Homework 3[1]

Due by 11:59pm 11/11/2018

Objective

This assignment is about building a reinforcement learning agent. In particular, you will have to build a Frogger agent, who must navigate its way home. Your frog needs to learn to perform the right actions -- up, down, left, right, or “no op” -- using Q-Learning.

Preliminaries

- Download and install the frogger support code.
- Follow the README within the frogger tar to install the other necessary modules.
    - If you use pip, make sure you are downloading and installing the 2.7 version of the modules
    - Make sure your python compiler knows where to look for the modules (set up an environment variable called PYTHONPATH if needed).

What You Need to Do

- Modify the Qagent.py code so that it performs Q-Learning. This will require you to run trials sequentially one after another. Make sure that you can save out and read in all the relevant information between trials.
- Decide how you want to set up the state space. In class, we talked about both a table-based state representation and a feature-based state representation. It’s up to you to decide which method to try.
- It is also up to you to experiment with different parameterization of the base algorithm:
    - Setting of the learning rate (alpha)
    - Exploration/exploitation trade-off
    - Reward discount constant (gamma)

What to commit

- All new source code you wrote (i.e., not the original frogger code)
    - Exception: if you’re **unable** to get your frog to work for the full environment, but you can demonstrate that your frog works for a simplified environment (e.g., no cars, or fixed cars that don’t move, etc.) then you should upload a copy of the code for the simplified environment. Be sure to state this clearly in the README (see below).
- Any relevant data file(s) you generate

[1] Shared Google Directory

- A README document:
  - Describe how to load and run your trained agent (note: the TA will not be able to train your frog from scratch, so you must submit appropriate data file(s) for your trained agent). You should also describe the format of your data file(s).
  - Describe how well your final agent is performing -- On average, how many frogs make it home safely per game? What is the average of the total reward the agent received per game? If your agent does not work for the full environment, describe to what extent your agent is able to exhibit any sort of learned behavior (e.g., can cross the street when the cars don't move)
  - Describe in sufficient details the choice you've made to get the learning framework going.
    - You must describe your final state representation in enough details for the grader so that he understands what you are trying to do.
    - If you have tried multiple state representations, discuss how they differ and how they impact your agent's learning.
  - Overall, how long did it take you to train the frog? What do you think you'd need to do to make the training more efficient? (Asked in another way, If you had more time to work on this project, what would you do next?)
  - if you used any additional tools or resources, give proper citations for them. If you've discussed the problem with other people, let us know the extent of your collaboration.

Grading

1. A serious attempt at training a frog agent -- for example, can save out and read in values, and the grader can recognize the basic structure of q-learning in code.
2. Basic infrastructure is there, but reinforcement learning is not entirely working -- perhaps the frog can successfully cross the road when the cars are stationary.
3. Demonstrate that the frog has learned to dodge moving cars: for example, show that the trained frog can make it to the middle island safely at least some of the times.
4. Demonstrate that at least one trained frog can get home in a game.
5. Demonstrate that at least three frogs can get home in a game; good representation choices. Insightful writeup.