Artificial Neural Networks (Gerstner). Solutions for week 3

Variants of TD-learning methods and eligibility traces

Exercise 0. Recap: SARSA on Linear Track[1]

Please make sure that you have finished **'Exercise 2. SARSA algorithm'** from last week (week 2) before you continue. Exercise 2 from last week is important for the lecture today at 14h15. The learning outcomes of the SARSA exercise are (i) to be at ease using the SARSA update formula and (ii) to understand why learning with SARSA takes many episodes.

Exercise 1. Expected SARSA and a new variant of TD learning[2]

We have seen the algorithm 'Expected Sarsa' and 'TD-algorithm in the narrow sense'. Suppose you invent a new algorithm that keeps tables of both Q-values and V-values. To do so consider the following:

a. Rewrite the updated step of 'Expected Sarsa' using both Q-values and V-values in the update rule.

b. Write down the full pseudo-algorithm.

   Hint: (i) keep track of the sequence of update steps of Q-values and V-values in your new algorithm. (ii) At which point in the sequence of online steps through the graph can you update $Q(s, a)$ and how far do you have to look backward?

c. Sketch the 'back-up-diagram' of your algorithm and compare it to that of SARSA. What are the costs and benefits of your new algorithm compared to SARSA?

Solution:

a. The update rule is given by

$$
\begin{aligned}
Q(s_t, a_t) &\leftarrow (1-\eta)\cdot Q(s_t, a_t) + \eta \cdot \Big(r_t + \gamma V(s_{t+1})\Big) \\
V(s_t) &\leftarrow (1-\eta)\cdot V(s_t) + \eta \cdot \Big(r_t + \gamma \sum_{a'} \pi(a'|s_{t+1}) Q(s_{t+1}, a')\Big)
\end{aligned}
$$

b. Pseudo-code:

```
• Initialize $Q(s,a)$ and $V(s)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$ with $V(term) = Q(term, .) = 0$
• Repeat (for each episode):
    – Initialize $s$
    – Choose $a$ from $s$ using policy derived from $Q$
    – Repeat (for each step of episode):
        ∗ Take action $a$, observe $r$ and $s'$
        ∗ $Q(s,a) \leftarrow (1-\eta)\cdot Q(s,a) + \eta \cdot \Big(r + \gamma V(s')\Big)$
        ∗ $V(s) \leftarrow (1-\eta)\cdot V(s) + \eta \cdot \Big(r + \gamma \sum_{a'} \pi(a'|s') Q(s', a')\Big)$
        ∗ Choose $a'$ from $s'$ using policy derived from $Q$
        ∗ $s \leftarrow s'$ and $a \leftarrow a'$
    – until $s$ is terminal.
```

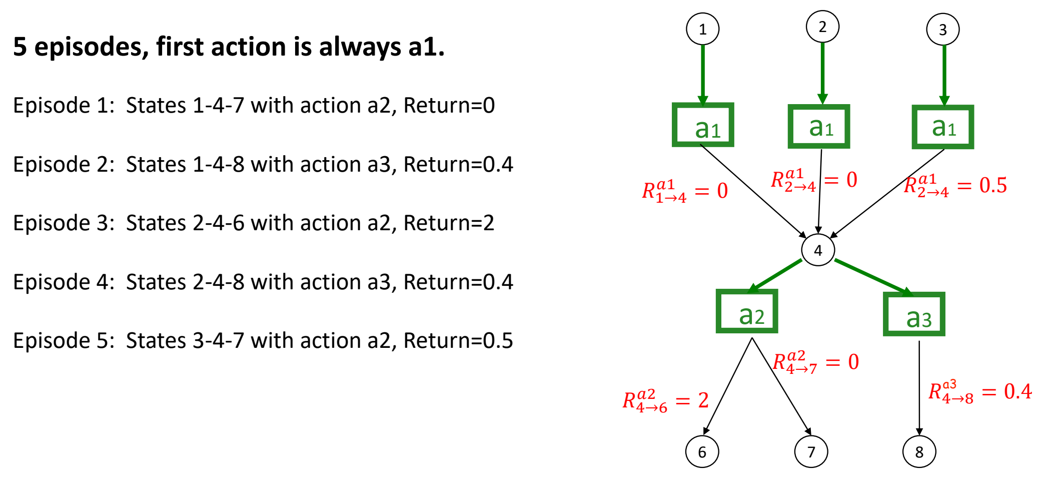
[1]The result of Exercise 0 will be used in the second lecture of week 3.
[2]The result of Exercise 1 will be used in the second lecture of week 3.

c. Cost: You need to keep (and update) two tables instead of a single one.

