Homework 2: Imitation Learning, GAIL Value Iteration and Policy Iteration

CMU 10-403: Deep Reinforcement Learning (Spring 2020)
OUT: Feb. 19, 2020
DUE: Mar. 05, 2020 by 11:59pm

Instructions: START HERE

- **Collaboration Policy:** You may work in groups of up to two people for this assignment. It is also OK to get clarification (but not solutions) from books or online resources after you have thought about the problems on your own. You are expected to comply with the University Policy on Academic Integrity and Plagiarism[1].
- **Late Submission Policy:** You are allowed a total of 10 grace days for your homeworks. However, no more than 3 grace days may be applied to a single assignment. Any assignment submitted after 3 days will not receive any credit. Grace days do not need to be requested or mentioned in emails; we will automatically apply them to students who submit late. We will not give any further extensions so make sure you only use them when you are absolutely sure you need them. See the Assignments and Grading Policy here for more information about grace days and late submissions: https://cmudeeprl.github.io/Spring202010403website/logistics/
- **Submitting Your Work:**
  - **Gradescope:** Please write your answers and copy your plots into the provided LaTeX template, and upload a PDF to the GradeScope assignment titled "Homework 2." Additionally, upload your code the GradeScope assignment titled "Homework 2: Code." Each team should only upload one copy of each part. Regrade requests can be made within one week of the assignment being graded.
  - **Autolab:** Autolab is not used for this assignment.
- **Kind Reminder:** Though this writeup looks long, some parts are quick questions that are designed to give you a better picture of what we have learnt so far.

[1] https://www.cmu.edu/policies/

Problem 0: Collaborators

Please list your name and Andrew ID, as well as those of your collaborators.

Problem 1: Behavior Cloning and DAGGER (50 pt)

In this problem, you will implement behavior cloning using supervised imitation learning from an expert policy. This problem and the GAIL one will be implemented in the following Colab notebook:

`https://colab.research.google.com/drive/1dlur0hqm9hrGBflReCzNCwLBt5MnGtb5`

This Colab notebook runs in the cloud, so you should not need to install anything on your laptop.

Background: Imitation Module

We have provided you with some function templates in the Colab notebook that you should implement. If you need to modify the function signatures, you may do so, but specify in your report what you changed and why.

Preliminaries (0pt)

The following is the same as HW1's CMA-ES question, you can copy your solutions from there.

First, you will implement a TensorFlow/Keras model. You will use this model a number of times in the subsequent problems. To test that your model is implemented correctly, you will use it to solve a toy classification task.

Next, you will implement a function to collect data using a policy in an environment. This function will be used in subsequent problems.

Behavior Cloning (25 pt)

Start by implementing the `train()` method of the `Imitation` class. To test your behavior cloning implementation, you will run the following cell titled "Experiment: Student vs Expert."

1. **[15 pts]** Run your behavior cloning implementation for 100 iterations. Plot the reward, training loss, and training accuracy throughout training.

2. **[10pts]** This question studies how the amount of expert data effects the performance. You will run the same experiment as above, each time varying the number of expert episodes collected at each iteration. Use values of 1, 10, 50, and 100. As before, plot the reward, loss, and accuracy for each, remembering to label each line.

DAGGER (25 pt)

In the previous problem, you saw that when the cloned agent is in states far from normal expert demonstration states, it does a worse job of controlling the cart-pole than the expert. In this problem you will implement the DAGGER algorithm [2]. Implementing DAGGER is quite straightforward. First, implement the `generate_dagger_data()` method of the `Imitation` class. Second, in the cell titled "Experiment: Student vs Expert," set `mode = dagger`.

1. [**10 pts**] Run your DAGGER implementation for 100. Plot the reward, training loss, and training accuracy throughout training. We have included code for plotting in the following cell.

2. [**10pts**] This question studies how the amount of expert data effects performance. You will run the same experiment as above, each time varying the number of expert episodes collected at each iteration. Use values of 1, 10, 50, and 100. As before, plot the reward, loss, and accuracy for each, remembering to label each line.

3. [**5pts**] Does your DAGGER implementation outperform your behavior cloning implementation? Generate a hypothesis to explain this observation. What experiment could you run to test this hypothesis?