Adaptive Intelligence

Dr. Matthew Ellis
Prof. Eleni Vasilaki

Assignment 2

April 26, 2021

Reinforcement Learning

A robot moves in a square room with four possible actions; North, South, East and West. It has to learn a "homing" task to return to a particular reward location (e.g where for instance it can charge its battery). There are no explicit landmarks, but to simplify the task we assume that the robot has been familiarised with the environment and therefore it has some internal representation of its own position in the space, but no explicit memory of the reward location. The experiment contains multiple trials over which the robot is allowed to explore its environment until it finds the reward. At the start of the experiment a reward location is randomly chosen (but remains fixed while the robot is learning over multiple trials). Repeated simulations of the experiment will therefore have different reward locations.

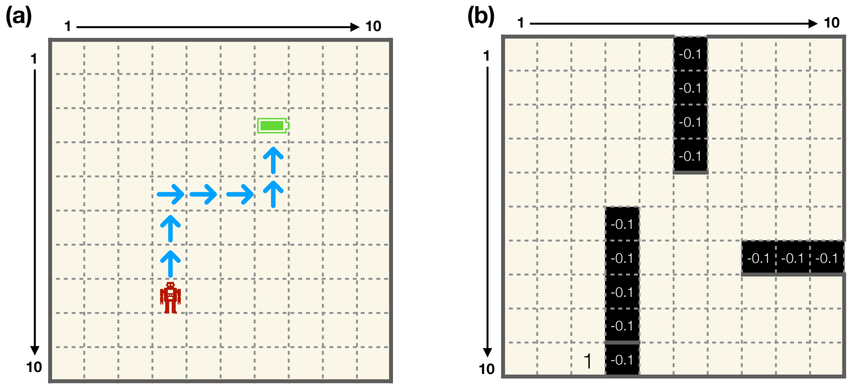

Figure 1: (a) Schematic of a robot agent searching for a reward location (battery). (b) Room layout for question 6 where each black section is a wall with a penalty of $r = 1$ and the reward location is location (3,10).

Your task is to write a program where the above-mentioned goal-oriented behaviour (homing) can be learned by using a reinforcement learning algorithm, in the following way:

1. The robot is placed at a random location (segment) of the room.

2. It explores the space/learns the goal oriented behaviour by using the SARSA algorithm with Q-values.

3. Reward is given when the robot reaches the segment where the charger is located.

4. Trial ends when the reward is reached or a predefined number of steps are exceeded in which case it is penalised with $r = -1$. The procedure starts from step 1 until a predefined number of trials is reached.

It is up to you whether the robot can move off the edge of the grid or not, and whether any penalty is applied. At the end, you should plot the number of steps it took the robot to reach its target vs the trial number (learning curve). In order to produce learning curves that allows for a comparison, you need to repeat the procedure (for at least 10 times, i.e. multiple runs) and produce a curve that is the average of the learning curves of the individual runs. On such an average curve we typically use error-bars that represent standard deviation or standard error (shaded errorbars if this helps in the visualisation). Given the random starting point, some trials will start closer to the reward location so it is important to first calculate what the minimum number of steps will be from the starting location to the final location. Successful learning means that the required number of steps is reduced as the trial number increases. We will call this procedure **one run** of the algorithm.

After developing the main routine (and confirming that your robot learns the desirable behaviour), use your code to study the properties of your model. Consider the robot moving in a grid world of 10 by 10 squares with the reward location giving a reward $r = 1$.
Your report should address these points:

1. Give a brief introduction to reinforcement learning and the tasks being considered here. What are the update rules that you will use for the other points of assignment and give a description of the algorithms.

2. Implement a policy that aids exploration (such as ε-greedy). **Introduce and discuss** the algorithm that you are using, what parameters control it and why, in general, exploration is important. **Plot** the average learning curves and show that the robot is exploring more states.

3. Find optimal values for the learning rate, discount factor and exploration factor (i.e epsilon for the ϵ-Greedy algorithm). You need only consider changing one parameter at a time. You can use the best parameter for the first value when trialing the second and so forth. The success of the learning can be measured by calculating the number of extra steps over the optimal path per trial averaged over the run. Any other suitable metric can be used as long as it is fully explained. **Plot your metric** against the parameter values and discuss your results.

