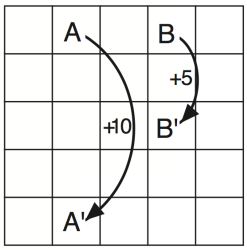
Advanced Machine Learning GR5242
Fall 2017

Homework 5

Due: Friday 01 December 4pm (for both sections of the class)

Homework submission: please submit your homework by publishing a notebook that cleanly displays your code, results and plots to pdf or html. You may wish to include another pdf file containing typeset or neatly scanned answers to the non-coding questions.

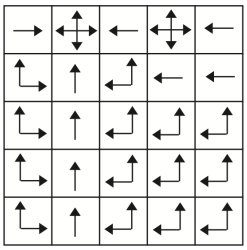
1. **Gridworld value** (20 points)
   In class we discussed Gridworld, its optimal policy $\pi_*(a|s)$, and state-value function $v_*(s)$:



Gridworld

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

v_*



π_*

   The discount factor for returns in this setup is $\gamma = 0.9$.

   Questions:

   (a) Recursively calculate $v_*(s = A)$, the value function at state A. Write a simple recursive function in python that follows the optimal policy and collects rewards. Terminate the recursion after $T = 200$ steps, starting from state A. Note that because the optimal policy is deterministic from $s = A$, the code need not be very complex (it doesn't need to know about Gridworld, just the intervals for accruing rewards 10.0, 0.0, 0.0,...). Produce your result to four decimal places. To check your work, note that this number should round to 24.4, as in the above figure.

   (b) Mathematically derive the exact expression for the same quantity $v_*(s = A)$. Your answer should not involve any sums, but instead should exploit the following geometric series identity (for some $\beta \in [0, 1]$, which (hint) is not $\gamma = 0.9$). Confirm this answer against your previous answer.

$$\sum_{k=0}^{\infty} \beta^k = \frac{1}{1-\beta}$$

   (c) Notice that the values $v_*(s) = \{22.0, 19.8, 17.8, 16.0, 14.4, 13.0\}$ are all repeated multiple times. Explain in words why this is the case, in terms of the optimal policy π_*. Relate these quantities mathematically to $v_*(s = A) = 24.4$.

2. **Deep Q-learning** (40 points)

In class we derived and implemented a Deep Q-learning algorithm for a game chosen from the OpenAI Gym. In this problem you will alter that implementation.

Questions:

(a) Set up the gym library in your environment; go to `https://gym.openai.com/docs/`.

(b) Download the example used in class and confirm that you can run it, reproducing the results shown in class. Save a few screenshots from tensorboard to confirm.

(c) Modify the network to increase performance. You should detail in writing what you tried and what performance resulted, and produce screenshots from tensorboard of the network configuration you consider best. Note that the amount of outperformance is less important than you having tried multiple network architectures and parameter choices to explore the differences. You should try at least 3 architectures and 2 different sets of parameters for each architecture.

3. **Project** (40 points)

Form a team and define a problem for your final project.

Questions:

(a) List the members of your team, including name, UNI, and course section. Recall that groups should be between 1 and 4 students.

(b) Describe the problem you will try to solve, in 5-7 sentences. It is understood that this answer and the following will be repeated by all team members.

(c) Describe the data you have for this problem. For example, how many training/validation/test samples do you have? What are the dimensionalities of the inputs and outputs? If an RL problem, what are the details of the states/actions/rewards?

(d) What is your starting point? For example, will a simple logistic regression get you started? What approaches already exist to solve this problem, and how difficult are they to implement? Describe in 3-5 sentences what steps you will have along the way to start from something simple and move to more complex networks. This, as we have discussed in class on several occasions, is critical to empiricism and working with deep learning.

(e) What do you anticipate the main challenge with the project will be? Describe in 3-5 sentences how you will approach identifying and fixing this issue if it arises.