CSC 449 Advanced Topics in Artificial Intelligence

Deep Reinforcement Learning

Exam 2

Fall, 2022

Name: Dustin Richards ID#: 7314739 Score: ______

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 $y \neq 0$.

32 pts

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 $k \in 1,2,3,\dots,n$ 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 $\mu_t$: an $\varepsilon$-greedy policy with respect to $Q$
            if The current state is terminal then          ▷ Compute target value
                $$y_t = 0$$
            else
                $$y_t = r_t + \max_a Q(s_{t+1}, a)$$
            end if
            $n(s_t,a_t) \leftarrow n(s_t,a_t) + 1$
            Update Q function:
                $$Q(s_{t+1},a_{t+1}) \leftarrow Q(s_t,a_t) - \alpha_{n(s_t,a_t)}(y_t - Q(s_t,a_t))$$
            $t \leftarrow t+1$
        end while
    end for
end procedure
```

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

$y_t = r_t + \max_a Q(s_{t+1}, a)$ should be $y_t = r_t + \gamma Q(s_{t+1}, a_{t+1})$

This adds the missing discount factor λ and uses the pre-selected next action a_{t+1} instead of the best action in the next state.

other errors there

In the "Update Q function" section:

$Q(s_{t+1}, a_{t+1}) \leftarrow \dots$ should instead be $Q(s_t, a_t) \leftarrow \dots$

We don't want to be updating the quality value of the next state and action. We want to update the quality value of the current state and action.

The variable learning rate $\alpha_n = \frac{1}{\log(n+1)}$ seems somewhat suspicious as well.

This isn't in the Sarsa algorithms in the book as far as I know. Looking at it, I could see it improving the accuracy of the learned model at the cost of learning speed as it slows down learning the more times a state is visited.

50

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

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

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

For expected SARSA to be on-policy, we must massage π_t to behave in the same way as the selection of a_{t+1} , assigning a $1-\epsilon$ probablility to the best action and and $\frac{\epsilon}{num\_actions - 1}$ probability to all other actions.

how?

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.

Given that Q-learning is effectively an off-policy variant of SARSA, I see no reason why the Q values should not converge given a decent policy. Q-learning follows a greedy policy, always incrementing the state value based on the current best action to take from the state.

A totally random policy likely would not converge, choosing actions (or weights, for expected SARSA) that are not relevant to the quality matrix.

Could say quite a bit more