Artificial Neural Networks (Gerstner). Solutions for week 2

Reinforcement Learning: Q-value and SARSA

Exercise 1. Recap of Online, Batch, and Expectations in ML (regression problem)[1]

We consider data coming from a distribution $p(x, y)$ where x is one-dimensional input and y the output. We have three possible x-values with $p(x = 0) = 0.2$ and $p(x = 1) = p(x = -1) = 0.4$. The output probabilities are

– $p(y = 1|x = 0) = p(y = 5|x = 0) = 0.5$

– $3 \cdot p(y = 1|x = 1) = p(y = 5|x = 1) = 0.75$

– $p(y = 1|x = -1) = 3 \cdot p(y = 5|x = -1) = 0.75$

The aim is to look at convergence of a linear estimator $f(x) = ax + b$ with a quadratic loss function $l\big(y, f(x)\big) = \big(y - f(x)\big)^2$, using gradient descent either in batch or in online mode.

a. Before you start, strengthen your intuitions. Suppose that we have drawn 50 points that match the statistical weights of the distribution. In total we have 50 points:
   – $\big\{(x_k, y_k) = (0, 1)\big\}_{k=1}^{5}$
   – $\big\{(x_k, y_k) = (0, 5)\big\}_{k=6}^{10}$
   – $\big\{(x_k, y_k) = (1, 1)\big\}_{k=11}^{15}$
   – $\big\{(x_k, y_k) = (-1, 5)\big\}_{k=16}^{20}$
   – $\big\{(x_k, y_k) = (-1, 1)\big\}_{k=21}^{35}$
   – $\big\{(x_k, y_k) = (1, 5)\big\}_{k=36}^{50}$

   Take a piece of paper, draw the points, and draw by eye the best linear fit. Trough which y-value should the line pass at $x = 0$ or at $x = 1$?

b. Here you can decide whether you would like to check the result in simulation (i) OR in theory (ii), i.e., you do not need to do both. Question (c) is again for everybody.

   (i) Simulation: Implement the task in Python: Use gradient descent with a small but fixed learning rate η. Follow the convergence of the parameters a and b to their final values. Compare batch mode (full batch of 50 samples) with online mode (one sample at a time). How do the fluctuations $\langle(\Delta b)^2\rangle$ scale with learning rate η? e.g., linear, quadratic, or other?

   Instead of the 50 handpicked examples above, draw N examples randomly and independently from the distribution $p(x, y)$. Run BATCH mode for a small η and observe the final result as a function of N. How close is it to the optimal solution?

   Start at the optimal solution for b that defines the minimum of the loss function for $N \to \infty$ but use only a few randomly picked samples. Do you stay at the exact minimum or does the solution move away? How about if you average your solution over 100 update steps? How does the answer to the last two questions change as a function of N?

   (ii) Theory: Solve for optimal parameters a^* and b^* analytically in the infinite data limit starting from the loss function. (*Hint:* you can also construct from general theorems of L2 loss and symmetries of the problem at hand).

   Then define the update step for the batch rule starting from arbitrary (a, b). Show that the update step is equal to 0 at (a^*, b^*).

   Start at the exact solution (a^*, b^*), but use the online rule by picking randomly from the 50 examples above. How big are the fluctuations? Based on calculations, how do the fluctuations $\langle(\Delta b)^2\rangle$ scale with the learning rate? e.g., linear, quadratic, or other?

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

c. Can you link the above results to the update of Q-values in the bandit problem? Can we relate Q-values to parameters? What would be states, actions, and rewards?

Solution:

a. See Figure 1. The line should pass through $y = \frac{5+1}{2} = 3$ at $x = 0$ and through $y = \frac{3*5+1}{4} = 4$ at $x = 1$.

