Q-learning – Reinforcement learning

Relevant book pages are saved in dropbox RLbook2020-153-154.pdf

Weekly task:

simple tabular Q-Learning, a reinforcement learning method. This includes:

- Implementing the Q-Learning algorithm including an interface for agents
- Implementing some property-based tests for Q-Learning
- Implementing a simple agent example
- Testing the agent
- Creating a program that trains the agent and prints the resulting policy

Notes:

What is Q-learning

Q-learning is atype of reinforcement learning. With an AI agent operating in an environment with states and rewards (inputs)
actions (outputs)

Q-learning involves model-free environments:

- The AI agent is nok seeking to learn about an underlying mathematical model, instead the AI agent attempts to construct an optimal policy directly by interacting with the environment.

Q-learning uses a trial-and-error based approach. The AI agent repeatedly tries to solve the problem using varied approaches and continuously updates its policy as it learns more and more about its environment.

Characteristics of Q-Learning models:

An input and output system, rewards (can be positive or negative), an environment, Markov decision processes and training and inference.

The number of possible states is finite. AI agent will always be in one of a fixed number of possible situations.

The number of possible actions is finite. The AI agent always needs to choose from among a fixed number of possible actions.

Cliff walking game:
Don't fall down the cliff. Walk safely from start to end (it doesn't even know where end is)

What are Q-values:

A Q-value indicates the quality of an action *a* in a given state s, represented by function: Q(s,*a*)

Q-value are our current estimates of the sum of future rewards. That is Q-value, estimate how much additional reward we can accumulate through all remaining steps in the current episode if the AI agent is in state *s* and takes action *a*

Q-values therefore increase as the AI agent gets closer and closer to the highest reward.

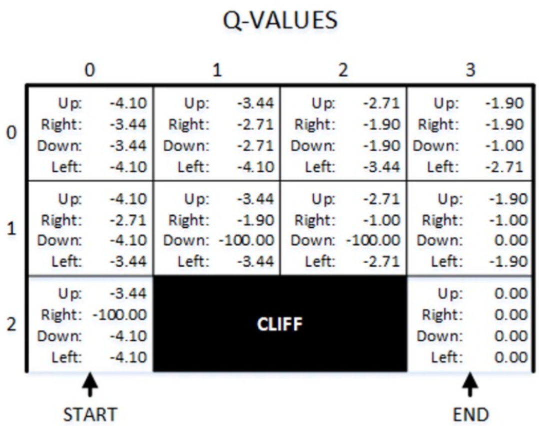
What are Q-tables (the AI agent's policy):

Q-value are stored in a Q-table, which has one row for each possible state, and one column for each possible action.

An optimal Q-table contains values that allow the AI agent to take the best action in any possible state, thus providing the agent wit the optimal path to the highest reward.

