EL2805 Reinforcement Learning - Homework 2

Authors
Lea Keller - `lmjke@kth.se` - 19980209-4889
Jannik Wagner - `wagne@kth.se` - 19971213-1433

1 Part 1. Q-learning and SARSA

a)

Unknown: s_1, a_1, r_1, a_2, r_2. Since $s_2 = A$, we know that $Q^{(2)}(B, c)$ could not have been updated in step 2, and must have been updated in step 1. We can thus deduce that $s_1 = B, a_1 = c$. Furthermore, $Q^{(2)}(A, s)$ must then have been updated in step 2, thus $a_2 = a$.

$$
\begin{aligned}
60 &= Q^{(1)}(B, c) \\
&= Q^{(0)}(B, c) + \alpha(r_1 + \lambda \max_{a'} Q^{(0)}(A, a') - Q^{(0)}(B, c)) \\
&= 0 + \alpha(r_1 + \lambda 0 - 0) \\
&= \alpha r_1 \\
&= \frac{1}{10} r_1 \\
\iff& \\
r_1 &= 600
\end{aligned}
$$

$$
\begin{aligned}
11 &= Q^{(2)}(A, a) \\
&= Q^{(1)}(A, a) + \alpha(r_2 + \lambda \max_{a'} Q^{(1)}(B, a') - Q^{(1)}(A, a)) \\
&= 0 + \alpha(r_2 + \lambda Q^{(1)}(B, c) - 0) \\
&= \frac{1}{10}(r_2 + \frac{1}{2}60) \\
&= \frac{1}{10} r_2 + 3 \\
\iff& \\
r_2 &= 80
\end{aligned}
$$

b)

The calculations are done with the script in file `task1.py`.

$$
\begin{aligned}
Q^1(B,c) &= Q^0(B,c) + \alpha(r_1 + \gamma \max_x Q^0(A,x) - Q^0(B,c)) = 0.0 + 0.1(600 + 0.5 \cdot 0.0 - 0.0) = 60.0 \\
Q^2(A,a) &= Q^1(A,a) + \alpha(r_2 + \gamma \max_x Q^1(B,x) - Q^1(A,a)) = 0.0 + 0.1(80 + 0.5 \cdot 60.0 - 0.0) = 11.0 \\
Q^3(B,a) &= Q^2(B,a) + \alpha(r_3 + \gamma \max_x Q^2(A,x) - Q^2(B,a)) = 0.0 + 0.1(100 + 0.5 \cdot 11.0 - 0.0) = 10.55 \\
Q^4(A,b) &= Q^3(A,b) + \alpha(r_4 + \gamma \max_x Q^3(B,x) - Q^3(A,b)) = 0.0 + 0.1(60 + 0.5 \cdot 60.0 - 0.0) = 9.0 \\
Q^5(B,c) &= Q^4(B,c) + \alpha(r_5 + \gamma \max_x Q^4(C,x) - Q^4(B,c)) = 60.0 + 0.1(70 + 0.5 \cdot 0.0 - 60.0) = 61.0 \\
Q^6(C,b) &= Q^5(C,b) + \alpha(r_6 + \gamma \max_x Q^5(A,x) - Q^5(C,b)) = 0.0 + 0.1(40 + 0.5 \cdot 11.0 - 0.0) = 4.55 \\
Q^7(A,a) &= Q^6(A,a) + \alpha(r_7 + \gamma \max_x Q^6(C,x) - Q^6(A,a)) = 11.0 + 0.1(20 + 0.5 \cdot 4.55 - 11.0) = 12.1275
\end{aligned}
$$

c)

The greedy policy wrt a Q function takes for a state the action maximizing the Q-value, i.e., $\pi(s) = \arg\max_a Q(s,a)$.

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

d)