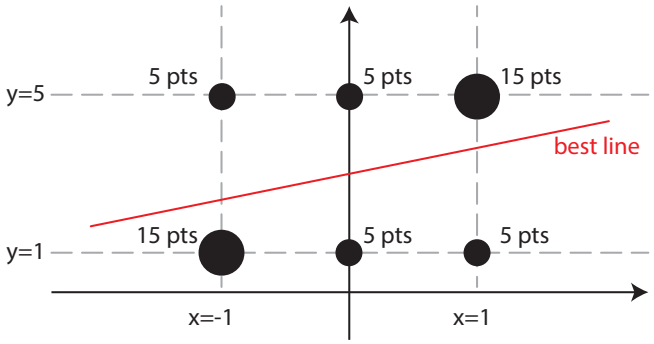

Figure 1: Solution to Exercise 1a.

b. (i) Simulation: See the corresponding Jupyter notebook for the code and the results. In summary, online mode estimates of a and b fluctuate around their mean (which is the optimal value), whereas the batch mode estimates converge to the optimal values. The online mode's fluctuations scale quadratically with the learning rate η. When sampling random data, the estimates are biased because of the finite sample size, but the bias is reduces as the sample size grows.

(ii) Theory: The loss function for the infinite data limit is

$$\mathcal{L}(a,b) = \sum_{x,y} p(x,y) l(y, f(x)) = \sum_{x,y} p(x,y)(y - ax - b)^2.$$

The optimal (a^*, b^*) is the solution to

$$\frac{\partial \mathcal{L}}{\partial a} = -2 \sum_{x,y} p(x,y)(y - a^* x - b^*) x = 0$$
$$\frac{\partial \mathcal{L}}{\partial b} = -2 \sum_{x,y} p(x,y)(y - a^* x - b^*) = 0$$

which results in

$$b^* = \mathbb{E}[y] - a^* \mathbb{E}[x] = \mathbb{E}[y] = 3$$
$$a^* = \frac{\mathbb{E}[xy] - b^* \mathbb{E}[x]}{\mathbb{E}[x^2]} = 1,$$

where we used $\mathbb{E}[x] = \sum_{x,y} p(x,y) x = 0$, $\mathbb{E}[y] = \sum_{x,y} p(x,y) y = 3$, $\mathbb{E}[x^2] = \sum_{x,y} p(x,y) x^2 = 0.8$, and $\mathbb{E}[xy] = \sum_{x,y} p(x,y) xy = 0.8$.
The update step for the batch rule is

$$\Delta a = \eta \frac{2}{N} \sum_{k=1}^{N} (y_k - a x_k - b) x_k = 2\eta \Big(\hat{\mathbb{E}}[xy] - a \hat{\mathbb{E}}[x^2] - b \hat{\mathbb{E}}[x] \Big)$$
$$\Delta b = \eta \frac{2}{N} \sum_{k=1}^{N} (y_k - a x_k - b) = 2\eta \Big(\hat{\mathbb{E}}[y] - a \hat{\mathbb{E}}[x] - b \Big),$$

where $\hat{\mathbb{E}}[x] = \frac{1}{N}\sum_k x_k = 0$, $\hat{\mathbb{E}}[y] = \frac{1}{N}\sum_k y_k = 3$, $\hat{\mathbb{E}}[x^2] = \frac{1}{N}\sum_k x_k^2 = 0.8$, and $\hat{\mathbb{E}}[xy] = \frac{1}{N}\sum_k x_k y_k = 0.8$. For the particular dataset mentioned in this exercise, we can replace $\hat{\mathbb{E}}$ by $\mathbb{E}$ in the equation above. Hence, $\Delta a = 0$ and $\Delta b = 0$ at (a^*, b^*).

The online update rule for sample k at (a^*, b^*) is

$$\Delta a = 2\eta(y_k - a^* x_k - b^*)x_k$$
$$\Delta b = 2\eta(y_k - a^* x_k - b^*).$$

The fluctuations $\langle(\Delta b)^2\rangle$ scales quadratically with η.

c. In the bandit problem, the state-action pair (s, a) is similar to the input x and the reward value r is similar to the output y. Each Q-value $Q(s, a)$ can be seen as a separate parameter. Iterative update of Q-values with the online rule leads to fluctuations of the Q-values. In contrast to the regression problem where the y-values of 3 different x-values are modeled by the function $f(x)$ with 2 parameters (which forces some interpolation), in the bandit problem, we have one separate Q-value for each $x = (s, a)$.

