15-780: Graduate Artificial Intelligence
Homework 5

Due date: `5/2/2025 11:59pm`

Instructions

- Please provide detailed and well-reasoned responses to the questions below.
- Answers should be clear, concise, and well-supported with examples or explanations.
- Submit your responses as a PDF document and code on the course's gradescope page by the due date.

1 Q-Learning [50 pts]

In this problem you will implement a basic version of tabular Q-learning to train an agent to play CartPole (`https://www.gymlibrary.dev/environments/classic_control/cart_pole/`). Install OpenAI Gym with `pip install gym`. Here, the agent controls a cart on a frictionless 1-D track. A pole is attached to the cart by a single unmovable hinge. The pole starts in the upright position, and the goal is to keep the pole balanced by moving the cart left and right. The state of the CartPole environment is described by four parameters $[x, v, \theta, \omega]$ where x is the position of the cart, v is the velocity of the cart, θ is the angle of the pole, and ω is the angular velocity of the pole. Each episode consists of at most 500 time steps, and a reward of 1 is obtained for each time step where the pole is upright. The episode ends if the cart leaves the track ($x \notin (-2.4, 2.4)$) or if the pole falls ($\theta \notin (-.2095, .2095)$).

Code template. `https://www.cs.cmu.edu/~./15780/15-780_files/homeworks/hw5.zip`.

Preparing the environment. Since we are implementing tabular Q-learning, we need to operate on a suitable discretization of the state space. This is achieved by discretizing the range of states into evenly-spaced buckets (see the starter code for details).

1. [15 points] Implement `create_discretization`, `get_discretized_state`, and `create_q_table` in `q_learning.py`.

Q-Learning. You will now implement the Q-learning algorithm and use the epsilon-greedy policy to select actions. We'll train our agent over 10000 episodes.

2. [15 points] Implement `epsilon_greedy` and `q_update` in `q_learning.py`.

3. [5 points] (*Written*) Set `epsilon` = 1 and execute `cartpole.py`. What action-selection policy does `epsilon` = 1 correspond to? What is the average reward obtained by the agent? (A rough answer rounded to the nearest tens place is okay.)

4. [15 points] (*Written*) Play around with the parameter `epsilon` in `cartpole.py` until you see average scores that are at least 195 along the 10000 episodes. Plot the average score for every 100th episode and upload your plot, along with an explanation of the parameters you chose, to Gradescope. You may wish to implement some form of *epsilon decaying*, where `epsilon` starts at 1 and decays gradually to 0 over future episodes. Feel free to modify the learning rate and discount as well.

2 Search [25 pts]

You and your friends are given an arbitrary graph where the edge weights are all 1, the start is labeled s, and the goal is labeled e for "end". Your friend decides to create a heuristic

$$h(n) = \frac{k}{1 + \exp g^*(n)}$$

(a logistic function) and $h(e) = 0$, which she hopes to use with A* tree search or graph search to receive an optimal answer. $g^*(n)$ is the optimal path cost (from n to e). At first, she tries $k = 2$.

1) A* Criteria [15pts] Can she use the heuristic with $k = 2$ for either search algorithm? For each algorithm, state whether the criteria are met for optimality as discussed in class.

2) Admissibility [10pts] An admissible heuristic function is one that never overestimates the cost of reaching the goal state from a given node. What is the biggest value of k that would make the heuristic admissible? Give an example of a graph for which $k+1$ would not be admissible.