Lecturer: Dr Eleni Vasilaki

Assignment

April 21, 2016

1 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, i.e. to return to a particular location, where for instance it can charge its battery (reward location). 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 position own position in the space, but no explicit memory of the reward location. 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 reward is reached (or a predefined number of steps is exceeded) and procedure starts from 1 until a predefined number of trials is reached.

At the end, you should plot the number of steps it took the robot to reach its target vs the trial number (learning curve). 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.

1. Running your code twice will produce a different learning curve. **Explain** why this happens. In order to produce learning curves that allow you comparison, you need to repeat the procedure (for at least 10 times, i.e. 10 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 errorbars that represent standard deviation or standard error.
2. Implement an eligibility trace. **Plot** the difference in performance (as average learning curves on the same figure with errorbars, see matlab commands *mean* and *errorbar*) with and without eligibility trace. **Explain** your results.

3. Find the optimal learning rate, discount factor and epsilon for the ϵ-Greedy algorithm. Show the **three plots** (combined in one), compare with the case above and **explain** your results.

4. Increase the number of actions from 4 to 8. What happens and for which reason? Show your results in a **plot** and **explain** them.

5. **Summarise** the problems that may occur with discrete state and action representation and **suggest a solution**.

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

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, but a limited working solution is preferable to one that doesn't work.

1.1 Implementation suggestion

You are free to choose your own implementation though the recommended method is to use an Artificial Neural Network (ANN), where the state is defined as an input vector to the network, and the action is determined by the output of the network. We assume that output units encode the Q values and learning takes place by updating their corresponding weights, see also class lectures and labs.

2 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 and legends. Your report should **NOT exceed 6 pages** (excluding Appendix and Cover Page). Additional pages will be ignored. Two-column format is recommended. The minimum font size is 11pt and margins should be reasonable. 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 proper 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 is not marked.

3 Suggested language

The recommended programming language is 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.

4 Marking

Assignments will be marked in a 0-100 scale. Results and explanations contribute for up to for 60 points, scientific presentation and code documentation for up to 20 points and originality in modelling the task for up to 20 points.

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.

5 Submission

The **deadline** for handing in **one copy** of the assignment to the Department of Computer Science Office (first floor, Regent Court, Portobello) is **Monday 16 May 2016, 3pm**. A **second copy** of the assignment (in PDF format) and the corresponding code should be uploaded in MOLE prior to the deadline, with appropriate amount of detail for running your code and reproduce your results.

6 Plagiarism, and Collusion

You are permitted to use Matlab code developed for the lab as a basis for the code you produce for this assignment with **appropriate acknowledgement**. You may discuss this assignment with other students, but the work you submit must be your own (or in pairs if submitting as groups assignment). 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 "Collaborative work, plagiarism and collusion" in the Undergraduate or MSc Handbook.