$$
\begin{aligned}
Q^1(B,c) &= Q^0(B,c) + \alpha(r_1 + \gamma Q^0(A,a) - Q^0(B,c)) = 0.0 + 0.1(600 + 0.5 \cdot 0.0 - 0.0) = 60.0 \\
Q^2(A,a) &= Q^1(A,a) + \alpha(r_2 + \gamma Q^1(B,a) - Q^1(A,a)) = 0.0 + 0.1(80 + 0.5 \cdot 0.0 - 0.0) = 8.0 \\
Q^3(B,a) &= Q^2(B,a) + \alpha(r_3 + \gamma Q^2(A,b) - Q^2(B,a)) = 0.0 + 0.1(100 + 0.5 \cdot 0.0 - 0.0) = 10.0 \\
Q^4(A,b) &= Q^3(A,b) + \alpha(r_4 + \gamma Q^3(B,c) - Q^3(A,b)) = 0.0 + 0.1(60 + 0.5 \cdot 60.0 - 0.0) = 9.0 \\
Q^5(B,c) &= Q^4(B,c) + \alpha(r_5 + \gamma Q^4(C,b) - Q^4(B,c)) = 60.0 + 0.1(70 + 0.5 \cdot 0.0 - 60.0) = 61.0 \\
Q^6(C,b) &= Q^5(C,b) + \alpha(r_6 + \gamma Q^5(A,a) - Q^5(C,b)) = 0.0 + 0.1(40 + 0.5 \cdot 8.0 - 0.0) = 4.4 \\
Q^7(A,a) &= Q^6(A,a) + \alpha(r_7 + \gamma Q^6(C,c) - Q^6(A,a)) = 8.0 + 0.1(20 + 0.5 \cdot 0.0 - 8.0) = 9.2
\end{aligned}
$$

e)

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

f)

If the rewards are a function of s, a, they are not deterministic, since $s_1 = s_5 = B, a_1 = a_5 = c$ but $r_1 = 600 \neq 70 = r_5$.

However, if the rewards are a function of s, a, s', there are no 2 observations $i \neq j$ such that $(s_i, a_i, s_{i+1}) = (s_j, a_j, s_{j+1}), r_i \neq r_j$, thus they might be deterministic.

2 Part 2: policy gradient and function approximation

a)

For $i = 1$:

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

For $i \in \{1, ..., n\}$:

$$\begin{aligned}
\pi_\theta(s,i) &= \frac{\theta_i}{f(s)} \prod_{j=1}^{i-1} (1 - \frac{\theta_j}{f(s)}) \\
&= \frac{\theta_i}{f(s)} \prod_{j=1}^{i-1} \frac{f(s) - \theta_j}{f(s)} \\
&= \frac{\theta_i}{f(s)^i} \prod_{j=1}^{i-1} (f(s) - \theta_j)
\end{aligned}$$

For $i = n + 1$:

$$\begin{aligned}
\pi_\theta(s, n+1) &= \prod_{j=1}^{n} (1 - \frac{\theta_j}{f(s)}) \\
&= \prod_{j=1}^{n} \frac{f(s) - \theta_j}{f(s)} \\
&= \frac{\prod_{j=1}^{n} (f(s) - \theta_j)}{f(s)^n}
\end{aligned}$$

b)

For $i = k$:

For $i = 1$:

$$\begin{aligned}
\frac{\partial \ln \pi_\theta(s,1)}{\partial \theta_1} &= \frac{\partial}{\partial \theta_1} \ln \frac{\theta_1}{f(s)} \\
&= \frac{\partial}{\partial \theta_1} (\ln \theta_1 - \ln f(s)) \\
&= \frac{1}{\theta_1}
\end{aligned}$$

For $1 \leq i < n + 1$:

$$\begin{aligned}
\frac{\partial \pi_\theta(s,i)}{\partial \theta_i} &= \frac{\partial}{\partial \theta_i} \frac{\theta_i}{f(s)^i} \prod_{j=1}^{i-1} (f(s) - \theta_j) \\
&= \frac{1}{f(s)^i} \prod_{j=1}^{i-1} (f(s) - \theta_j) \\
&= \frac{\pi_\theta(s,i)}{\theta_i}
\end{aligned}$$

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

Alternatively:

$$\begin{aligned}
\frac{\partial \ln \pi_\theta(s,i)}{\partial \theta_i} &= \frac{\partial}{\partial \theta_i} \ln(\frac{\theta_i}{f(s)^i} \prod_{j=1}^{i-1}(f(s)-\theta_j)) \\
&= \frac{\partial}{\partial \theta_i}(\ln \theta_i - \ln f(s)^i + \sum_{j=1}^{i-1} \ln(f(s)-\theta_j)) \\
&= \frac{\partial}{\partial \theta_i} \ln \theta_i - \frac{\partial}{\partial \theta_i} \ln f(s)^i + \sum_{j=1}^{i-1} \frac{\partial}{\partial \theta_i} \ln(f(s)-\theta_j) \\
&= \frac{1}{\theta_i}
\end{aligned}$$

