Introduction to Artificial Intelligence

Exam Prep 5 Solutions

Q1. 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: $Q(s, a) = \mathbf{w}^\top \mathbf{f}(s, a)$ for a weight vector $\mathbf{w}$ and feature function $\mathbf{f}(s, a)$.

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

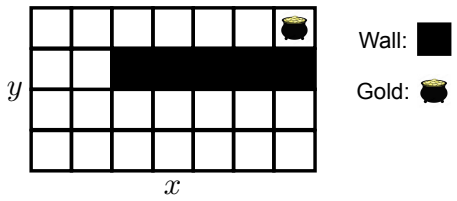
Which of the following is the correct sample-based update for $\mathbf{w}$?

- ( ) $\mathbf{w} \leftarrow \mathbf{w} + \alpha[Q(s_t, a_t) - Q_{samp}]$
- ( ) $\mathbf{w} \leftarrow \mathbf{w} - \alpha[Q(s_t, a_t) - Q_{samp}]$
- ( ) $\mathbf{w} \leftarrow \mathbf{w} + \alpha[Q(s_t, a_t) - Q_{samp}]\mathbf{f}(s_t, a_t)$
- (●) $\mathbf{w} \leftarrow \mathbf{w} - \alpha[Q(s_t, a_t) - Q_{samp}]\mathbf{f}(s_t, a_t)$
- ( ) $\mathbf{w} \leftarrow \mathbf{w} + \alpha[Q(s_t, a_t) - Q_{samp}]\mathbf{w}$
- ( ) $\mathbf{w} \leftarrow \mathbf{w} - \alpha[Q(s_t, a_t) - Q_{samp}]\mathbf{w}$

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

- [ ] model-based
- [■] model-free

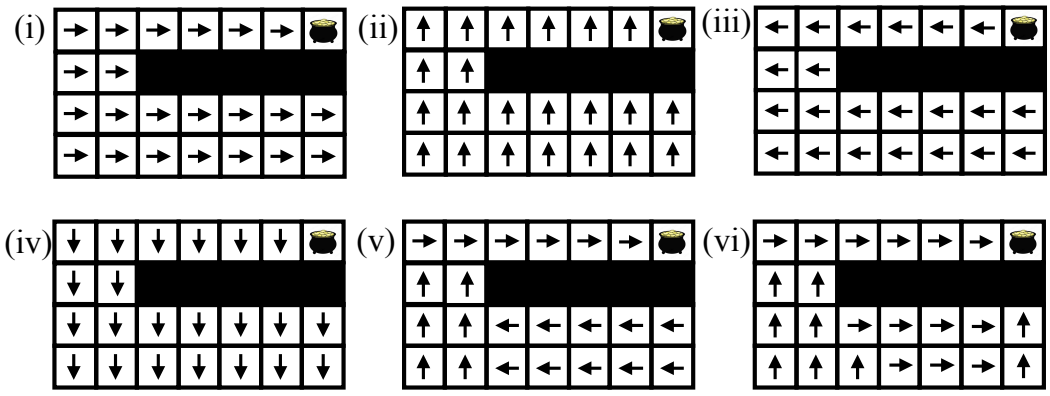
Q2. MDPs & RL



Consider the grid-world MDP above. The goal of the game is to reach the pot of gold. As soon as you land on the pot of gold you receive a reward and the game ends. Your agent can move around the grid by taking the following actions: North, South, East, West. Moving into a square that is not a wall is always successful. If you attempt to move into a grid location occupied by a wall or attempt to move off the board, you remain in your current grid location.

Our goal is to build a value function that assigns values to each grid location, but instead of keeping track of a separate number for each location, we are going to use features. Specifically, suppose we represent the value of state (x, y) (a grid location) as $V(x, y) = w^\top f(x, y)$. Here, $f(x, y)$ is a feature function that maps the grid location (x, y) to a vector of features and w is a weight vector that parameterizes our value function (note that entries in w can be any real number, positive or negative).

In the next few questions, we will look at various possible feature functions $f(x, y)$. We will think about the value functions that are representable using each set of features, and, further, think about which policies could be extracted from those value functions. Assume that when a policy is extracted from a value function, ties can be broken arbitrarily. In our definition of feature functions we will make use of the location of the pot of gold. Let the gold's location be (x^*, y^*). Keep in mind the policies (i), (ii), (iii), (iv), (v), and (vi) shown below.



