Artificial Neural Networks (Gerstner). Exercises for week 4

Policy Gradient

Exercise 1. (in Class): Single neuron as an actor

Assume an agent with binary actions $Y \in \{0,1\}$. Action $y = 1$ is taken with a probability $\pi(Y = 1|\vec{x};\vec{w}) = g(\vec{w} \cdot \vec{x})$, where $\vec{w}$ are a set of weights and $\vec{x}$ is the input signal that contains the state information. The function g is monotonically increasing and limited by the bounds $0 \leq g \leq 1$.

For each action, the agent receives a reward $R(Y, \vec{x})$.

a. Calculate the gradient of the mean reward $\langle R \rangle = \sum_{Y,\vec{x}} R(Y,\vec{x})\pi(Y|\vec{x};\vec{w})P(\vec{x})$ with respect to the weight w_j.
Hint: Insert the policy $\pi(Y = 1|\vec{x};\vec{w}) = g(\sum_k w_k x_k)$ and $\pi(Y = 0|\vec{x};\vec{w}) = 1 - g(\sum_k w_k x_k)$. Then take the gradient.

b. The rule derived in (a) is a batch rule. Can you transform this into an 'online rule'?
Hint: Pay attention to the following question: what is the condition that we can simply 'drop the summation signs'?

Exercise 2. Subtracting the mean

You have two stochastic variables, x and y with means $\langle x \rangle$ and $\langle y \rangle$. Angles denote expectations. We are interested in the product $z = (x - b)(y - \langle y \rangle)$ with a fixed parameter b.

a. Show that $\langle z \rangle$ is independent of the choice of the parameter b.

b. Show that $\langle z^2 \rangle$ is minimal if $b = \frac{\langle x f(y) \rangle}{\langle f(y) \rangle}$, where $f(y) = (y - \langle y \rangle)^2$.
Hint: write $\langle z^2 \rangle = F(b)$ and set $dF/db = 0$.

c. What is the optimal b, if x and $f(y)$ are approximately independent?

d. Make the connection to policy gradient rules.
Hint: take $x = r$ (reward) and y the action taken in state s. Compare with the policy gradient formula of the simple 1-neuron actor. What can you conclude for the best value of b? Consider different states s. Why should b depend on s?

Exercise 3. Policy gradient

a. **Policy gradient for binary actions**: Find an online policy gradient rule for the weights $\vec{w}$ for the same setup as in exercise 1 by calculating the gradient of the log-likelihood $\log \pi(Y|\vec{x};\vec{w})$ with respect to the weights. Hint: the policy π can be written as $\pi(Y|\vec{x};\vec{w}) = (1-\rho)^{1-Y}\rho^Y$ with $\rho = g(\vec{w} \cdot \vec{x})$.

b. **Other parameterizations**: What happens to the policy gradient rule in exercise 3.a if the likelihood ρ of action 1 is parameterized not by the weights $\vec{w}$ but by other parameters: $\rho = \rho(\theta)$? Derive a learning rule for θ.

c. **Generalization to the natural exponential family**: The natural exponential family is a family of probability distributions that is widely used in statistics because of its favorable properties. These distributions can be written in the form

$$p(Y) = h(Y)\exp\left(\theta Y - A(\theta)\right) . \tag{1}$$

This family includes many of the standard probability distributions. The Bernoulli, the Poisson and the Gaussian distribution (with fixed variance) are all member of this family. A nice property of these distributions is that the mean can easily be calculated from the function $A(\theta)$:

$$E[Y] = A'(\theta) . \tag{2}$$

Assume that the policy $\pi(Y|\vec{x};\theta)$ is an element of the natural exponential family. Show that the online rule for the policy gradient has the shape:

$$\Delta\theta = R(Y - E[Y]) . \tag{3}$$

Can you give an intuitive interpretation of this learning rule?

Exercise 4. Debugging of RL algorithms

You work with an implementation of 2-step SARSA and have doubts whether your algorithm performs correctly.

You have 2 possible actions from each state. You read-out the values after n episodes and find the following values:

$Q(1,a1) = 0$, $Q(2,a1) = 5$ $Q(3,a1) = 3$ $Q(4,a1) = 4$ $Q(5,a1) = 6$ $Q(6,a1) = 12$ $Q(7,a1) = 10$ $Q(8,a1) = 11$ $Q(9,a1) = 9$ $Q(10,a1) = 10$

$Q(1,a2) = 1$, $Q(2,a2) = 1$ $Q(3,a2) = 3$ $Q(4,a2) = 2$ $Q(5,a2) = 1$ $Q(6,a2) = 4$ $Q(7,a2) = 2$ $Q(8,a2) = 6$ $Q(9,a2) = 11$ $Q(10,a1) = 10$

You run one episode and observe the following sequence (state, action, reward)

$(1,a2,1)$ $(2,a2,1)$ $(3,a1,0)$ $(5,a1,4)$ $(6,a1,1)$ $(8,a2,1)$

What are the updates of 2-step SARSA that the algorithm should produce?

Exercise 5. Analysis of RL algorithms

Your friend proposes the following algorithm, using the pseudocode convention of Sutton and Barto.

Initialize $Q(s,a)$ = 0 for all $s \in \mathcal{S}, a \in \mathcal{A}$
Initialize π to be ε-greedy
Parameters: step size $\alpha \in (0,1]$, small $\varepsilon > 0$
All store and access operations (for S_t, A_t, and R_t) can take their index mod 4

Repeat (for each episode):
  Initialize and store $S_0 \neq$ terminal
  Select and store an action $A_0 \sim \pi(\cdot|S_0)$
  $T \leftarrow$ 10000
  For $t = 0, 1, 2, \ldots$:
  | If $t < T$, then:
  | Take action A_t
  | Observe and store the next reward as R_{t+1} and the next state as S_{t+1}
  | If S_{t+1} is terminal, then:
  | $T \leftarrow t+1$
  | else:
  | Select and store an action $A_{t+1} \sim \pi(\cdot|S_{t+1})$
  | $\tau \leftarrow t -$ 3
  | If $\tau \geq 0$:
  | $X \leftarrow \sum_{i=\tau+1}^{\min(\tau+4,T)} \gamma^{i-\tau-1} R_i$
  | If $\tau + 4 < T$, then $X \leftarrow X + \gamma^4 Q(S_{\tau+4}, A_{\tau+4})$
  | $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha\,[X - Q(S_\tau, A_\tau)]$
  |
  Until $\tau = T - 1$

a. Is the algorithm On-Policy or Off-Policy?
   Answer: ....

b. What does the variable X represent?
   Answer ....

c. Is this algorithm novel, similar to, or equivalent to an existing algorithm?
   Answer (fill in/choose)
   This algorithm is identical/very similar to .... .
   There is no difference to the named algorithm/the main difference is ....

d. Is this algorithm a TD algorithm? What is the reason for your answer?

   Answer: Yes/No, because ....