hat{T}$ and reward function $\hat{R}$, replaying this experience repeatedly will result in Q-learning converging to its optimal values $\hat{V}^*$.

Q2. Policy Evaluation

In this question, you will be working in an MDP with states S, actions A, discount factor γ, transition function T, and reward function R.

We have some fixed policy $\pi : S \to A$, which returns an action $a = \pi(s)$ for each state $s \in S$. We want to learn the Q function $Q^\pi(s, a)$ for this policy: the expected discounted reward from taking action a in state s and then continuing to act according to π: $Q^\pi(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma Q^\pi(s', \pi(s'))]$. The policy π will not change while running any of the algorithms below.

(a) Can we guarantee anything about how the values Q^π compare to the values Q^* for an optimal policy π^*?

- ● $Q^\pi(s, a) \leq Q^*(s, a)$ for all s, a
- ○ $Q^\pi(s, a) = Q^*(s, a)$ for all s, a
- ○ $Q^\pi(s, a) \geq Q^*(s, a)$ for all s, a
- ○ None of the above are guaranteed

(b) Suppose T and R are *unknown*. You will develop sample-based methods to estimate Q^π. You obtain a series of *samples* $(s_1, a_1, r_1), (s_2, a_2, r_2), \ldots (s_T, a_T, r_T)$ from acting according to this policy (where $a_t = \pi(s_t)$, for all t).

(i) Recall the update equation for the Temporal Difference algorithm, performed on each sample in sequence:

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

which approximates the expected discounted reward $V^\pi(s)$ for following policy π from each state s, for a learning rate α.

Fill in the blank below to create a similar update equation which will approximate Q^π using the samples.

You can use any of the terms $Q, s_t, s_{t+1}, a_t, a_{t+1}, r_t, r_{t+1}, \gamma, \alpha, \pi$ in your equation, as well as $\sum$ and max with any index variables (i.e. you could write $\max_a$, or $\sum_a$ and then use a somewhere else), but no other terms.

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

(ii) Now, we will approximate Q^π using a linear function: $Q(s, a) = \sum_{i=1}^{d} w_i f_i(s, a)$ for weights $w_1, \ldots, w_d$ and feature functions $f_1(s, a), \ldots, f_d(s, a)$.

To decouple this part from the previous part, use Q_{samp} for the value in the blank in part (i) (i.e. $Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha Q_{samp}$).

Which of the following is the correct sample-based update for each w_i?

- ○ $w_i \leftarrow w_i + \alpha[Q(s_t, a_t) - Q_{samp}]$
- ○ $w_i \leftarrow w_i - \alpha[Q(s_t, a_t) - Q_{samp}]$
- ○ $w_i \leftarrow w_i + \alpha[Q(s_t, a_t) - Q_{samp}]f_i(s_t, a_t)$
- ● $w_i \leftarrow w_i - \alpha[Q(s_t, a_t) - Q_{samp}]f_i(s_t, a_t)$
- ○ $w_i \leftarrow w_i + \alpha[Q(s_t, a_t) - Q_{samp}]w_i$
- ○ $w_i \leftarrow w_i - \alpha[Q(s_t, a_t) - Q_{samp}]w_i$

(iii) The algorithms in the previous parts (part i and ii) are:

☐ model-based ■ model-free