6CCS3ML1 and 6CCS3PRE Coursework 2 FAQ

Adrian Salazar and Yani

1 COURSEWORK OVERVIEW

This coursework exercise requires you to implement a Reinforcement Learning (RL) algorithm that will control Pacman's movement.

The RL algorithms are explained in the lectures from Weeks 8 and 9. Please consult the lecture slides and textbook for a complete explanation of them.

2 FREQUENTLY ASKED QUESTIONS

2.1 What are the important dates for this assignment?

This coursework will be released on the 11th of March and the deadline is on the 1st of April. The marks and feedback will be released on the 4th of May.

2.2 What do I need to submit?

Your submission should consist of a single ZIP file. (KEATS will be configured to only accept a single file.) This ZIP file must include a single Python file (your code called `mlLearningAgents.py`). The ZIP file must be named: `cw2-<lastname>-<firstname>.zip`

Submitting in a different format will lead to points being deducted.

DO NOT SUBMIT THE `mlLearningAgents.pyc` FILE! That is the compiled file, not the human readable file.

2.3 Which map should I use to test my agent?

Your code will be evaluated on smallGrid and another secret map. Your agent needs to win 8 out of 10 games on the smallGrid.

2.4 Why are you running it on a secret map?

This is done so we can check if your agent's behaviour is hard coded.

2.5 What does hard coded mean?

Hard coded behaviour means that the agent has a predefined set of actions he will be making, and not deciding based on what the RL algorithm is telling it to do.

2.6 Which one is better to use for this coursework, Q-learning or SARSA?

Either is good and will get to a big win rate for this coursework.

2.7 Can I implement an algorithm with function approximation?

Yes, but it is not needed for this map, as the number of states is small.

2.8 I am worried that the uncertainty of the environment will affect my grade. What will you do about that?

RL approaches are specifically designed to behave well in environments with high uncertainty. A well implemented RL agent will win all 10 games most of the time (the map is really small and the win condition is really easy to achieve). However, just to be sure, any agent that will not reach the requested win ratio will be re-run. If it does not reach the threshold the second time, then the 20 marks will not be awarded.

2.9 How should my code be styled?

You should comment your code and have a consistent style over all the file and a good separation of tasks in methods and classes.

2.10 Can I use code that I find online?

Depends if you do significant changes to it. Just copy/pasting will lead to marks being deducted or even to a review by the department's plagiarism committee. You will need to put references in your code where this happens and specify where you took it from.

2.11 What is that `final()` method?

When running several games in a run (i.e. you use the `-n` parameter), the `__init__()` will only be called at the beginning, when the object is created. As such, you will need to use the `final()` method to reset your values for the next game to the initial values (of course, that if it is needed).

2.12 Which reinforcement learning method should I use?

You can use any reinforcement learning approach. It does not matter if it is passive reinforcement learning or active reinforcement learning. Bear in mind that we will train the agent for 2000 rounds. So, make sure to use an algorithm that can win 8 out of 10 games in the smallGrid with that number of learning rounds.

2.13 Which reinforcement learning methods are sophisticated and which are not?

If you just use a simple bandit learner, that will not get as much credit as a Q-learner, or an implementation of adaptive dynamic programming. (And if you use the bandit learner that is in the solution posted for Practical 9 without acknowledgement, that is plagiarism.) Your algorithm needs to win at least 8 times. For example, we have algorithm A that is a reinforcement learning algorithm. Algorithm A works, but it is not consistent with the winning rate. Then, you should use another, more suitable algorithm for this coursework. Is it up to you to decide which method is the most suitable.

2.14 What libraries can I use?

You can use the standard python libraries. If you do not know what the standard python libraries are, here is a list with all of them: [LIST]. Additionally, you are allowed to use other basic libraries such as `numpy` and `pandas`.

2.15 What is a good coding style?

Depending on who you ask, this answer is going to be different. Generally speaking, the most important aspect for this matter is readability. To reach high level of readability, we recommend you use functions and classes. Use these so that the important aspects of your implementation are separated and well organised. Also, try to avoid the use of global variables.

2.16 How can I get the maximum grade in the comments part?

Good comments should explain how the different parts of your implementation work. These can be high-level explanations with some references to theory. If you are using functions, make sure to explain their input parameters. Finally, if you want, you can explain why you decided to use a specific approach and give details about your implementation. Use creativity for the last point.

2.17 Can I reuse my code from the 6CCS3AIN Coursework?

You can reuse parts of the code you wrote for that coursework, but it will need significant changes for it to get a good mark. Just resubmitting your MDP agent will not get you many points.