EL2805 Reinforcement Learning

Homework 2

November 24, 2020

Department of Decision and Control Systems
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology

Instructions (read carefully):

- Answer the questions of Parts 1 and 2.
- Work in groups of 2 persons.
- **Both** students in the group should upload their scanned report as a .pdf-file to Canvas before December 13, 23:59. The deadline is strict. Please mark your answers directly on this document, and **append** hand-written or typed notes justifying your answers. Reports without justification will not be graded.

Good luck!

1 Part 1. Q-learning and SARSA

Consider a discounted MDP with $\mathcal{S} = \{\text{A, B, C}\}$ and $\mathcal{A} = \{a, b, c\}$. We plan to use either the Q-learning or the SARSA algorithm in order to learn to control the system. We initialize the estimated Q-function as all zeros – that is:

$$Q^{(0)} = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array} .$$

The observed trajectory is as follows (for these transitions, we are imposed a policy):

$$(?, ?, ?); (A, ?, ?); (B, a, 100); (A, b, 60); (B, c, 70); (C, b, 40); (A, a, 20); (C, c, \ldots)$$

where each triplet represents the state, the selected action, and the corresponding reward. Some of the information has been corrupted (marked with question marks) in the above sequence.

a) Before the information became corrupt, we ran the Q-learning algorithm and obtained that

$$Q^{(2)} = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \begin{bmatrix} 11 & 0 & 0 \\ 0 & 0 & 60 \\ 0 & 0 & 0 \end{bmatrix} \end{array} .$$

The discount factor was $\lambda = 0.5$ and the learning rate was fixed to $\alpha = 0.1$. Can you infer what the corrupt information was (i.e., the first state, the first and second selected actions, and the first and second observed rewards? **Answer**:

$(\underline{\text{B}}, \underline{\text{c}}, \underline{600}); (A, \underline{\text{a}}, \underline{80}); (B, a, 100); (A, b, 60); (B, c, 70); (C, b, 40); (A, a, 20); (C, c, \ldots)$

b) Provide the updated Q-values, using the Q-learning algorithm, at the 7th iteration. Use the same values for λ and α as in a). **Answer**:

$$Q^{(7)} = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \begin{bmatrix} 12.1275 & 9 & 0 \\ 10.55 & 0 & 61 \\ 0 & 4.55 & 0 \end{bmatrix} \end{array} .$$

c) What is the greedy policy at the 7th iteration? $\pi(A) = \underline{\text{a}}, \pi(B) = \underline{\text{c}}, \pi(C) = \underline{\text{b}}$.

d) Provide the updated Q-values at the 7th iteration using the SARSA algorithm (initialized with $Q^{(0)}$ as all zeros). Take the first two (state, action, reward)-triplets as those given in your answer to a). Let the discount factor be $\lambda = 0.5$ and the learning rate fixed to $\alpha = 0.1$. **Answer**:

$$Q^{(7)} = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \begin{bmatrix} 9.2 & 9 & 0 \\ 10 & 0 & 61 \\ 0 & 4.4 & 0 \end{bmatrix} \end{array} .$$

e) What is the greedy policy at the 7th iteration? $\pi(A) = \underline{\text{a}}, \pi(B) = \underline{\text{c}}, \pi(C) = \underline{\text{b}}$.

2 Part 2: policy gradient and function approximation

Policy gradients. We consider an episodic RL problem with finite state-space $\mathcal{S}$ and action space $\mathcal{A} = \{1, \ldots, n+1\}$. For all states s, let $f(s)$ be a real valued function in $[1, 2]$. We parameterize the policy using parameter vector $\theta = (\theta_1, \ldots, \theta_n) \in [0, 1]^n$ according to the following recursion: For $i \in \{1, \ldots, n\}$, initialize $i = 1$ and draw independent random variable Z_i uniformly from $[0, f(s)]$. If $Z_i \leq \theta_i$, choose action $a = i$, otherwise, set $i \leftarrow i+1$ and repeat. At the last step of the recursion, if $Z_n > \theta_n$, choose $a = n + 1$.

