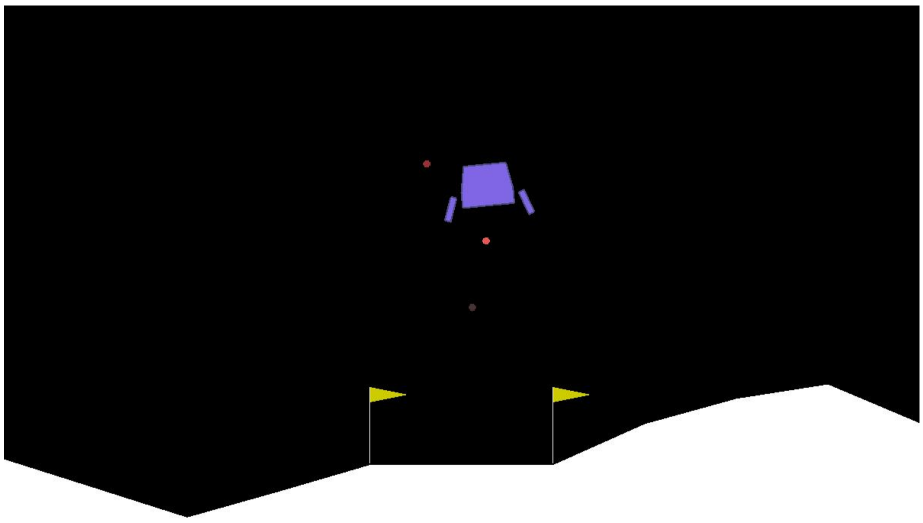

COMP47590

Advanced Machine Learning

Assignment 2: Lunar Lander

Introduction

In the OpenAI Gym game Lunar Lander (https://gym.openai.com/envs/LunarLander-v2/) the player's job is to control a small spaceship to land if safely on a landing pad. There are three *thrusters* which can be used for control. These work in three directions: *up, left,* and *right.* The player can also choose to do nothing. A dataset has been collected from an expert player of LunarLander that contains screenshots of the state of the game and the player's associated action (*none*, *up*, *left*, and *right).* A dataset describing the same scenario using high-level derived features representing the state of the Lunar Lander world has also been collected through the same process.

Tasks

Perform the following tasks:

1. Train a supervised machine learning model to control the Lunar Lander craft based on the extracted high-level state representations.
    - This dataset is contained in the file LunarLanderStateVectors.csv
    - The features in this dataset are as follows:
        - step: The step in the game that the player was on
        - pos_x: The x position of the centre of the spaceship - the middle of the landing pad is always at (0, 0)
        - pos_y: The y position of the centre of the spaceship - the middle of the landing pad is always at (0, 0)
        - vel_x: The velocity of the spaceship in the x direction
        - vel_y: The velocity of the spaceship in the y direction
        - ship_lander_angle: The angle between the centre of the spaceship and the centre of the landing pad
        - ship_lander_angular_vel: The spaceship's angular velocity with respect to the landing pad
        - leg_1_ground_contact: A binary flag indicating if the left leg of the spaceship is in contact with the ground
        - leg_2_ground_contact: A binary flag indicating if the left leg of the spaceship is in contact with the ground
        - action: The action that the expert player took in this scenario
    - Try out a few classifier types and select sensible hyper-parameters
    - Perform a suitable evaluation experiment to determine how effective the model trained is
2. Train a supervised machine learning model to control the Lunar Lander craft based on the image dataset.
    - Each image shows the state of the world and the image filename indicates the action that the expert player took in that scenario
    - You should consider converting the images to greyscale
    - You should considering shrinking the images
    - Try out a couple of model architectures and select sensible hyper-parameters

    - Perform a suitable evaluation experiment to determine how effective the model trained is
3. Use the DeepQLearning reinforcement learning algorithm to train an agent to play the Lunar Lander
    - Select sensible hyper-parameters
    - Perform a suitable evaluation experiment to determine how effective the model trained is
4. Deploy each of the three models trained to the Lunar Lander Game play 200 episodes and analyse the reward achieved by the models trained using each approach
    - The **lunar_lander_ml_images_player.py**, **lunar_lander_ml_state_player.py** and **lunar_lander_rl_player.py** python scripts contain the code to load a saved model and run iterations of the game using that model.
    - Write a short document (no more that 350 words) in a Jupyter notebook to describe the results of the experiments
    - Reflect on the performance of each model
    - Reflect on the amount of computation required to train each model

Notes

The following notes may be useful:

- **Can I Use Scikit-Learn, Keras And Other Python Packages?** Yes, and you absolutely should!
- **It's Taking Forever!** The datasets are big, and reinforcement learning takes a long time. If you find things are taking too long feel free to down-sample the dataset or not let reinforcement learning run too long. Submissions will not be penalised for this. Google Colaboratory (https://colab.research.google.com/) might be a useful resource for accessing computation.
- **Can I Work In A Team?** Teams of up to two people are allowed. All team members will receive the same mark. There is no penalty for submitting as a team, and no reward for submitting as an individual.
- **My Computer Controlled Players Are Not Very Good!** It is tricky for the supervised machine learning models to learn a complete model of the game so performance is unlikely to be amazing. Also the transition from desktop based evaluation looking at accuracy scores to actual deployment can shine a light on hidden limitations of a model. Lastly, if you do a lot of sampling of the data you might struggle to build something really good. Don't worry though, there will be no penalties for lack of performance as long as sensible modelling decisions are being made.
- **What No Templates?** No there are no templates, but there are plenty of template notebooks from the tasks performed during the course that you can use as a starting point. We will provide a template demonstrating keras-rl this week.

Submission

The key submission details for the assignment are as follows:

- **Submission date:** Monday 30th May 2018 before 23:59.
- **Submission method:** Submissions should be made through the module Moodle site.
- **Submission format:** Submissions should compose a zip file containing the following:
    - completed jupyter notebooks for each of the main tasks (image based learning, state based learning, reinforcement learning)
    - .html exports of each Jupyter notebook after execution that contains all output
    - any other files required to execute your code
    - a short notebook describing your conclusions from the experiments performed.
- **Late submissions:** Late submissions will be penalised at 5% penalty per day.

Marking

Marking of tasks will be based on the following weighting.

Task 1	25%	State based supervised learner
Task 2	30%	Image based supervised learner
Task 3	35%	RL based learner
Task 4	10%	Experiment reflections