   Benefit: you can update the Q-value of (s, a) immediately at state s' (without playing round with potential next actions) because you already updated the V-value last time you visited the state s'. Hence the back-up diagram is shorter than for SARSA (you only need to keep (s, a, r, s') in the buffer whereas for SARSA you need to keep s, a, r, s', a').

Exercise 2. Monte Carlo versus expected SARSA[3]

We compare two algorithms: The Batch-Monte-Carlo algorithm (around slide 47), and the Online-Expected-SARSA (around slide 19). The aim of the exercise is to understand why bootstrap algorithms (i.e., TD algorithms) perform, under some conditions, more efficiently than a naive estimation of Q-Values via Monte-Carlo algorithms. We set the discount factor to $\gamma = 1$ and run 5 episodes in a given environment:


5 episodes, first action is always a1.

Episode 1: States 1-4-7 with action a2, Return=0

Episode 2: States 1-4-8 with action a3, Return=0.4

Episode 3: States 2-4-6 with action a2, Return=2

Episode 4: States 2-4-8 with action a3, Return=0.4

Episode 5: States 3-4-7 with action a2, Return=0.5

Each episode starts in one of the states 1 or 2 or 3 with action a_1. In state 4 there is a choice between actions a_2 and a_3 which are taken with equal probability $\pi = 0.5$. The transition sequence and total return for each of the 5 episodes is given in the figure above. All rewards are deterministic and only depend on the transition (s, a, s').

a. Calculate the Q-values in state 1, 2, 3, and 4 using Batch-Monte-Carlo (i.e., average total returns from each starting state).

b. Calculate the Q-values in state 1, 2, 3, and 4 using Online-Expected SARSA. For a given Q-value $Q(s, a)$, use $\eta_1 = 1$ the FIRST TIME you update this value and $\eta \in [0, 0.5]$ for all later update steps.

   Hint: You can neglect terms of order η^2.

c. You can choose the initial state for episode 6. Which initial state looks best in case a (Batch-Monte-Carlo)? Which initial state looks best in case b (Online-Expected SARSA)?

   Where does the difference come from? Which solution is closer to the exact Q-values under the $\pi = 0.5$ policy in state 4? Why?

d. What happens qualitatively if the order of episodes 1 and 2 were switched? What would be the value of $Q(s = 1, a_1)$ in this case? How would the other Q-values change?

Solution:

a. Batch-Monte-Carlo:

[3]The result of Exercise 2 will be used in the second lecture of week 3.

- $Q(1,a_1) = \frac{1}{2}(0+0.4) = 0.2$
- $Q(2,a_1) = \frac{1}{2}(2+0.4) = 1.2$
- $Q(3,a_1) = 0.5$
- $Q(4,a_2) = \frac{1}{3}(0+2+0) = 2/3$
- $Q(4,a_3) = \frac{1}{2}(0.4+0.4) = 0.4$

b. Online Expected SARSA:

Episode 1:

- $Q(1,a_1) \leftarrow 0 \cdot Q(1,a_1) + 1 \cdot \left(0 + \frac{Q(4,a_2)+Q(4,a_3)}{2}\right) = 0$
- $Q(4,a_2) \leftarrow 0 \cdot Q(4,a_2) + 1 \cdot 0 = 0$

Episode 2:

- $Q(1,a_1) \leftarrow (1-\eta) \cdot Q(1,a_1) + \eta \cdot \left(0 + \frac{Q(4,a_2)+Q(4,a_3)}{2}\right) = 0$
- $Q(4,a_3) \leftarrow 0 \cdot Q(4,a_3) + 1 \cdot 0.4 = 0.4$

Episode 3:

- $Q(2,a_1) \leftarrow 0 \cdot Q(2,a_1) + 1 \cdot \left(0 + \frac{Q(4,a_2)+Q(4,a_3)}{2}\right) = 0.2$
- $Q(4,a_2) \leftarrow (1-\eta) \cdot Q(4,a_2) + \eta \cdot 2 = 2\eta$

