Aren Ashlock
Department of Computer Science
Iowa State University

April 24, 2024

HW7 — Reinforcement Learning

1. Consider the "FrozenLake-v1" task (see link) and the 4×4 map (let `is_slippery` be True). Recall the MDP $(S, \mathcal{A}, \mathcal{R}, \mathbb{P}, \gamma)$ and let $\gamma = 0.9$:

   (a) What is the cardinality of S (note: for a set X of finite elements, the cardinality of X, $|X|$, refers to the number of elements in X)?

   Answer: 16

   (b) What is the cardinality of $\mathcal{A}$?

   Answer: 4

   (c) Consider the state-action pair with state s being the fourth row and the third column of the map, and the action a being 3 (move-up), what is $\mathcal{R}(s, a)$?

   Answer: $\mathcal{R} = \begin{cases} 0 & 67\% \\ 1 & 33\% \end{cases}$

   (d) Consider the same state-action pair from the above question, what is $\mathbb{P}(s, a)$?

   Answer: $\mathbb{P} = \begin{cases} 15 \to 11 & 33\% \\ 15 \to 14 & 33\% \\ 15 \to 16 & 33\% \end{cases}$

   (e) Let $Q'(s, a) = 1, \forall s, a$. Is Q' the optimal Q function? Why?

   Answer: Q' is not the optimal policy because $Q'(s, a) = 1, \forall s, a$ essentially means that taking any action in any state is optimal. However, we know that some state-action pairs are not optimal since they will lose you the game (or not win as efficiently).

   (f) Let the length of a single episode be 2 (rather than 100), $\gamma = 0.9$ and the policy $\pi(s) = 2, \forall s \in S$, consider the same state-action pair from (c), what is $V^{\pi}(s)$? Show me your calculation steps (or code). (Note: please give explicit values, not just the equations. If you plan to use programming to solve these problems (manual calculation is very possible with the modified episode length), you don't need to install gym or run FrozenLake, your programming is to help you with something else, and it only requires basic and standard python packages.)

   Answer: $V^{\pi}(s) = 1.9/7 = 0.2714$
   $15 \to 16 = 0.9^0 \times 1 = 1$
   $15 \to 11 \to 12 = 0.9^0 \times 0 + 0.9^1 \times 0 = 0$
   $15 \to 11 \to 7 = 0.9^0 \times 0 + 0.9^1 \times 0 = 0$
   $15 \to 11 \to 15 = 0.9^0 \times 0 + 0.9^1 \times 0 = 0$
   $15 \to 15 \to 16 = 0.9^0 \times 0 + 0.9^1 \times 1 = 0.9$
   $15 \to 15 \to 11 = 0.9^0 \times 0 + 0.9^1 \times 0 = 0$
   $V15 \to 15 \to 15 = 0.9^0 \times 0 + 0.9^1 \times 0 = 0$

   (g) Consider a parameterized Q-function in the linear form, i.e.,

   $$\hat{Q}(s, a; \omega, b) = \omega^T \begin{bmatrix} s \\ a \end{bmatrix} + b, \omega \in \mathbb{R}^d, b \in \mathbb{R}. \tag{1-1}$$

   i. What is the value of d?
   Answer: 2

ii. Assume ω is a vector with all-one entries and $b = 2$. Is $\hat{Q}$ the optimal Q function? Why?
Answer: No, because the all-one entries makes it so that the max $\hat{Q}$ is when you move up since that has the highest encoded value. Therefore, it won't optimally converge to the optimum since we know you need to move down at some point.

iii. Consider the policy $\hat{\pi}(s) = \arg\max_a \hat{Q}(s, a; \omega, b)$, and any linear Q-function of the form (1-1). For any fixed set of parameters satisfying $\omega \neq \mathbf{0}, b \neq 0$, is it possible to have different action selections for different states with the given policy and $\hat{Q}$? Why?
Answer: Yes, because ω and b affect the state-action pairs differently based on the values given to ω and b. This is combined with the fact that the environment is stochastic, so there is probability involved in the action that is taken. Therefore, it is possible to have different action selections for different states.

iv. Consider the parameterized Q-function as described in (1-1), and let ω be a vector with all-one entries and $b = 2$. Given (s, a) the same state-action pair from (c), let s' be the third-row and the third-column of the map, what is the *temporal difference value,* δ, given the transition (s, a, s')? What is the Huber loss value given the transition (s, a, s')? (Please give both the equation and the explicit value output.)
Temporal Difference Value: $\delta = r + \gamma max_{a'} Q(s', a'; \theta) - Q(s, a; \theta) = 0 + 0.9 \times (1(11) + 1(3) + 2) - (1(15) + 1(3) + 2) = 0 + 0.9 \times 16 - 20 = 0 + 14.4 - 20 = -5.6$
Huber Loss: First, determine which equation to use. $|y - Q(s, a; \theta)| = |0 + 0.9(1(11) + 1(3) + 2) - (1(15) + 1(3) + 2)| = |0 + 0.9(16) - 20| = |0 + 14.4 - 20| = |-5.6| = 5.6$, which $5.6 > 1$, so I will use the 2nd equation for Huber loss...
$L(\theta) = \delta \times (|y - Q(s, a; \theta)| - \frac{1}{2}\delta) = 1 \times (5.6 - \frac{1}{2}(1)) = 5.6 - 0.5 = 5.1$

2. Should "epsilon greedy policy" and "experience replay" typically used in DQN training also be used in all tabular Q-learning tasks? If yes, why? If no, should they at least be considered for some tabular Q-learning tasks? What qualities should those tasks have?

Answer: I don't think "epsilon greedy policy" and "experience replay" should be used in ALL tabular Q-learning tasks. Typically, tabular is used when the table of states and actions will be smaller, so exploitation is typically favored. However, the two techniques should be considered when the table is larger or sparse. This would allow for more exploration to be more efficient and converge faster towards the optimum.

3. Read the paper Deep Reinforcement Learning with Double Q-learning.

(a) In comparison with DQN learning introduced in the lecture, what makes the proposal different? What are the advantages? What are the disadvantages?

Answer: This proposal is different because DQN calculates the target value using only 1 set of weight parameters, whereas the proposal calculates with 2 different sets of weight parameters. In the proposal, θ is for estimating while θ' is for evaluating. The advantage is that this proposal handles overestimation, which is a known issue with DQN. As for a disadvantage, having to calculate based on 2 different sets of weights adds overhead.

(b) Consider the tabular Q-learning algorithm (i.e., equation (10) on page 17 of the 9-RL-I slides), how would you implement a tabular double-Q learning algorithm? Write down the updates in equation(s) following a format that is **as close as possible** to equation (10) in our lecture slides.

Answer: My answer utilizes two Q-tables with each update referencing the other table...
$Q_1(s, a) = (1 - \alpha)Q_1(s, a) + \alpha(r + \gamma Q_2(s', max_{a'} Q_1(s', a')))$
$Q_2(s, a) = (1 - \alpha)Q_2(s, a) + \alpha(r + \gamma Q_1(s', max_{a'} Q_2(s', a')))$

Submitted by Aren Ashlock on April 24, 2024.