a) Compute in state s, the probability $\pi_\theta(s, i)$ of choosing action i. **Answer**:

$$\pi_\theta(s, 1) = \frac{\theta_1}{f(s)}$$

$$\pi_\theta(s, i) = \left[\prod_{k=1}^{i-1} \left(1 - \frac{\theta_k}{f(s)}\right)\right] \frac{\theta_i}{f(s)} \qquad \text{for } i \in \{2, \ldots, n\}$$

$$\pi_\theta(s, n + 1) = \prod_{k=1}^{n} \left(1 - \frac{\theta_k}{f(s)}\right)$$

b) What is the Monte-Carlo REINFORCE update of θ upon observing an episode $\tau = (s_1, a_1, r_1, \ldots, s_T, a_T, r_T)$? Provide explicit formulas using the function f, θ and τ only.

$$\frac{\partial \ln \pi_\theta(s, i)}{\partial \theta_i} = \frac{1}{\theta_i}$$

$$\frac{\partial \ln \pi_\theta(s, i)}{\partial \theta_k} = \frac{1}{\theta_k - f(s)} \qquad \text{for } k < i$$

$$\frac{\partial \ln \pi_\theta(s, i)}{\partial \theta_k} = 0 \qquad \text{for } k > i$$

On-policy control with function approximation. Consider a discounted RL problem, that we wish to solve using approximations of the (state, action) value function (i.e., parametrized by vector θ).

c) We observe the transition (s_t, a_t, r_t, s_{t+1}). State the Q update in the Q-learning algorithm with function approximation. Why is it a semi-gradient algorithm? **Answer:**

The parameter updates for Q-learning with function approximation are given as

$$\theta \leftarrow \theta + \alpha(r + \lambda \underset{b}{max} Q_\theta(s', b) - Q_\theta(s, a)) \cdot \nabla_\theta Q_\theta(s, a)$$

This equation is referred to as a semi-gradient method, since it does not consider the derivative of the target (that is $r + \lambda \underset{b}{max} Q_\theta(s', b)$) with respect to θ. The intuition behind this heuristic is to ease convergence of the algorithm by not considering the convergence target during optimization.

d) In the previous updates, the "target" evolves in every step, which could affect the algorithm convergence. What do we mean by target? Can you propose a modification that addresses this problem? **Answer:**

The target is $r + \lambda \underset{b}{max} Q_\theta(s', b)$. By updating θ, $Q_\theta(s', a')$ also changes. One solution is to have a second parameterization for the estimation of $Q_\phi(s', a')$ that is independent of θ and is only updated after multiple updates of $Q_\theta(s, a)$. The update then becomes

$$\theta \leftarrow \theta + \alpha(r + \lambda \underset{b}{max} Q_\phi(s', b) - Q_\theta(s, a)) \cdot \nabla_\theta Q_\theta(s, a)$$

$$\phi \leftarrow (1 - \eta)\phi + \eta\theta \quad \text{at iteration } modulo(i, c) = 0, \quad \eta \in (0, 1)$$

Reinforcement Learning Homework 2 - Annotations

Damian Valle, Martin Schuck

December 10, 2020

Q-learning and SARSA

a) Corrupted information

It is known that $S_2 = A$ and, because only two values of the Q-function at time step 2 are updated, we can conclude that $S_1 = B$.

$$Q^{(1)}(B, c) = Q^{(0)}(B, c) + \alpha[R_1 + \lambda \max_{a'} Q^{(0)}(A, a') - Q^{(0)}(B, c)]$$
$$Q^{(1)}(B, c) = 0 + 0.1[R_1] = 60 \longrightarrow R_1 = 600$$

$$Q^{(2)}(A, a) = Q^{(1)}(A, a) + \alpha[R_2 + \lambda \max_{a'} Q^{(1)}(B, a') - Q^{(1)}(A, a)]$$
$$Q^{(2)}(A, a) = 0 + 0.1[R_1 + 0.5 \cdot 60 - 0] = 110 \longrightarrow R_2 = 80$$