Episode 4:

- $Q(2,a_1) \leftarrow (1-\eta) \cdot Q(2,a_1) + \eta \cdot \left(0 + \frac{Q(4,a_2)+Q(4,a_3)}{2}\right) = 0.2 + \eta^2 \approx 0.2$
- $Q(4,a_3) \leftarrow (1-\eta) \cdot Q(4,a_3) + \eta \cdot 0.4 = 0.4$

Episode 5:

- $Q(3,a_1) \leftarrow 0 \cdot Q(3,a_1) + 1 \cdot \left(0.5 + \frac{Q(4,a_2)+Q(4,a_3)}{2}\right) = 0.7 + \eta$
- $Q(4,a_2) \leftarrow (1-\eta) \cdot Q(4,a_2) + \eta \cdot 0 = 2\eta - 2\eta^2 \approx 2\eta.$

As a result:

$$Q(1,a_1) = 0; \quad Q(2,a_1) = 0.2; \quad Q(3,a_1) = 0.7 + \eta; \quad Q(4,a_2) = 2\eta; \quad Q(4,a_3) = 0.4$$

c. With Batch MC state 2 looks best, but with online expected SARSA state 3 looks best.

Also expected SARSA is closer to the exact solution for Q-values:

$$Q(1,a_1) = 0.7; \quad Q(2,a_1) = 0.7; \quad Q(3,a_1) = 1.2; \quad Q(4,a_2) = 1; \quad Q(4,a_3) = 0.4$$

Note that with a choice $\eta = 0.5$ or $\eta = 0.3$, the difference between the Expected SARSA and the excat solution is really small.

The reason is that for the update of Q-values for states 2 and 3, SARSA uses ALL previous trials that involve state 4, and hence uses better statistics for 4 than the Batch MC algorithm.

d. then the return of 0.4 of the new first episode would influence the Q-value of state 4 used to update the Q-value of state 1 in the second episode. The other Q-values would not change Hence, we would have

$$Q(1,a_1) = 0.2\eta; \quad Q(2,a_1) = 0.2; \quad Q(3,a_1) = 0.7 + \eta; \quad Q(4,a_2) = 2\eta; \quad Q(4,a_3) = 0.4$$

Conclusion: TD-algorithms make better use of the data since return information from intermediate states is shared!

Exercise 3. Eligibility traces

In week 2 in exercise 2, we applied the SARSA algorithm to the case of a linear track with actions 'up' and 'down'. We found that it takes a long time to propagate the reward information through state space. The eligibility trace is introduced as a solution to this problem.

Reconsider the linear maze from Figure 1 in exercise 2, but include an eligibility trace: for each state s and action a, a memory $e(s,a)$ is stored. At each time step, all the memories are reduced by a factor $\lambda < 1$: $e(s,a) = \lambda e(s,a)$, except for the memory corresponding to the current state s^* and action a^*, which is incremented:

$$e(s^*, a^*) = \lambda e(s^*, a^*) + 1\,. \tag{1}$$

Now, unlike the case without eligibility trace, all Q-values are updated at each time step according to the rule

$$\forall (s,a) \quad \Delta Q(s,a) = \eta \left[r - \left(Q(s^*, a^*) - Q(s', a') \right) \right] e(s,a)\,. \tag{2}$$

where s^*, a^* are the current state and action, and s', a' are the immediately following state and action.

We want to check whether the information about the reward propagates more rapidly. To find out, assume that the agent goes straight down in the first trial. In the second trial it uses a greedy policy. Calculate the Q-values after two complete trials and report the result.

Hint: Reset the eligibility trace to zero at the beginning of each trial.

