Artificial Neural Networks (Gerstner). Exercises for week 6

From Policy Gradient to Actor-Critic

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

Complete the computer exercise for environment 2.

Exercise 2. From Policy Gradient to eligibility traces

In this exercise you will show that eligibility traces appear naturally in any policy gradient algorithm. Eligibility traces are nice because they lead to a transparent and easy–to–interpret algorithm. Moreover, eligibility traces enable a direct online implementation of the algorithm in distributed hardware (or biology).

Consider a discrete multistep reinforcement learning problem with the usual graph, the usual notations and transitions: an action a_t leads you (stochastically) from state s_t to s_{t+1} and on this transition you collect the reward r_t. Suppose that you always start in state $s_{t=0} = s_{\text{start}}$. We assume that there is a simple terminal state s_{target}. You get a particularly strong positive reward when you reach s_{target}.

Your policy $\pi(a_t|s_t;\theta)$ depends on parameters θ. For the moment your aim is to optimize the parameters of the policy such that you maximize the expected discounted reward

$$\mathbb{E}_\theta[\text{Return}(s_{\text{start}} \to s_{\text{target}})] = \mathbb{E}_\theta[r_0 + \gamma r_1 + \gamma^2 r_2 + ...].$$

We proceed in five steps.

a. Derive a batch version of the policy gradient algorithm over multiple time steps by optimizing $\mathbb{E}_\theta[\text{Return}(s_{\text{start}} \to s_{\text{target}})]$ through gradient descent.

   Hint: Use the log-likelihood trick and take the derivative with respect to parameter θ_j.

b. A batch algorithm means averaging over many episodes. Transform the batch algorithm into an online algorithm where you consider one episode at a time. Assume that in one episode you traverse the state-action sequence: $s_0, a_0, r_0; s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; s_4, a_4, r_4; s_5 = s_{\text{target}}$.

   Show that the parameter updates can be written as

$$\begin{aligned}
\Delta\theta_j = \quad & [r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 r_3 + \gamma^4 r_4] \frac{\partial}{\partial \theta_j} \log[\pi(a_0|s_0;\theta)] \\
+ \quad & [\gamma r_1 + \gamma^2 r_2 + \gamma^3 r_3 + \gamma^4 r_4] \frac{\partial}{\partial \theta_j} \log[\pi(a_1|s_1;\theta)] \\
+ \quad & [\gamma^2 r_2 + \gamma^3 r_3 + \gamma^4 r_4] \frac{\partial}{\partial \theta_j} \log[\pi(a_2|s_2;\theta)] \\
+ \quad & [\gamma^3 r_3 + \gamma^4 r_4] \frac{\partial}{\partial \theta_j} \log[\pi(a_3|s_3;\theta)] \\
+ \quad & \gamma^4 r_4 \frac{\partial}{\partial \theta_j} \log[\pi(a_4|s_4;\theta)] \qquad (1)
\end{aligned}$$

c. So far we were only interested in maximizing the discounted future reward from the INITIAL state, with the discount factor computed relative to that state ($t = 0$). However, while you move along the trajectory you pass by other states s_1, s_2, s_3, s_4. For each of these states s_t, you should now also optimize the future expected discounted reward starting from s_t; that is you want to maximize

$$\mathbb{E}_\theta[\text{Return}(s_t \to s_{\text{target}})] = \mathbb{E}_\theta[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ...].$$

[1] Start this exercise in the first exercise session of week 6.

More generally, you should optimize the future discounted returns from every step t, assuming that the discounting started at the current step or at any possible step m in the past (i.e. $m \leq t$). Assume that m runs from $-\infty$ to t.

Redo the calculation in (b), but calculate the parameter update resulting from returns starting in arbitrary states.

Hint: Copy, but time-shift the results from (b).

d. Sum all the updates from (b) and (c) and reorder all terms such that updates that are multiplied with the same reward are grouped together.

Show that this results in updates of the form

$$\Delta\theta_j = \tag{2}$$

$$c \sum_t r_t \left[\frac{\partial}{\partial\theta_j} \log[\pi(a_t|s_t;\theta)] + \gamma \frac{\partial}{\partial\theta_j} \log[\pi(a_{t-1}|s_{t-1};\theta)] + \gamma^2 \frac{\partial}{\partial\theta_j} \log[\pi(a_{t-2}|s_{t-2};\theta)] + ... \right] \tag{3}$$

with some constant c. What is this constant?

e. Now we introduce eligibility traces by defining for each parameter θ_j a 'shadow variable' z_j which, in each time step t, decreases by a factor $\lambda < 1$

$$z_j \longleftarrow \lambda z_j \tag{4}$$

and then (in the same time step) increase by an amount

$$z_j \longleftarrow \frac{\partial}{\partial\theta_j} \log[\pi(a_t|s_t;\theta)] \tag{5}$$

where a_t is the action taken in time step t.

What is the relation of λ and γ? What is the final weight update?

f. Suppose that all rewards are zero, except the reward in the final time step $r_4 > 0$. Furthermore suppose that parameter θ is only sensitive to a_2, s_2. To be specific, say $\frac{\partial}{\partial\theta_j} \log[\pi(a_2|s_2;\theta)] > 0$ and $\frac{\partial}{\partial\theta_j} \log[\pi(a_t|s_t;\theta)] = 0$ for $t \neq 2$.

How can you interpret the resulting algorithm? How much will the parameter θ_j change?

Exercise 3. Recap and prepration for the next week: Why target networks help

States $s^{(j)}$ are represented by three-dimensional vectors $(s_1^{(j)}, s_2^{(j)}, s_3^{(j)})$. Actions are labeled by a 1-dimensional index $a = \{1, 2\}$. We look at semi-gradient Q-learning with linear function approximation, i.e. $Q(s^{(j)}, a) = \sum_{i=1}^{3} w_{ai} s_i^{(j)}$. We start with $w_{ai} = 0$ for all a and i.

Assume we observe state $s^{(1)} = (1, 1, 0)$, take action $a = 1$, receive reward $r = 1$ and observe the next state $s^{(2)} = (0, 1, 1)$.

a. Compute $Q(s^{(1)}, 1)$ with the semi-gradient learning rule $\Delta w_{ai} = \eta(r + \gamma \max_{a'} Q(s', a') - Q(s^{(1)}, a)) s_i^{(1)}$ with $\gamma = 1$ and $\eta = 0.1$.

b. Show that $Q(s^{(2)}, 1)$ has also changed.

c. Assume $\hat{Q}(s, a) = \sum_i w_{ai} s_i + \epsilon$, where ϵ is a Gaussian noise term with mean 0 and variance σ^2. Show that $\langle \max_a \hat{Q}(s, a) \rangle > \max_a \langle \hat{Q}(s, a) \rangle$.

Hint: Evaluations are for fixed state s. Expectations run over the Gaussian variable ϵ. The noise term ϵ is drawn independently for each action. Exploit that the mean of the Gaussian vanishes and that expectations can be easily evaluated for linear operators.