HW10: Reinforcement Learning

1 Getting started

Download the starter code from canvas. It consists of two files: Q-Learning.py and tests.py. You can create and activate your virtual environment with the following commands:

```
python3 -m venv /path/to/new/virtual/environment
```

```
source /path/to/new/virtual/environment/bin/activate
```

Once you have sourced your environment, you can run the following commands to install the necessary dependencies:

```
pip install --upgrade pip
```

```
pip install gymnasium==0.29.1 pygame==2.5.2 numpy
```

You should now have a virtual environment which is fully compatible with the skeleton code. You should set up this virtual environment on an instructional machine to do your final testing.

2 Q-Learning

For the Q-learning portion of HW10, we will be using the environment FrozenLake-v1 from Farama Gymnasium. This is a discrete environment where the agent can move in the cardinal directions, but is not guaranteed to move in the direction it chooses. The agent gets a reward of 1 when it reaches the tile marked G, and a reward of 0 in all other settings. You can read more about FrozenLake-v1 here: `https://gymnasium.farama.org/environments/toy_text/frozen_lake/`. **You will not need to change any code outside of the area marked TODO, but you are free to change the hyper-parameters if you want to.** For each sampled tuple $(s, a, r, s', \text{done})$, the update rule for Q-learning is:

$$Q(s,a) = \begin{cases} (1-\alpha)Q(s,a) + \alpha(r + \gamma \max_{a' \in A} Q(s',a')) & \text{if !done} \\ (1-\alpha)Q(s,a) + \alpha r & \text{if done} \end{cases}$$

The agent should act according to an ϵ-greedy policy as defined in the **Reinforcement Learning 1** slide. In this equation, α is the learning rate hyper-parameter, and γ is the discount factor hyper-parameter.

- *HINT: tests.py is worth looking at to gain an understanding of how to use the Farama Gymnasium env.*

3 Gymnasium Environment

You will need to use several Farama Gymnasium functions in order to operate your gym environment for reinforcement learning. As stated in a previous hint, tests.py has a lot of the function calls you need. Several important functions are as follows:

env.step(action)
Given that the environment is in state s, env.step takes an integer specifying the chosen action, and returns a tuple of the form $(s, r, \text{done}, \text{info})$. 'terminated' and 'truncated' specify whether the episode is completed, previously known as 'done' in OpenAI gym. 'info' is unused in this assignment.

env.reset()
Resets the environment to its initial state, and returns that state.

env.action_space.sample()
Samples an integer corresponding to a random choice of action in the environment's action space.

env.action_space.n
In the setting of the environments we will be working with for this assignment, this is an integer corresponding to the number of possible actions in the environment's action space.

You can read more about gymnasium here: `https://gymnasium.farama.org/api/env/`.

4 Submission

Please submit your file **Q_learning.py** and **QTABLE.pkl** to Gradescope. Do not submit a Jupyter notebook .ipynb file. You can test your learned policies, by calling python3 ./tests.py. Make sure to test your saved Q-tables using **tests.py** on the instructional machines with a virtual environment set up as specified above. This is the same program that we will be using to test your Q-tables and Q-network. Gradescope will test on various thresholds for score. You should aim for a score of ≥ 0.6 if you want to achieve full points.

The assignment is due Tuesday April 30 at 11:00 am central time. We are not accepting late submissions for this assignment.