We have now determined all the corrupted information:

$$(B, c, 600); (A, a, 80); (B, a, 100); ...$$

b) Q-learning algorithm iterations

Step 3:

$$(B, a, 100); s' = A$$
$$Q(B, a) = 0 + 0.1[100 + 0.5 \cdot 11 - 0] = 10.55$$

Q	a	b	c
A	11	0	0
B	10.55	0	60
C	0	0	0

Step 4:

$$(A, b, 60); s' = B$$
$$Q(A, b) = 0 + 0.1[60 + 0.5 \cdot 60 - 0] = 9$$

Q	a	b	c
A	11	9	0
B	10.55	0	60
C	0	0	0

Step 5:

$$(B, c, 70); s' = C$$
$$Q(B, c) = 60 + 0.1[70 + 0.5 \cdot 0 - 60] = 61$$

Q	a	b	c
A	11	9	0
B	10.55	0	61
C	0	0	0

Step 6:

$$(C, b, 40); s' = A$$
$$Q(C, b) = 0 + 0.1[40 + 0.5 \cdot 11 - 0] = 4.55$$

Q	a	b	c
A	11	9	0
B	10.55	0	61
C	0	4.55	0

Step 7:

$$(A, a, 20); s' = C$$
$$Q(A, a) = 11 + 0.1[20 + 0.5 \cdot 4.55 - 11] = 12.1275$$

Q	a	b	c
A	12.1275	9	0
B	10.55	0	61
C	0	4.55	0

c) Q-learning greedy policy

We calculate the greed policy as

$$\pi(s) = \arg\max_a Q(s, a),$$

therefore:

$$\pi(A) = a$$
$$\pi(B) = c$$
$$\pi(C) = b$$

d) SARSA algorithm iterations

The Q-value for a state-action pair is now updated by the following expression:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R(s) + \lambda \cdot Q(s',a') - Q(s,a)].$$

Step 1:

$$(B, c, 600); s' = A; a' = a$$
$$Q(B,c) = 0 + 0.1[600 + 0.5 \cdot 0 - 0] = 60$$

Q	a	b	c
A	0	0	0
B	0	0	60
C	0	0	0

Step 2:

$$(A, a, 80); s' = B; a' = a$$
$$Q(A,a) = 0 + 0.1[80 + 0.5 \cdot 0 - 0] = 8$$

Q	a	b	c
A	8	0	0
B	0	0	60
C	0	0	0

Step 3:

$$(B, a, 100); s' = A; a' = b$$
$$Q(B,a) = 0 + 0.1[100 + 0.5 \cdot 0 - 0] = 10$$

Q	a	b	c
A	8	0	0
B	10	0	60
C	0	0	0

Step 4:

$$(A, b, 60); s' = B; a' = c$$
$$Q(A,b) = 0 + 0.1[60 + 0.5 \cdot 60 - 0] = 9$$

Q	a	b	c
A	8	9	0
B	10	0	60
C	0	0	0

Step 5:

$$(B, c, 70); s' = C; a' = b$$
$$Q(B, c) = 60 + 0.1[70 + 0.5 \cdot 0 - 60] = 61$$

Q	a	b	c
A	8	9	0
B	10	0	61
C	0	0	0

Step 6:

$$(C, b, 40); s' = A; a' = a$$
$$Q(C, b) = 0 + 0.1[40 + 0.5 \cdot 8 - 0] = 4.4$$

Q	a	b	c
A	8	9	0
B	10	0	61
C	0	4.4	0

Step 7:

$$(A, a, 20); s' = C; a' = c$$
$$Q(A, a) = 8 + 0.1[20 + 0.5 \cdot 0 - 8] = 9.2$$

Q	a	b	c
A	9.2	9	0
B	10	0	61
C	0	4.4	0

e) SARSA greedy policy

We calculate the greedy policy as

$$\pi(s) = \arg\max_a Q(s, a),$$

therefore:

$$\pi(A) = a$$
$$\pi(B) = c$$
$$\pi(C) = b$$

Policy gradient and function approximation

Policy gradients

The probabilities of choosing action i are given as

$$\pi_\theta(s,1) = \frac{\theta_1}{f(s)}$$
$$\pi_\theta(s,i) = \left[\prod_{k=1}^{i-1}\left(1-\frac{\theta_k}{f(s)}\right)\right]\frac{\theta_i}{f(s)}$$
$$\pi_\theta(s,n+1) = \prod_{k=1}^{n}\left(1-\frac{\theta_k}{f(s)}\right).$$