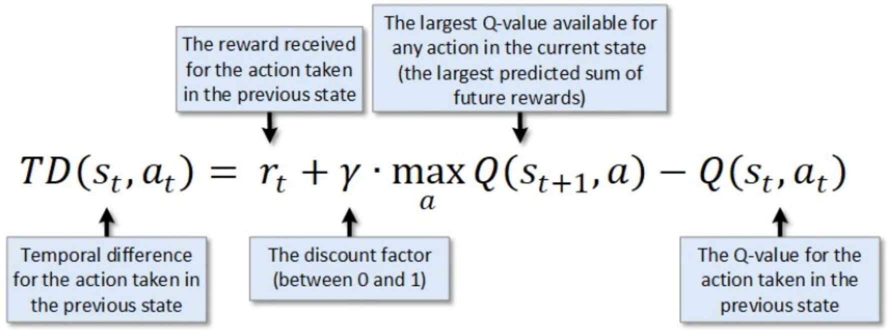
Q-TABLE

Possible States (Row, Column)	Possible Actions			
	Up	Right	Down	Left
0, 0	-4.10	-3.44	-3.44	-4.10
0, 1	-3.44	-2.71	-2.71	-4.10
0, 2	-2.71	-1.90	-1.90	-3.44
0, 3	-1.90	-1.90	-1.00	-2.71
1, 0	-4.10	-2.71	-4.10	-3.44
1, 1	-3.44	-1.90	-100.00	-3.44
1, 2	-2.71	-1.00	-100.00	-2.71
1, 3	-1.90	-1.00	0.00	-1.90
2, 0	-3.44	-100.00	-4.10	-4.10
2, 1	0.00	0.00	0.00	0.00
2, 2	0.00	0.00	0.00	0.00
2, 3	0.00	0.00	0.00	0.00



What are temporal differences:

Temporal differences (TD) provide us with a method of calculating how much the Q-value for the action taken in the previous state should be changed based on what the AI agent has learned about the Q-values for the current states actions. Previous Q-values are therefore updates after each step.



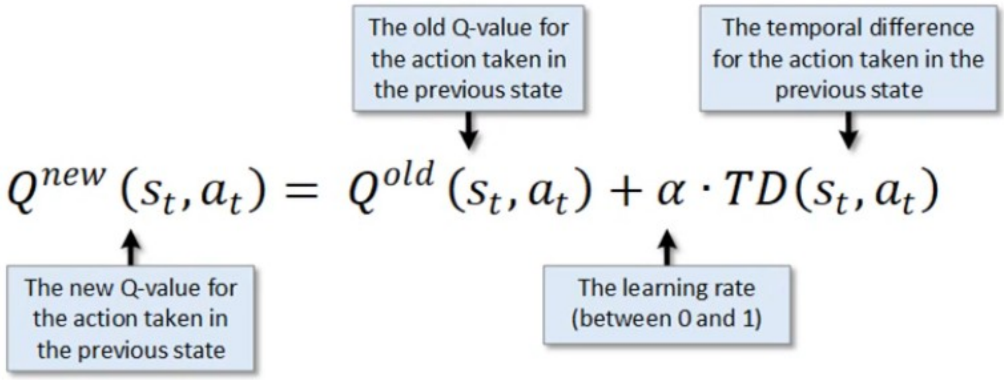
What is the Bellman equation:

“It's like a learning rate dial that controls how quickly the robot updates its understanding. “

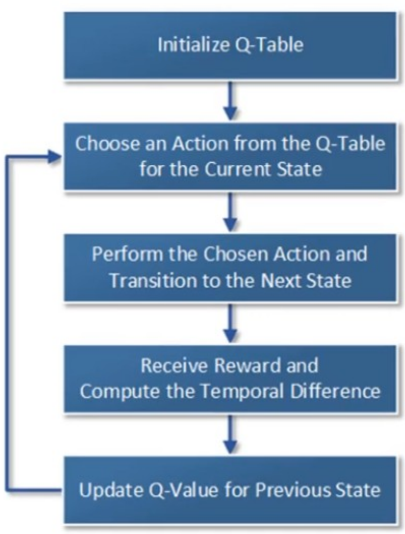
The Bellman Equation tells us what new value to use as the Q-value for the action taken in the previous state.

Relies on a both the old Q-value for the action taken in the previous state, and what has been learned after moving to the next state.

Includes a learning rate parameter that defines how quickly Q-values are adjusted.



The process / 1 episode:



(this is 1 episode)

Epsilon greedy algorithm: our ε-greedy policy (action selection strategy) picks a random action with probability ε. Will take best Q-value 90 % of the time, but 10 % of the time take random action. To force it to explore different possibilities, so in long term it might learn better paths.

Inference mode:

When the Q-learning model is fully trained, it can be used for inference.

In inference mode:

- Q-values are no longer updated
- For any state, the action that the AI agent chooses to take is simply the action with the largest Q-value

The book:

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose A from S using policy derived from Q (e.g., ε-greedy)
        Take action A, observe R, S'
        $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
        $S \leftarrow S'$
    until S is terminal