3. Q-learning Algorithm

Project due May 10, 2023 08:59 -03 Completed

In this section, you will implement the Q-learning algorithm, which is a model-free algor optimal Q-function. In the tabular setting, the algorithm maintains the Q-value for all po pairs. Starting from a random Q-function, the agent continuously collects experiences updates its Q-function.

From now on, we will refer to $c = (a, b)$ as "an action" although it really is an action wit

Q-learning Algorithm

- The agent plays an action c at state s, getting a reward $R(s, c)$ and observing the n
- Update the single Q-value corresponding to each such transition:

$$Q(s, c) \leftarrow (1 - \alpha) Q(s, c) + \alpha [R(s, c) + \gamma \max_{c' \in C} Q(s', c')]$$

Tip: We recommend you implement all functions from this tab and the next one before online. Make sure you achieve reasonable performance on the *Home World* game

---

Single step update

1.0/1 point (graded)

Write a function `tabular_q_learning` that updates the single Q-value, given the trans $(s, c, R(s, c), s')$.

Reminder: You should implement this function locally first. You can read through the ne the context in which this function is called

Available Functions: You have access to the NumPy python library as `np`. You should `ALPHA` and `GAMMA` in your code

```
def tabular_q_learning(q_func, current_state_1, current_state_2, action_ind
                       object_index, reward, next_state_1, next_state_2,
                       terminal):
    """Update q_func for a given transition

    Args:
        q_func (np.ndarray): current Q-function
        current_state_1, current_state_2 (int, int): two indices describing
        action_index (int): index of the current action
        object_index (int): index of the current object
```

Epsilon-greedy exploration

1.0/1 point (graded)

Note that the Q-learning algorithm does not specify how we should interact in the worl
It merely updates the values based on the experience collected. If we explore randomly
actions at random, we would most likely not get anywhere. A better option is to exploit
learned, as summarized by current Q-values. We can always act greedily with respect t
i.e., take an action . Of course, early on, these are not ne
actions. For this reason, a typical exploration strategy is to follow a so-called -greedy
take a random action out of with probability follow
balances exploration vs exploitation. A large value of means exploring more (randoml
what we have learned. A small , on the other hand, will generate experience consisten
estimates of Q-values.

Now you will write a function `epsilon_greedy` that implements the -greedy explorat
current Q-function.

Reminder: You should implement this function locally first. You can read through the ne
the context in which this function is called

Available Functions: You have access to the NumPy python library as `np`. Your code
constants `NUM_ACTIONS` and `NUM_OBJECTS`.

```
def epsilon_greedy(state_1, state_2, q_func, epsilon):
    """Returns an action selected by an epsilon-Greedy exploration policy

    Args:
        state_1, state_2 (int, int): two indices describing the current sta
        q_func (np.ndarray): current Q-function
        epsilon (float): the probability of choosing a random command

    Returns:
        (int, int): the indices describing the action/object to take
    """
    coin = np.random.random_sample()

    if coin < epsilon:
        action_index = np.random.randint(NUM_ACTIONS)
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Test results

? Random index value of Epsilon-greedy exploration
I passed the first test case (epsilon==0) of Epsilon-greedy exploration and failed on the second test case(ep

"if random.random() < epsilon" for calculating <epsilon probability does not work for epsilon-
Hi with same implementation for rest of code, but changing the if condition from what is mentioned in the titl

? Question on the "terminal" flag for Single-step update
I'm not clear what is meant to happen in the single step update when the terminal parameter is True. I figured

? The grader on the greedy epsilon question picks a Q-value that is not the maximum
After a few tries and misses on the epsilon greedy, I decided to add some prints to see what the Q matrix loo

Meaning of (s,c,R(s,c),s')

q_func
Is q_func a 4-D matrix? If it is a 4-D matrix why is it called q_func instead of q_values?

? Q_func and states
I have two questions. 1. Why are there two indexes for current_state? I thought state is just a room index out

? Basic question on q_func input
A very basic (or even silly) question: I can see for q_func input: q_func.shape is (2, 5, 3, 7) I understand there

Single step update : tabular_q_learning
For this function are we setting q_func[current_state_1, current_state_2, action_index, object_index] = 0 but i

Single Step answers off by < 0.01
My answers for the single step are off by less than 0.01. I seem to have entered in the formula correctly. Rath

? Absolutely no idea what "terminal" means
Getting correct output for false terminal, can't get the correct output for true terminal. According to framewo

Epsilon-greedy exploration