Course › Unit 5 Reinforcement Learning (2 weeks) › Project 5: Text-Based Game › 3. Q-learning Algorithm

**Audit Access Expires May 25, 2020**
You lose all access to this course, including your progress, on May 25, 2020.

# 3. Q-learning Algorithm

In this section, you will implement the Q-learning algorithm, which is a model-free algorithm used to learn an optimal Q-function. In the tabular setting, the algorithm maintains the Q-value for all possible state-action pairs. Starting from a random Q-function, the agent continuously collects experiences $(s, c, R(s, c), s')$ and updates its Q-function.

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

**Q-learning Algorithm**

- The agent plays an action $c$ at state $s$, getting a reward $R(s, c)$ and observing the next state $s'$.

- Update the single Q-value corresponding to each such transition:

Generating Speech Output

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

**Tip:** We recommend you implement all functions from this tab and the next one before submitting your code 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, aiven the transition date $\left(s, c, R\left(s,c\right), s'\right)$.

**Reminder:** You should implement this function locally first. You can read through the next tab to understand the context in which this function is called

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

```
def tabular_q_learning(q_func, current_state_1, current_state_2, action
                       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 descri
        action_index (int): index of the current action
        object_index (int): index of the current object
        reward (float): the immediate reward the agent recieves from pl
        next_state_1, next_state_2 (int, int): two indices describing t
        terminal (bool): True if this epsiode is over

    Returns:
```

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

Correct

Generating Speech Output

```
def tabular_q_learning(q_func, current_state_1, current_state_2, action_index,
                       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 the
        action_index (int): index of the current action
        object_index (int): index of the current object
        reward (float): the immediate reward the agent recieves from playing cu
        next_state_1, next_state_2 (int, int): two indices describing the next
        terminal (bool): True if this epsiode is over

    Returns:
        None
    """
    if terminal:
        maxq_next_state = 0
    else:
        q_values_next_state = q_func[next_state_1, next_state_2, :, :]
        maxq_next_state = np.max(q_values_next_state)

    q_value = q_func[current_state_1, current_state_2, action_index,
                     object_index]
    q_func[current_state_1, current_state_2, action_index, object_index] = (
        1 - ALPHA) * q_value + ALPHA * (reward + GAMMA * maxq_next_state)
```

## Test results

**See full output**

**CORRECT**

**See full output**

Submit You have used 11 of 25 attempts

Answers are displayed within the problem

Generating Speech Output

## Epsilon-greedy exploration

1.0/1 point (graded)

Note that the Q-learning algorithm does not specify how we should interact in the world so as to learn quickly. It merely updates the values based on the experience collected. If we explore randomly, i.e., always select actions at random, we would most likely not get anywhere. A better option is to exploit what we have already learned, as summarized by current Q-values. We can always act greedily with respect to the current estimates, i.e., take an action $\pi\left(s\right) = \arg\max_{c \in C} Q\left(s, c\right)$. Of course, early on, these are not necessarily very good actions. For this reason, a typical exploration strategy is to follow a so-called $\varepsilon$-greedy policy: with probability $\varepsilon$ take a random action out of $C$ with probability $1 - \varepsilon$ follow $\pi\left(s\right) = \arg\max_{c \in C} Q\left(s, c\right)$. The value of $\varepsilon$ here balances exploration vs exploitation. A large value of $\varepsilon$ means exploring more (randomly), not using much of what we have learned. A small $\varepsilon$, on the other hand, will generate experience consistent with the current estimates of Q-values.

Now you will write a function `epsilon_greedy` that implements the $\varepsilon$-greedy exploration policy using the current Q-function.

**Reminder:** You should implement this function locally first. You can read through the next tab to understand the context in which this function is called

**Available Functions:** You have access to the NumPy python library as `np`. Your code should also use 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 poli

    Args:
        state_1, state_2 (int, int): two indices describing the current
        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
    """
    action_index, object_index = None, None
    if np.random.random() < epsilon:
        action_index, object_index = np.random.randint(0, NUM_ACTIONS),
    else:
```

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

Generating Speech Output

Correct

```
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 state
        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)
        object_index = np.random.randint(NUM_OBJECTS)
    else:
        q_values = q_func[state_1, state_2, :, :]
        (action_index,
         object_index) = np.unravel_index(np.argmax(q_values, axis=None),
                                          q_values.shape)

    return (action_index, object_index)
```

## Test results

See full output

**CORRECT**

See full output

Submit You have used 4 of 25 attempts

Answers are displayed within the problem

Generating Speech Output

**Topic:** Unit 5 Reinforcement Learning (2 weeks) :Project 5: Text-Based Game / 3. Q-learning Algorithm

Hide Discussion

Add a Post

Show all posts by recent activity

💬 Test file for tabular_q_learning() and epsilon_greedy()
I've added a test file here: https://github.com/Praful/MITx-6.86x-version1/tree/master/tests/p... 2
Pinned Community TA

? What happens when episode ends?
When terminal is True, I understood that the rewards are inmediately equal to zero. Q_value ... 1 new

☑ How to check these functions?
Is there any test function in the code? 3

? help staff:tabular_q_learning gets right answer for terminal= true,but can not find why terminal= false is wrong
for terminal = false, sequence is right, q_func is wrong,thanks. q_func = current_q + ALPHA * (... 5

? Staff-Info
Hello, I am doing wrong, despite the code seems correct. For the Q function, I update the cur... 2

☑ Epsilon-greedy exploration
My understanding of exploration is action_index = np.random.randint(0, NUM_ACTIONS-1) o... 4 new

☑ What is in the Q_func?
Is the Q function set up as current room (current state1), current quest(current state 2), actio... 7

☑ How to choose "randomly" the action and object index that matches the grader
To generate the object and action indexes in the the Epsilon-greedy exploration I'm using: np... 2

? What is terminal's role in tabular_q_learning function.
I am not sure whether I understand it correctly or not but ,for me, I understand that if we fou... 2

☑ Exploitation in Epsilon Greedy function
My `epsilon_greedy()` function has code for both exploration and exploitation. The explorati... 1 new

☑ Stuck on Single Step Update, terminal=True.
I can't seem to get the update right when terminal=True, and I don't know why. I have a condi... 4

Generating Speech Output

Generating Speech Output