(a) Suppose we use a single feature: the x-distance to the pot of gold. Specifically, suppose $f(x, y) = |x - x^*|$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

● (i) ○ (ii) ● (iii) ○ (iv) ○ (v) ○ (vi)

(b) Suppose we use a single feature: the y-distance to the pot of gold. Specifically, suppose $f(x, y) = |y - y^*|$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

○ (i) ● (ii) ○ (iii) ● (iv) ○ (v) ○ (vi)

(c) Suppose we use a single feature: the Manhattan distance to the pot of gold. Specifically, suppose $f(x, y) = |x - x^*| + |y - y^*|$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

○ (i) ○ (ii) ○ (iii) ○ (iv) ○ (v) ● (vi)

(d) Suppose we use a single feature: the length of the shortest path to the pot of gold. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w is not allowed to be 0. Fill in all that apply.

○ (i) ○ (ii) ○ (iii) ○ (iv) ● (v) ○ (vi)

(e) Suppose we use two features: the x-distance to the pot of gold and the y-distance to the pot of gold. Specifically, suppose $f(x, y) = (|x - x^*|, |y - y^*|)$. Which of the policies could be extracted from a value function that is representable using this feature function? Assume the weights vector w must have at least one non-zero entry. Fill in all that apply.

● (i) ● (ii) ● (iii) ● (iv) ○ (v) ● (vi)

Q3. RL: Amusement Park

After the disastrous waterslide experience you decide to go to an amusement park instead. In the previous questions the MDP was based on a single ride (a water slide). Here our MDP is about choosing a ride from a set of many rides.

You start off feeling well, getting positive rewards from rides, some larger than others. However, there is some chance of each ride making you sick. If you continue going on rides while sick there is some chance of becoming well again, but you don't enjoy the rides as much, receiving lower rewards (possibly negative).

You have never been to an amusement park before, so you don't know how much reward you will get from each ride (while well or sick). You also don't know how likely you are to get sick on each ride, or how likely you are to become well again. What you do know about the rides is:

Actions / Rides	Type	Wait	Speed
Big Dipper	Rollercoaster	Long	Fast
Wild Mouse	Rollercoaster	Short	Slow
Hair Raiser	Drop tower	Short	Fast
Moon Ranger	Pendulum	Short	Slow
Leave the Park	Leave	Short	Slow

We will formulate this as an MDP with two states, well and sick. Each ride corresponds to an action. The 'Leave the Park' action ends the current run through the MDP. Taking a ride will lead back to the same state with some probability or take you to the other state. We will use a feature based approximation to the Q-values, defined by the following four features and associated weights:

Features	Initial Weights
$f_0(state, action) = 1$ (this is a bias feature that is always 1)	$w_0 = 1$
$f_1(state, action) = \begin{cases} 1 & \text{if } action \text{ type is Rollercoaster} \\ 0 & \text{otherwise} \end{cases}$	$w_1 = 2$
$f_2(state, action) = \begin{cases} 1 & \text{if } action \text{ wait is Short} \\ 0 & \text{otherwise} \end{cases}$	$w_2 = 1$
$f_3(state, action) = \begin{cases} 1 & \text{if } action \text{ speed is Fast} \\ 0 & \text{otherwise} \end{cases}$	$w_3 = 0.5$

(a) Calculate Q(*'Well'*, *'Big Dipper'*):

$1 + 2 + 0 + 0.5 = 3.5$

(b) Apply a Q-learning update based on the sample (*'Well'*, *'Big Dipper'*, *'Sick'*, -10.5), using a learning rate of $\alpha = 0.5$ and discount of $\gamma = 0.5$. What are the new weights?

Difference $= -10.5 + 0.5 * max(4, 3.5, 2.5, 2.0, 2.0) - 3.5 = -12$
$w_0 = 1 - 6 * 1 = -5$
$w_1 = 2 - 6 * 1 = -4$
$w_2 = 1 - 6 * 0 = 1$
$w_3 = 0.5 - 6 * 1 = -5.5$

(c) Using our approximation, are the Q-values that involve the sick state the same or different from the corresponding Q-values that involve the well state? In other words, is Q(*'Well'*, action) = Q(*'Sick'*, action) for each possible action? Why / Why not? (in just one sentence)

Same
They are the same because we have no features that distinguish between the two states.

Now we will consider the exploration / exploitation tradeoff in this amusement park.

(d) Assume we have the original weights from the table on the previous page. What action will an ϵ-greedy approach choose from the well state? If multiple actions could be chosen, give each action and its probability.

With probability $(1 - \epsilon\frac{4}{5})$ we will choose the Wild Mouse. Each other action will be chosen with probability $\frac{\epsilon}{5}$

(e) When running Q-learning another approach to dealing with this tradeoff is using an exploration function:

$$f(u, n) = u + \frac{k}{n}$$

(i) How is this function used in the Q-learning equations? (a single sentence)
The update replaces the max over Q values with a max over this function (with Q and N as arguments)

What are each of the following? (a single sentence each)

(ii) u :
The utility, given by Q

(iii) n :
The number of times this state-action pair has been visited

(iv) k :
A constant, by adjusting it we can change how optimistic we are about states we haven't visited much.