CSC 449 Advanced Topics in Artificial Intelligence

Deep Reinforcement Learning
Exam 2
Fall, 2022

Name: Dillon Dahlke ID#: 7532491 Score: 90

Your solutions to these problems should be uploaded to D2L as a single pdf file by the deadline. You may turn in the solution up to two days late, with a penalty of 10% per day, and you should only upload one version of your solutions.

This exam is individual and open book. You may consult any reference work. If you make specific use of a reference outside those on the course web page in solving a problem, include a citation to that reference.

You may discuss the course material in general with other students, but you must work on the solutions to the problems on your own.

It is difficult to write questions in which every possibility is taken into account. As a result, there may sometimes be "trick" answers that are simple and avoid addressing the intended problem. Such trick answers will not receive credit. As an example, suppose we said, use the chain rule to compute $\frac{\partial z}{\partial x}$ with $z = \frac{7}{y}$ and $y = x^2$. A trick answer would be to say that the partial deriviative is not well defined because y might equal 0. A correct answer might note this, but would then give the correct partial derivative when $y \neq 0$.

35pts

1. (40 pts) Consider the following pseudo-code for a faulty SARSA algorithm:

procedure SARSA(number of episodes $N \in \mathbb{N}$
discount factor $\lambda \in (0, 1]$
learning rate $\alpha_n = \frac{1}{\log(n+1)}$)
Initialize matrices $Q(s,a)$ and $n(s,a)$ to $0, \forall s, a$
for episode $k \in 1, 2, 3, \ldots, n$ **do**
$t \leftarrow 1$
Initialize s_1
Choose a_1 from a ~~uniform distribution over the actions~~
while Episode k is not finished **do**
Take action a_t: observe reward r_t and next state s_{t+1}
Choose a_{t+1} from s_{t+1} using μ_t: an ε-greedy policy with respect to Q
if The current state is terminal **then** ▷ *Compute target value*

$$y_t = 0$$

else

$$y_t = r_t + \lambda \max_a \text{[illegible]}_{+1}, a)$$

end if
$n(s_t, a_t) \leftarrow n(s_t, a_t) + 1$
Update Q function:

$$Q(s_{t+1}, a_{t+1}) \leftarrow Q(s_t, a_t) + \alpha_{n(s_t,a_t)} \left(y_t - Q(s_t, a_t)\right)$$

$t \leftarrow t + 1$
end while
end for
end procedure

Find all of the mistakes in the algorithm. Explain why they are mistakes, and correct them.

Initialize Q values non-zero to be able to update value function

Should be a_t, not a_1 every iteration.

w.r.t. Q (eg. ε-greedy) Sarsa Actions are chosen based on the current policy.

update should be based on s_{t+1} & a_{t+1}

state is terminal

Should be discounted by multiplying with discount factor (specified as λ) Derived from SARSA update $a_t + a_{t+1}$

should be adding (+) because it is the difference between the target and current value (Derived from SARSA Update Rule)

55pts

2. (60 pts) Your friend found a variant of SARSA which is defined through a sequence of policies π_t (where $t \geq 1$), and consists of just changing (in the previous algorithm **after corrections**) the way the target is computed. The target becomes

weights the Q-value by action probs.

$$y_t = r_t + \lambda \sum_a \pi_t(a|s_{t+1}) Q(S_{t+1}, a),$$

proba given S_{t+1}

where $\pi_t(a|s)$ is the probability that a is selected in state s under policy π_t.

a) What sequence of policies (π_t) should you choose so that the corresponding variant of SARSA is on-policy? This variant is called Expected SARSA.

As long as the expected return is not greedy with respect to expected return, Expected SARSA is on-policy. If the action was greedy the sum would turn into a max function and essentially become Q-learning.

b) Consider an off-policy variant of SARSA corresponding to a stationary policy $\pi = \pi_t \forall t$. Under this algorithm, do the Q values converge? If so, what are the limiting Q values? Justify your answer.

One policy that might be used is ε-greedy. ε-greedy simplifies the Expected Sarsa update to essentially that of Q-learning. This means that it converges under the same conditions as Q-learning.