Solution:

The table below shows the evolution of the Q values for each relevant state action pair during the first 2 trials, starting at the first step when there is a non-zero update $\Delta = \eta\,[r - (Q(s^*, a^*) - Q(s', a'))]$. We assume that the agent goes straight down in the first trial, that it always picks the best action, and that the eligibility traces are reset when the agent picks the reward, and the agent is put back to the starting position.

trial	transition		(s, a_1)	(s', a_1)	(s'', a_1)	Δ
1	$s'' \to s'''$	Q	0	0	0	η
		e	λ^2	λ	1	$-$
2	$s \to s'$	Q	$\eta\lambda^2$	$\eta\lambda$	η	$\eta^2(\lambda - \lambda^2)$
		e	1	0	0	$-$
2	$s' \to s''$	Q	$\eta\lambda^2 + \eta^2(\lambda - \lambda^2)$	$\eta\lambda$	η	$\eta^2(1 - \lambda)$
		e	λ	1	0	$-$
2	$s'' \to s'''$	Q	$\eta\lambda^2 + 2\eta^2\lambda - 2\eta^2\lambda^2$	$\eta\lambda + \eta^2(1 - \lambda)$	η	$\eta - \eta^2$
		e	λ^2	λ	1	$-$
3	$s \to s'$	Q	$2\eta\lambda^2 + 2\eta^2\lambda - 3\eta^2\lambda^2$	$2\eta\lambda + \eta^2 - 2\eta^2\lambda$	$2\eta - \eta^2$	$\cdots$
		e	1	0	0	$-$

$\cdots$

Although the Q-values for s'' are the same as without the eligibility trace (see exercise from last week), the Q-values for s and s' already start to approach their asymptotical value (i. e. 1) in the first trial.

Exercise 4. Eligibility traces in continuous space

The left part of Figure 1 shows a different representation of last week's "linear track" exercise: the vertical divisions represent different states, and the two column correspond to the two possible actions available to the agent: go up or down. Each square represents a possible state-action combination, and thus a Q-value. (Note that the uppermost "up" action and the lowermost "down" action should be "greyed out": they are impossible. But this is not relevant to the rest of this exercise.)

Suppose now that the agent moves in a continuous 1-dimensional space $0 \leq x \leq 1$, with the target located at $x = 0$. Separate this state space into n equal bins of width $\Delta x = 1/n$. In each time step, the agent moves by one bin. Vary the discretization by varying n: $n = 4, 8, 16 \ldots$

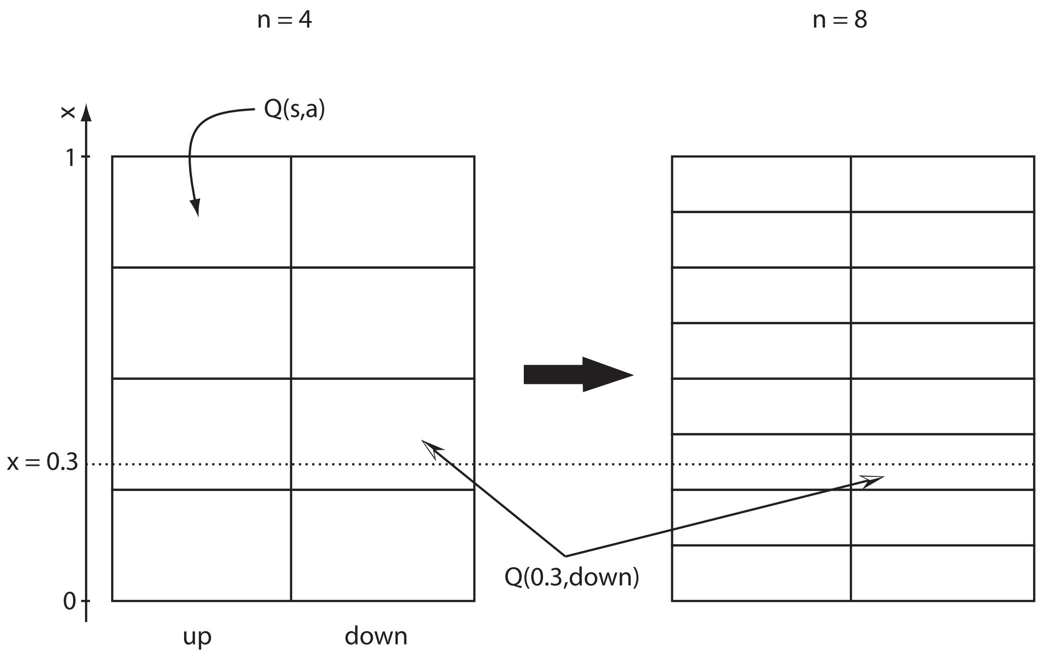


Figure 1: Figure for Exercise 4

a. Suppose that one action (such as move down) corresponds to one time step Δt in 'real time'. How should we rescale the parameter Δt, so that the speed $v = \Delta x / \Delta t$ remains constant when we change the discretization?

b. We use an eligibility trace with decay parameter λ. How should we rescale λ, in order that the "speed of information propagation" in SARSA(λ) remains constant?

