Reinforcement Learning
Exercise 4

October 22, 2019

1 Function approximation

In many real world scenarios, the dimensionality of the state space may be too high to compute and store the Q-values for each possible state and action in a Q-table. In addition, as you'll find out in this exercise, tabular methods may suffer from low sample efficiency.

The state value and action value functions can be learned by using a function approximator, such as a radial basis functions or a neural network. In this exercise, you will start with basic features and build your way up to a simple DQN.

Please start working on this assignment early and don't leave it for the last moment, as the DQN part will take some time to train.

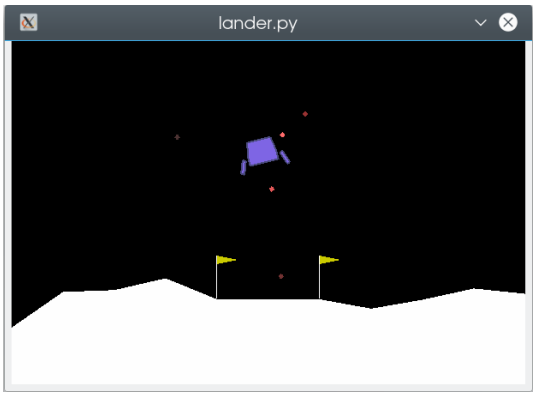

Figure 1: The Lunar lander environment.

Aalto University
School of Electrical
Engineering

Reinforcement Learning course staff
Intelligent Robotics Group
aalto.fi, irobotics.aalto.fi

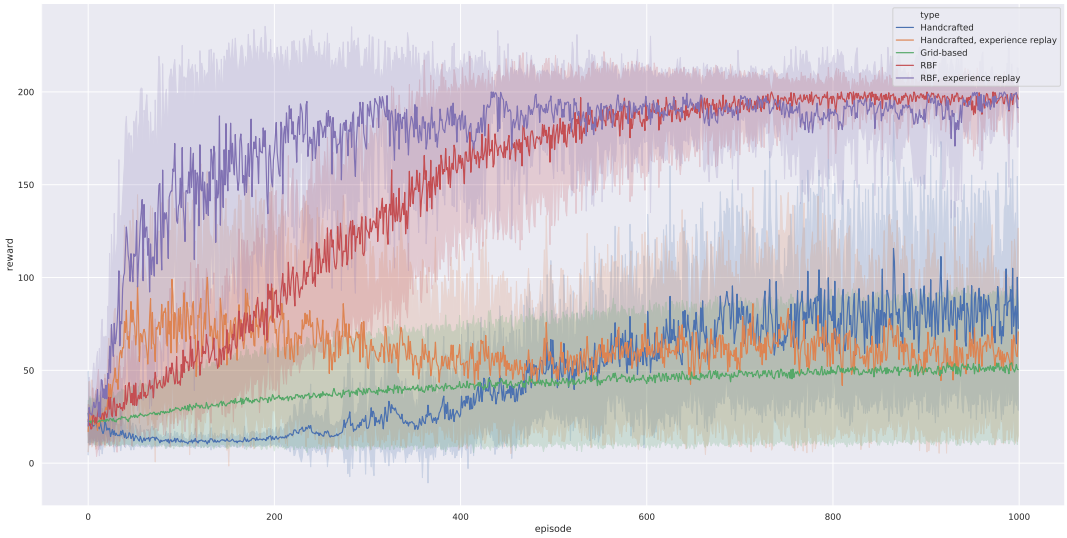

Figure 2: Training process with different methods

2 Approximation with non-linear features

Task 1 - 15 points Implement Q-learning using function approximation, at every timestep using the latest state transition to perform a TD(0) update. Test the implementation on the Cartpole environment. Test two different features for state representations:

(a) handcrafted feature vector $\phi(s) = (s \quad |s|)^T$,
(b) radial basis function representations (use the featurizer inside the Agent class).

Hint: You might need to install *scikit-learn*. It can be done by running: `pip install -U scikit-learn`.

Question 1 - 10 points Would it be possible to learn Q-values for the Cartpole problem using linear features (by passing the state directly to a linear regressor)? **Why/why not?**

Task 2 - 15 points Modify your Task 1 implementation to perform minibatch updates and use experience replay (**while keeping the original code for Task 1 submission**). Run the experiments with Cartpole with both feature representations.

Hint: Pass the elements to the `store_transition` function in the following order: s, a, s', r, $done$.

Question 2 Figure 2 shows the training performance of all four methods from Task 1, together with grid-based learning from Exercise 3 evaluated using GLIE with $a = 50$.

Question 2.1 - 10 points Which method is the most sample efficient, **and why**?

Question 2.2 - 5 points How could the efficiency of handcrafted features be improved?

Question 2.3 - 5 points Do grid based methods look sample-efficient compared to any of the function approximation methods? Why/why not?

Question 2.4 - 10 points Which hyperparameters and design choices impact the sample efficiency of function approximation methods?

Task 3 - 10 points Create a 2D plot of policy (best action in terms of state) learned with RBF with experience replay in terms of x and θ for $\dot{x} = 0$ and $\dot{\theta} = 0$.

3 A (not-so-)deep Q-network

Task 4 - 5 points Replace the RBFs in your Task 2 implementation with a neural network (**while keeping the original code for Task 2 submission**). A basic DQN implementation can be found in `dqn_agent.py`. Evaluate the method's performance in CartPole and LunarLander environments.

Question 3.1 - 5 points Can Q-learning be used directly in environments with continuous action spaces?

Question 3.2 - 15 points Which steps of the algorithm would be difficult to compute in case of a continuous action space? If any, what could be done to solve them?

4 Submission

Your report should be submitted as a **PDF file** as RL_Exercise4_*LastName_FirstName*.pdf.

The report *must* include:

1. **Answers to all questions** posed in the text (remember to answer all questions, specially in cases where there is a why/why not).
2. **Training plots** of all methods.
3. **Policy plot** from Task 3.

In addition to the report, you must submit as separate files, in the same folder as the report:

1. Python code used to solve **all task exercises**.

Please don't include any unnecessary files and directories, such as `__pycache__`, `.pyc` or `__MACOSX`.

Deadline to submit your answers is **04.11.2019, 11:55 am** (**in the morning**).

If you need help solving the tasks, you are welcome to visit the exercise sessions on Monday, Tuesday and Wednesday.

Good luck!