Project #2

Problem

Description

For this project, you will be writing an agent to successfully land the “Lunar Lander” that is implemented in OpenAI gym. You are free to use and extend any type of RL agent discussed in this class.

Lunar Lander Environment

The problem consists of a 8-dimensional continuous state space and a discrete action space. There are four discrete actions available: do nothing, fire the left orientation engine, fire the main engine, fire the right orientation engine. The landing pad is always at coordinates (0,0). Coordinates consist of the first two numbers in the state vector. The total reward for moving from the top of the screen to landing pad ranges from 100 - 140 points varying on lander placement on the pad. If lander moves away from landing pad it is penalized the amount of reward that would be gained by moving towards the pad. An episode finishes if the lander crashes or comes to rest, receiving additional -100 or +100 points respectively. Each leg ground contact is worth +10 points. Firing main engine incurs a -0.3 point penalty for each occurrence. Landing outside of the landing pad is possible. Fuel is infinite, so, an agent could learn to fly and then land on its first attempt. The problem is considered solved when achieving a score of 200 points or higher.

Agent Representation

As noted earlier, there are four actions available to your agent:

do nothing, fire the left orientation engine, fire the main engine, fire the right orientation engine

Additionally, at each time step, the state is provided to the agent as a 8-tuple:

(x, y, vx, vy, θ, vθ, left-leg, right-leg)

x and y are the x and y-coordinates of the lunar lander's position. vx and vy are the lunar lander's velocity components on the x and y axes. θ is the angle of the lunar lander. vθ is the angular velocity of the lander. Finally, left-leg and right-leg are binary values to indicate whether the left leg or right leg of the lunar lander is touching the ground. It should be noted, again, that you are working in a six dimensional continuous state space with the addition of two more discrete variables.

Procedure

This problem is more sophisticated than anything you have seen so far in this class. Make sure you reserve enough time to consider what an appropriate solution might involve and, of course, enough time to build it.

- Create an agent capable of solving the Lunar Landar problem found in OpenAI gym
    - Upload/maintain your code to in private repo at https://github.gatech.edu
    - Use any RL agent discussed in the class as your inspiration and basis for your program
- Make a README.md file for your repository
    - Include thorough and detailed instructions on how to run your source code
    - Include the link to your private repo
- Create graphs demonstrating
    - The reward for each training episode while training your agent
    - The reward per trial for 100 trials using your trained agent
    - The effect of hyper-parameters (alpha, lambda, epsilon) on your agent
        - You pick the ranges
        - Be prepared to explain why you chose them
    - Anything else you may think appropriate
- Write a paper describing your agent and the experiments you ran
    - 5 pages maximum -- really, you will lose points for longer papers.
    - The paper should include your graphs.
        - And, discussions regarding them
    - Discuss your results
    - Explain the algorithms you selected and why you did
    - Describe any problems/pitfalls you encountered
    - Explain your experiments

    - Include a link to your presentation
    - Do not include a link to your code
- Create another README.txt with a link to your code.

Resources

The concepts explored in this project are covered by:

- Lectures
    - Generalization
- Readings
    - Sutton Ch. 9 On-Policy Prediction with Approximation
    - http://incompleteideas.net/sutton/book/bookdraft2017june19.pdf
- Source Code
    - https://github.com/openai/gym
    - https://github.com/openai/gym/blob/master/gym/envs/box2d/lunar_lander.py
- Documentation
    - https://gym.openai.com/docs
- Examples (use for inspiration, you are still required to write **your own** code)
    - https://gym.openai.com/envs/LunarLander-v2

Submission Details

Due Date: **October 29, 2017 (AOE)**

Submit:

1. Your **paper** as a pdf (containing your report)
2. A **README.txt** file (containing a link to your GitHub repository)

via T-Square:

Tips

- If you worked on HW4, you most likely checked out an older version of OpenAI gym. Please remember to update to the latest version for this assignment.
- If you get a Box2D error when running gym.make('LunarLander-v2'), you will have to compile Box2D from source. Please follow these steps:

```
pip uninstall box2d-py
git clone https://github.com/pybox2d/pybox2d
cd pybox2d/
python setup.py clean
python setup.py build
sudo python setup.py install
```

  Try running Lunar Lander again. Source: https://github.com/openai/gym/issues/100

- Even if you don't get perfect results or even build an agent that solves the problem you can still write a solid paper.