   Hint: Consider the Q-value at a fixed x, for example at $x = 0.5$, after 2 complete learning trials.

Solution:

a. If $\Delta x / \Delta t$ has to remain constant, then Δt should vary like Δx, i.e., $\Delta t \propto 1/n$.

b. A point "sitting" at x is $x/\Delta x = x \cdot n$ steps away from the reward (assuming we always choose the "down" action). As we have seen in exercise 2, on the first trial, the Q-value of a state d steps from the trial is updated proportional to λ^d. Thus, if we want the update to stay constant under rescaling, we need

$$\Delta Q(s,a) \propto \lambda^d = \lambda^{x \cdot n} = cst$$

   This holds if we use λ with the following "rescaling": $\lambda = \tilde{\lambda}^{\frac{1}{n}}$ where $\tilde{\lambda}$ is constant.

Exercise 5. 3-step SARSA algorithm

In class we have discussed the SARSA algorithm and shown that, after convergence, the resulting Q-values solve (in expectation) the Bellman equation for *neighboring* states (Variant A in the slides, fixed/non-fluctuating Q-values after convergence). Your friend claims that a 3-step SARSA for

$$\Delta Q(s_t, a_t) = \eta \left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 Q(s_{t+3}, a_{t+3}) - Q(s_t, a_t) \right] , \tag{3}$$

should work just as well.

To simplify the analysis, we assume that the environment has no loops (i.e., the graph is directed) so that we can consider $\gamma = 1$.

a. Assume that the 3-step SARSA algorithm converges in expectation. Proceed as during the lecture to show that $\mathbb{E}\big[\Delta Q(s_t, a_t)\big] = 0$ implies

$$Q(s_t, a_t) = \sum_{s'} P^{a_t}_{s_t \to s'} \left[R^{a_t}_{s_t \to s'} + \sum_{a'} \pi(s', a') B_1(s', a') \right]$$

where

$$B_1(s', a') = \sum_{s''} P^{a'}_{s' \to s''} \left[R^{a'}_{s' \to s''} + \sum_{a''} \pi(s'', a'') B_2(s'', a'') \right]$$

$$B_2(s'', a'') = \sum_{s'''} P^{a''}_{s'' \to s'''} \left[R^{a''}_{s'' \to s'''} + \sum_{a'''} \pi(s''', a''') Q(s''', a''') \right]$$

b. Show the equivalence of the previous equation to the 1-step Bellman equation.

Solution:

a. $\mathbb{E}\big[\Delta Q(s_t, a_t)\big] = 0$ implies

$$Q(s_t, a_t) = \sum_{s'} P^{a_t}_{s_t \to s'} R^{a_t}_{s_t \to s'} + \langle \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 Q(s_{t+3}, a_{t+3}) \rangle_{s_{t+1} = s'} \tag{4}$$

$$= \sum_{s'} P^{a_t}_{s_t \to s'} \left(R^{a_t}_{s_t \to s'} + \gamma \sum_{a'} \pi(s', a') \left(\sum_{s''} P^{a'}_{s' \to s''} R^{a'}_{s' \to s''} + \langle \gamma r_{t+2} + \gamma^2 Q(s_{t+3}, a_{t+3}) \rangle_{s_{t+2} = s''} \right) \right) \tag{5}$$

where we used the fact that the probability of action a' given state s' is determined by the policy $\pi(s', a')$. Continuing along the same lines we find the claimed result.

b. If we expand the 1-step Bellman equation by iteratively inserting the right-hand-side of the Bellman equation, we arrive at the 3 equations in a.

Exercise 6. Computer exercises: Environment 1 (part 2)[1]

Complete the computer exercise for environment 1.

[1]Start this exercise in the second exercise session of week 3.