Q-Learning Example by Hand

Hochschule Fulda
Angewandte Informatik M.Sc.

Reinforcement Learning

von

Adrianus Jonathan Engelbracht, Alexander Hartung, Tabea Runzheimer

December 16, 2024

Contents

1 Q-Learning

In this document Q-Learning is explained for a grid example (2).

1.1 Description

Problem:

$\min_\pi V^\pi = \mathbb{E}_p \left[\sum_{t=0}^{\infty} \gamma^t C_t \right]$

under Markov policy $\pi = \pi(s)$

Q-function: $Q(s,a) \triangleq c(s,a) + \mathbb{E}_{p^a} [V^*(S)]$

Q-learning (value iteration format):

Input: Q-function guess $Q_0 \equiv 0$
for Learning iteration $i := 0 \ldots \mathcal{I}$ **do**
  for State $s_t \in \mathbb{S}$ and $a_t \in \mathbb{A}$ **do**
    Policy update
    $\pi_i(s_t) \leftarrow \arg\min_{a_t \in \mathbb{A}} Q_i(s_t, a_t)$
    Value update $Q_{i+1}(s_t, a_t) \leftarrow \quad c(s_t, a_t) + \gamma \min_a \mathbb{E}_{p^{a_t}} [Q_i(S_{t+1}, a)]$
  end for
end for

return Near-optimal policy $\pi_{\mathcal{I}}$

Alternative update (with a learning rate α_{crit}):
$Q_{i+1} \leftarrow (1 - \alpha_{\text{crit}}) Q_i + \alpha_{\text{crit}} (C_+ + \gamma \min_a \mathbb{E} [Q_i(S_+, a)])$

Figure 1: Qlearning Formula

2 Value Iteration in a Gridworld

2.1 Gridworld Example

We demonstrate the value iteration algorithm on a 3×3 gridworld with the following properties:

- **States:** Each cell (i, j) represents a state.
- **Actions:** At each state, the agent can choose from the following actions: *up, down, left, right, stay.*
- **Costs:**
  - Moving to any adjacent state costs $c = 1$.
  - Staying in the yellow cell has no cost ($c = 0$).
  - Entering a grey cell has a high cost ($c = 10$).
  - Staying in a white cell incurs $c = 2$.

(0,0)	(0,1)	(0,2)
(1,0)	(1,1)	(1,2)
(2,0)	(2,1)	(2,2)

Figure 2: An illustrative grid table for Q-Learning.

2.2 Detailed Explanation of Iteration 1

The tables in chapter 2.3 show the progression of the value function V_i and the optimal policy π_i over iterations $i = 0, 1, 2, \ldots, 4$. In Iteration 1, the value function V and the policy π are updated based on the initial values from Iteration 0. The table 2.3 represents the updated costs associated with each action in every state after this iteration.

2.2.1 Analysis

- States and Actions:
  - The gridworld consists of states labeled from $(0, 0)$ to $(2, 2)$.
  - At each state, the agent can choose one of five actions: *up*, *down*, *left*, *right*, or *stay*.
- Action Costs:
  - Numerical values represent the cost of taking a specific action from a given state.
  - A dash ‘-‘ indicates that the action is invalid from that state (e.g., moving up from the top row).
  - The *stay* action has a uniform cost of ‘2‘ across all states.
  - Certain actions incur higher costs (e.g., ‘10‘), representing undesirable moves or obstacles.
- Policy Updates:
  - The policy π is updated to choose the action with the minimum cost for each state.
  - For example, from state $(1, 1)$, the *up*, *down*, *left*, and *right* actions all have a cost of ‘1‘, so the policy may choose any of these actions as optimal.
  - High-cost actions (cost ‘10‘) are less likely to be chosen unless no other valid actions are available.
- Implications of This Iteration:
  - The values in the table reflect the improved estimates of the cost-to-go from each state based on the initial guess.

- The *stay* action remains relatively costly, encouraging the agent to move towards lower-cost actions.
- States adjacent to high-cost cells ($2,1$ and $2,2$) show higher costs when actions lead into these states, guiding the policy to avoid them when possible.