Exercise 2. SARSA algorithm

In the lecture, we introduced the SARSA (state-action-reward-state-action) algorithm, which (for discount factor $\gamma = 1$) is defined by the update rule

$$\Delta Q(s, a) = \eta\left[r - \big(Q(s, a) - Q(s', a')\big)\right], \tag{1}$$

where s' and a' are the state and action subsequent to s and a. In this exercise, we apply a greedy policy, i.e., at each time step, the action chosen is the one with maximal expected reward, i.e.,

$$a_t^* = \arg\max_a Q_t(s, a)\,. \tag{2}$$

If the available actions have the same Q-value, we take both actions with probability 0.5.

Consider a rat navigating in a 1-armed maze (=linear track). The rat is initially placed at the upper end of the maze (state s), with a food reward at the other end. This can be modeled as a one-dimensional sequence of states with a unique reward ($r = 1$) as the goal is reached. For each state, the possible actions are going up or going down (Figure 2). When the goal is reached, the trial is over, and the rat is picked up by the experimentalist and placed back in the initial position s and the exploration starts again.

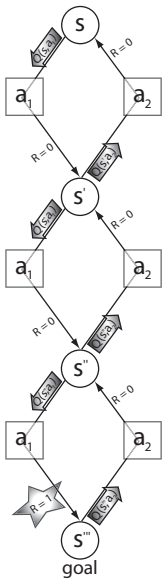

Figure 2: A linear maze.

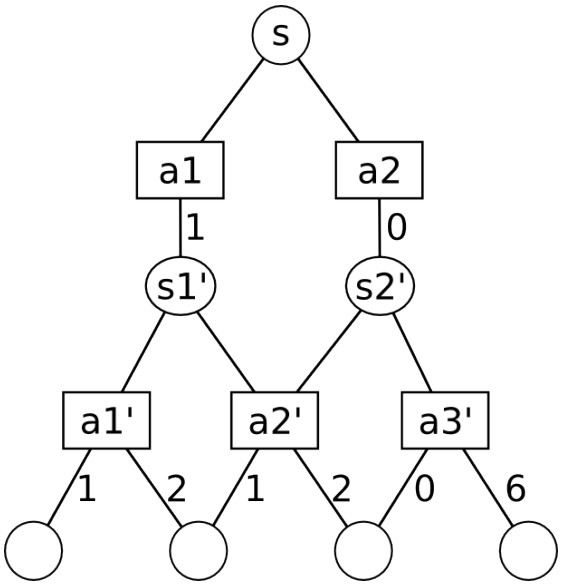

Figure 3: A tree–like environment.

a. Suppose we discretize the linear track by 6 states, $s_1, \ldots, s_6$ where s_1 is the initial state and s_6 is the goal state. Initialize all the Q-values at zero. How do the Q-values develop as the rat walks down the maze in the first trial?

b. Calculate the Q-values after 3 complete trials. How many Q-values are non-zero? How many trials do we need so that information about the reward has arrived in the state just 'below' the starting state?

c. What happens to the learning speed if the number of states increases from 6 to 12? How many Q-values are non-zero after 3 trials? How many trial do we need so that information about the reward has arrived in the state just 'below' the starting state?

Solution:

a. In the first trial, since all Q's are zero, the term $(Q(s,a) - Q(s',a'))$ is always zero. Learning only occurs when there is a reward ie, the first time action a_1 is taken from state s_5. The learning is then

$$\Delta Q(s_5, a_1) = \eta \left[r - (Q(s_5, a_1) - Q(s_6, a_2)) \right] = \eta \,, \tag{3}$$

so that now all Q are zero except for $Q(s_5, a_1) = \eta$.

b. In the second trial, the first time $\Delta Q(s,a)$ is not zero is when the agent takes action a_1 from state s_4, and we have

$$\Delta Q(s_4, a_1) = \eta \left[r - (Q(s_4, a_1) - Q(s_5, a_1)) \right] = \eta(0 - (0 - \eta)) = \eta^2 \,. \tag{4}$$

Next, from state s_5, the agent chooses the action with the highest Q value, a_1, and the weight update is

$$\Delta Q(s_5, a_1) = \eta \left[r - (Q(s_5, a_1) - Q(s_6, a_2)) \right] = \eta(1 - (\eta - 0)) = \eta - \eta^2 \,. \tag{5}$$

