Reinforcement Learning Tutorial
COMS3007

Benjamin Rosman, Devon Jarvis

1. Use your notes and any resources you find online to answer the following questions. This does not need to be submitted.

   In this tutorial, we will create an agent that moves towards a goal on a grid world using reinforcement learning. The agent occupies a cell on a grid, and has to navigate to a goal location.

   A file "helpers.py" has been included which contains a small class called "Domain". This class may be used to create and interact with a grid world, including giving rewards, updating state and reflecting if a terminal state has been reached. To create an instance of the domain you can use "$world = helpers.Domain(length, breadth, seed)$" where $length$ and $breadth$ determine the dimensions of the grid. For this lab we will use $length = breadth = 4$. $seed$ is a value which be used to regulate the random behaviour of the domain. For our purposes it can be set by sampling a value for $np.random.seed(np.random.randint(0, 1000000))$. The goal state is always the bottom right most state of the grid. More details on the class can be found in the comments of helpers.py.

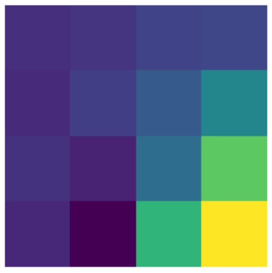
   (a) Create a function which chooses which action to take based on the current state of the grid world and learned state values. Use an ϵ-greedy action selection approach.

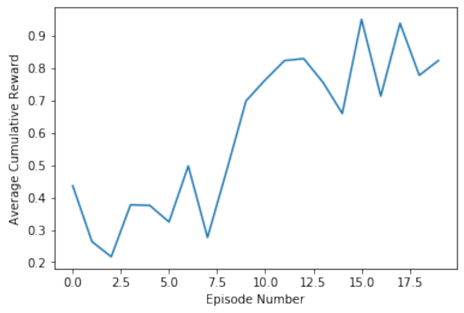
   (b) Implement the TD-learning algorithm to learn the state values of the grid world. Use the following parameter values: $\alpha = 0.2, \gamma = 0.9, \epsilon = 0.5$ to start. Learn the state values using 20 episodes. An episode runs until the agent moves into the goal state or has taken 15 moves at which point the agent is randomly reset to a new (non-goal) state in the domain. At the end of each episode print the current state values and also display them as an image (hint: plt.imshow, see Figure 1a for an example).

   (c) Track the cumulative reward of the agent for each episode. Plot the agent's cumulative reward at the end of each episode of training (see Figure 1b for an example). How consistent is the agent's performance? Does the agent improve consistently? Average the cumulative episode rewards over 20 separate trainings (remember to reset the learned state values between separate trainings). How consistent is the agent's performance on average? Does the agent improve consistently on average?

   (d) Reset the domain and use the learned state values to guide the agent through the domain. Display the current state of the domain after each step so that we can track the agent's trajectory through the domain (see Figure 1c for an example of a single time step of the trajectory).

   Note: if you want to see or record the final trajectory, turn off exploration, otherwise it will carry on taking some random actions, which wouldn't reflect what it has learned.


(a) Example State Values Image



(b) Example Average Cumulative Reward for 20 Training Episodes

(c) Example State Image

Figure 1: Example Plots