For $k < i$:

$$\begin{aligned}
\frac{\partial \ln \pi_\theta(s,i)}{\partial \theta_k} &= \frac{\partial}{\partial \theta_k} \ln(\frac{\theta_i}{f(s)^i} \prod_{j=1}^{i-1}(f(s)-\theta_j)) \\
&= \frac{\partial}{\partial \theta_k}(\ln \theta_i - \ln f(s)^i + \sum_{j=1}^{i-1} \ln(f(s)-\theta_j)) \\
&= \frac{\partial}{\partial \theta_k} \ln \theta_i - \frac{\partial}{\partial \theta_k} \ln f(s)^i + \sum_{j=1}^{i-1} \frac{\partial}{\partial \theta_k} \ln(f(s)-\theta_j) \\
&= \frac{\partial}{\partial \theta_k} \ln(f(s)-\theta_k) \\
&= \frac{1}{f(s)-\theta_k} \frac{\partial}{\partial \theta_k}(f(s)-\theta_k) \\
&= -\frac{1}{f(s)-\theta_k} \\
&= \frac{1}{\theta_k - f(s)}
\end{aligned}$$

For $k > i$:

$$\frac{\partial \ln \pi_\theta(s,i)}{\partial \theta_k} = 0$$

c)

The update rule is:

$$\theta \leftarrow \theta + \alpha_t(r_t + \gamma \max_b Q_\theta(s_{t+1}, b) - Q_\theta(s_t, a_t))\nabla_\theta Q_\theta(s_t, a_t) \tag{1}$$

Our goal is to approximate the optimal state action value function, the Q function, which follows the Bellman equation

$$Q(s,a) = r(s,a) + \gamma \sum_j p(j|s,a) \max_b Q(j,b)$$

For a parametrized approximation Q_θ, this leads to the Bellman error

$$BE(s,a) = r(s,a) + \gamma \sum_j p(j|s,a) \max_b Q_\theta(j,b) - Q_\theta(s,a) \tag{2}$$

where

$$y(s,a) = r(s,a) + \gamma \sum_j p(j|s,a) \max_b Q_\theta(j,b)$$

is the target and

$$Q_\theta(s,a)$$

the current estimate.

As a possible optimization objective it follows to minimize the mean square Bellman error

$$\begin{aligned}
J(\theta) &= \frac{1}{2}\mathbb{E}_{(s,a)\sim\mu_b}[BE(s,a)^2] \\
&= \frac{1}{2}\mathbb{E}_{(s,a)\sim\mu_b}[(r(s,a) + \gamma \sum_j p(j|s,a) \max_b Q_\theta(j,b) - Q_\theta(s,a))^2]
\end{aligned}$$

Since the target and the current estimate depend on θ, the gradient wrt θ of the objective $J(\theta)$ is

$$\begin{aligned}
\nabla_\theta J(\theta) =& \mathbb{E}_{(s,a)\sim\mu_b}[(r(s,a) + \gamma \sum_j p(j|s,a) \max_b Q_\theta(j,b) - Q_\theta(s,a)) \\
&(\sum_j p(j|s,a) \max_b \nabla_\theta Q_\theta(j,b) - \nabla_\theta Q_\theta(s,a))]
\end{aligned}$$

However, for the update rule, a semi gradient is used where only the derivative of the current estimate is taken

$$\cancel{\nabla}_\theta J(\theta) = -\mathbb{E}_{(s,a)\sim\mu_b}[(r(s,a) + \gamma \sum_j p(j|s,a) \max_b Q_\theta(j,b) - Q_\theta(s,a))\nabla_\theta Q_\theta(s,a)]$$

Its stochastic approximation at time step t is

$$\widehat{\cancel{\nabla}_\theta J(\theta)} = -(r_t + \gamma \max_b Q_\theta(s_{t+1},b) - Q_\theta(s_t,a_t))\nabla_\theta Q_\theta(s_t,a_t))$$

leading to the aforementioned update rule 1.

d)

The target refers to

$$y_t = r_t + \gamma \max_b Q_\theta(s_{t+1},b)$$

which is the stochastic approximation of the aforementioned target $y(s,a)$ in the bellman error 2.

Since the current parameter θ is also used in calculating the target, the target might vary quickly between successive update steps, which could lead to more unstable training.

To reduce this effect, the target parameterization can be fixed for a certain number of updates, i.e., a second parameter vector ϕ is introduced, which is updated to

$$\phi \leftarrow \theta$$

every C steps and the target is changed to

$$y_t = r_t + \gamma \max_b Q_\phi(s_{t+1},b)$$

leading to the new update rule

$$\theta \leftarrow \theta + \alpha_t(r_t + \gamma \max_b Q_\phi(s_{t+1}, b) - Q_\theta(s_t, a_t))\nabla_\theta Q_\theta(s_t, a_t)$$

This is for example used in the DQN algorithm.