So at the end of the second trial, the non-zero Qs are:

$$Q(s_4, a_1) = \eta^2 \qquad \text{and} \qquad Q(s_5, a_1) = 2\eta - \eta^2 \,.$$

In the third trial, the first Q update happens for $Q(s_3, a_1)$

$$\Delta Q(s_3, a_1) = \eta \left[r - (Q(s_3, a_1) - Q(s_4, a_1)) \right] = \eta(0 - (0 - \eta^2)) = \eta^3 \,. \tag{6}$$

The subsequent updates are

$$\begin{aligned} \Delta Q(s_4, a_1) &= \eta \left[r - (Q(s_4, a_1) - Q(s_5, a_1)) \right] = \eta(0 - (\eta^2 - 2\eta + \eta^2)) = 2(\eta^2 - \eta^3) \\ \Delta Q(s_5, a_1) &= \eta \left[r - (Q(s_5, a_1) - Q(s_6, a_2)) \right] = \eta(1 - (2\eta - \eta^2 - 0)) = \eta - 2\eta^2 + \eta^3 \,. \end{aligned}$$

So after three trials, the Qs are:

$$Q(s_3, a_1) = \eta^3 \,, \qquad Q(s_4, a_1) = 3\eta^2 - 2\eta^3 \qquad \text{and} \qquad Q(s_5, a_1) = 3\eta - 3\eta^2 + \eta^3 \,.$$

Note that terms for all the Qs converge towards 1 (the reward after). The higher η is, the faster the convergence, with convergence in 1 step in the extreme case $\eta = 1$.

We need one more trial until $Q(s_2, a_1)$ becomes non-zero, i.e. in total 4 trials.

c. Also with 12 states only 3 Q-values are non-zero after 3 trials. It takes 10 trials until the reward has arrived just 'below' the starting state.

Exercise 3. Bellman equation

Use the Bellman equation to calculate $Q(s, a1)$ and $Q(s, a2)$ for the environment shown in Figure 3. Consider two different policies:

- Total exploration: All actions are chosen with equal probability.
- Greedy exploitation: The agent always chooses the best action.

Note that the rewards/next states are stochastic for the actions $a1'$, $a2'$ and $a3'$. Assume that the probabilities for the outcome of these actions are all equal, and the discount factor γ is 1.

Solution:

Total exploration: Since we have a directed graph without loops, the Bellman equation is solved via dynamic programming, starting at the bottom of the tree We start by computing the state-action values for states s_1' and s_2':

$$\begin{aligned} Q(s_1', a_1') &= \frac{1}{2}(1+2) = \frac{3}{2}, \\ Q(s_1', a_2') &= \frac{1}{2}(1+2) = \frac{3}{2}, \\ Q(s_2', a_2') &= \frac{1}{2}(1+2) = \frac{3}{2} \qquad \text{and} \\ Q(s_2', a_3') &= \frac{1}{2}(0+6) = 3. \end{aligned}$$

We can now compute the state-action values for state s:

$$\begin{aligned} Q(s, a_1) &= 1 + \frac{1}{2}(Q(s_1', a_1') + Q(s_1', a_2')) = \frac{5}{2} \qquad \text{and} \\ Q(s, a_2) &= 0 + \frac{1}{2}(Q(s_2', a_2') + Q(s_2', a_3')) = \frac{9}{4}. \end{aligned}$$

Greedy exploitation: In that case, the state-action values for the s_1' and s_2' are unchanged, but those for s reflect the fact that we now take the best action:

$$\begin{aligned} Q(s, a_1) &= 1 + Q(s_1', a_1') = \frac{5}{2} \qquad \text{and} \\ Q(s, a_2) &= 0 + Q(s_2', a_3') = 3. \end{aligned}$$

Notice that now the "best" action in state s is a_2, whereas it was a_1 for the total exploration policy.

Exercise 4. Computer exercises: Environment 1 (part 1)[1]

Download the Jupyter notebook of the 1st computer exercise, setup your Python environment, and complete 'Exercise 0: One step horizon'.

[1] Start this exercise in the second exercise session of week 2.