• Policy Visualization:

- Arrows (not shown in the table) would typically represent the chosen action for each state based on the minimum cost.
- For instance, in state $(1,0)$, the *up* action has the lowest cost ('1'), so the policy would direct the agent to move *up* from this state.

Description of the Value Update-Function

$$Q_{i+1}(s_t, a_t) \leftarrow c(s_t, a_t) + \gamma \min_a \mathbb{E}_{p_{a_t}} \left[Q_i(S_{t+1}, a)\right].$$

$c(s_t, a_t) \rightarrow$ describe the current state

$\gamma \rightarrow$ The discount factor. It determines the weight given to future rewards relative to immediate rewards.

$\min_a \mathbb{E}_{p_{a_t}} \rightarrow$ Represents a risk-averse or conservative approach to estimating the future value of the state-action pair.

$\left[Q_i(S_{t+1}, a)\right] \rightarrow$ Represents the Q-value at iteration i for the next_state-action pair

Example Calculation of the Value Update-Function

Calculating the stat (0,0) with action=down in the 1st iteration

$c(s_t, a_t) = 1$ [cost to move from (0,0) to (1,0)]

$\gamma = 1$

$\mathbb{E}_{p_{a_t}} = 1$ (Grid world example is deterministic)

$\left[Q_i(S_{t+1}, a)\right] =$ 0,0,0,0 [All States for the next State (1,0)]

$Q_{i+1}(s_t, a_t) \leftarrow 1 + 1 * min(0, 0, 0, 0)$

$Q_{i+1}(s_t, a_t) = 1$

2.3 Full Iterations Example

0. Iteration

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	0	0	0	0	0	0
down	0	0	0	0	0	0	-	-	-
left	-	0	0	-	0	0	-	0	0
right	0	0	-	0	0	-	0	0	-
stay	0	0	0	0	0	0	0	0	0

1. Iteration

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	1	1	1	1	10	10
down	1	10	10	1	1	1	-	-	-
left	-	1	1	-	1	10	-	1	1
right	1	1	-	10	10	-	1	1	-
stay	0	2	2	2	2	2	2	2	2

2. Iteration

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	1	2	2	2	11	11
down	2	11	11	2	2	2	-	-	-
left	-	1	2	-	2	11	-	2	2
right	2	2	-	11	11	-	2	2	-
stay	0	3	3	3	3	3	3	3	3

3. Iteration

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	1	2	3	2	12	12
down	2	12	12	3	3	3	-	-	-
left	-	1	2	-	2	12	-	3	3
right	2	3	-	12	12	-	3	3	-
stay	0	3	4	3	4	4	4	4	4

4. Iteration

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	1	2	3	2	12	13
down	2	12	13	3	4	4	-	-	-
left	-	1	2	-	2	12	-	3	4
right	2	3	-	12	13	-	4	4	-
stay	0	3	4	3	4	5	4	5	5

5. Iteration

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	1	2	3	2	12	13
down	2	12	13	3	4	5	-	-	-
left	-	1	2	-	2	12	-	3	4
right	2	3	-	12	13	-	4	5	-
stay	0	3	4	3	4	5	4	5	6

6. Iteration

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	1	2	3	2	12	13
down	2	12	13	3	4	5	-	-	-
left	-	1	2	-	2	12	-	3	4
right	2	3	-	12	13	-	4	5	-
stay	0	3	4	3	4	5	4	5	6

7. Iteration

The cells with the **red** color describe the optimal policy.

	(0,0)	(0,1)	(0,2)	(1,0)	(1,1)	(1,2)	(2,0)	(2,1)	(2,2)
up	-	-	-	**1**	**2**	**3**	**2**	12	13
down	2	12	13	3	4	5	-	-	-
left	-	**1**	**2**	-	2	12	-	**3**	**4**
right	2	3	-	12	13	-	4	5	-
stay	**0**	3	4	3	4	5	4	5	6