4. Implement an eligibility trace with the SARSA(λ) algorithm. **Plot** the performance of the SARSA(λ) algorithm as you did for the SARSA algorithm in the last question. **Plot and explain** your results and compare them to your results in question 1.

5. Propose a method to reveal the information about the preferred direction stored in the weights (or Q values). Plot and **explain** your **plot**.

6. Imagine now that the space to be explored becomes bigger, and each edge of the square would be composed by N=1000 parts. What is the difficulty for the agent that has to find a reward location in this case? **Suggest a solution** (without programming it) to this problem.

7. Adjust your set up so that it includes some obstacles shown in figure 1.(b). There is a reward location (3,10) and each black square is a 'wall'. If the robot enters a 'wall' space then it receives a penalty of -0.1 and moves back to its previous position. In this situation, the robot cannot move off the edge of the grid so if it does return in to it's previous position. **Plot the preferred direction of each square** as above to show that your robot has learnt to navigate the space.

Please note: If you find difficult to implement the problem in 2D you may produce an 1D model and respond to the same questions. It will not get full marks (a scaling by 50% for this question will be applied), but a limited working solution is preferable to one that doesn't work.

1 Report

This is an individually written report of a scientific standard, i.e. in a journal-paper like format and style. It is recommended that you use the web to find relevant journal papers and mimic their structure. Results should be clearly presented and justified. **Figures should have captions, legends and readable axes labels**. Your report should **NOT exceed 5 pages** (excluding Appendix, References and Cover Page). Additional pages will be ignored. Two-column format is recommended. **The minimum font size is 11pt**. Kindly note that the readability and clarity of your document plays a major role in the assessment.

In the report you should include:

1. A response to all points requested by the assignment (including graphs and explanations). It is suggested to adopt a similar numbering scheme to make clear that you have responded all questions.
2. An Appendix with snippets of your code referring to the algorithm implementations, with comments/explanations.
3. A description of how your results can be reproduced, see also "Important Note".

Important Note: Please make sure that your results are reproducible. Together with the assignment, please upload your code well commented and with sufficient information (e.g. Readme file) so that we can easily test how you produced the results. If your results are not reproducible by your code (or you have not provided sufficient information on how to run your code in order to reproduce the figures), the assessment cannot receive full points. If no code is uploaded, the submission is considered incomplete and will not be marked.

2 Suggested language

The recommended programming language is Python/Matlab. However, on your own responsibility, you may submit the project in any language you wish, but you should agree it beforehand with the Lecturer.

3 Marking

Assignments will be marked with the following breakdown: results and discussions contribute up to 70%, scientific presentation and code documentation up to 20% and originality in modelling the task for up to 10%.

A mark greater than 39% indicates an understanding of the basic concepts covered in this course. A mark greater than 69% indicates a deep knowledge of the concepts covered in this course, with evidence of independent thinking or engagement beyond the level of material directly covered during lectures/laboratory sessions.

To maximise your mark, make sure you explain your model and that results are supported by appropriate evidence (e.g. plots with captions, scientific arguments). Figures should be combined wherever appropriate to show a clear message. Any interesting additions you may wish to employ should be highlighted and well explained.

4 Submission

The **deadline** for uploading the assignment to the VLE is **Monday 17th May 2021, 23:59** (in PDF format). Please also upload the corresponding code (zip file), with appropriate amount of detail for reproducing your results. Please double check that you have correctly submitted both components.

5 Plagiarism, and Collusion

You are permitted to use Python/Matlab code developed for the lab as a basis for the code you produce for this assignment with **appropriate acknowledgement**. This will not affect your mark. You may discuss this assignment with other students, but the work you submit must be your own. Do not share any code or text you write with other students. Credit will not be given to material that is copied (either unchanged or minimally modified) from published sources, including web sites. Any sources of information that you use should be cited fully. Please refer to the guidance on “Plagiarism, Collusion & Unfair Means” in the Undergraduate or MSc Handbooks.[1, 2]

References

[1] *Under-graduate Handbook*. URL `https://sites.google.com/sheffield.ac.uk/comughandbook/general-information/assessment/unfair-means`.

[2] *Post-graduate Handbook*. URL `https://sites.google.com/sheffield.ac.uk/compgtstudenthandbook/menu/referencing-unfair-means`.