REINFORCE θ update

The general update rule for the REINFORCE algorithm is

$$\theta \leftarrow \theta + \alpha\left(\sum_{i\in\tau}\nabla_\theta \ln \pi(s_i,a_i)\right)\left(\sum_{i\in\tau} r_i\right)$$

With the equations from the policy gradients, the derivatives for the parameter update can be computed as

$$\frac{\partial \ln \pi_\theta(s,i)}{\partial\theta_i} = \frac{\partial}{\partial\theta_i}\ln\left\{\left[\prod_{k=1}^{i-1}\left(1-\frac{\theta_k}{f(s)}\right)\right]\frac{\theta_i}{f(s)}\right\} = \frac{\partial}{\partial\theta_i}\left[\sum_{k=1}^{i-1}\ln\left(1-\frac{\theta_k}{f(s)}\right)+\ln\left(\frac{\theta_i}{f(s)}\right)\right] = \frac{1}{\theta_i}.$$

It is easy to see that $\frac{\partial}{\partial\theta_1}\pi_\theta(s,1) = \frac{1}{\theta_1}$ and therefore has the same solution as a general θ_i. Similarly, for $k < i$, the derivative becomes

$$\frac{\partial \ln \pi_\theta(s,i)}{\partial\theta_k} = \frac{\partial}{\partial\theta_k}\ln\left\{\left[\prod_{j=1}^{i-1}\left(1-\frac{\theta_j}{f(s)}\right)\right]\frac{\theta_i}{f(s)}\right\} = \frac{\partial}{\partial\theta_k}\left[\sum_{j=1}^{i-1}\ln\left(1-\frac{\theta_j}{f(s)}\right)+\ln\left(\frac{\theta_i}{f(s)}\right)\right] = \frac{1}{\theta_k - f(s)}.$$

For any $k > i$, there is no corresponding term in the product of $\pi_\theta(s,i)$ and therefore, the derivative becomes

$$\frac{\partial \ln \pi_\theta(s,i)}{\partial\theta_k} = \frac{\partial}{\partial\theta_k}\ln\left\{\left[\prod_{j=1}^{i-1}\left(1-\frac{\theta_j}{f(s)}\right)\right]\frac{\theta_i}{f(s)}\right\} = 0$$

On-policy control with function approximation

The parameter updates for Q-learning with function approximation are given as

$$\theta \leftarrow \theta + \alpha(r + \lambda\, \underset{b}{max} Q_\theta(s',b) - Q_\theta(s,a)) \cdot \nabla_\theta Q_\theta(s,a)$$

This equation is referred to as a semi-gradient method, since it does not consider the derivative of the target (that is $r + \lambda\, \underset{b}{max} Q_\theta(s',b)$) with respect to θ. The intuition behind this heuristic is to ease convergence of the algorithm by not considering the convergence target during optimization. The target is $r + \lambda\, \underset{b}{max} Q_\theta(s',b)$. By updating θ, $Q_\theta(s',a')$ also changes. One solution is to have a second parameterization for the estimation of $Q_\phi(s',a')$ that is independent of θ and is only updated after multiple updates of $Q_\theta(s,a)$. The update then becomes

$$\theta \leftarrow \theta + \alpha(r + \lambda\, \underset{b}{max} Q_\phi(s',b) - Q_\theta(s,a)) \cdot \nabla_\theta Q_\theta(s,a)$$
$$\phi \leftarrow (1-\eta)\phi + \eta\theta \quad \text{at iteration } modulo(i,c) = 0, \;\; \eta \in (0,1)$$