Assignment 3 - Reinforcement Learning

Task 1 - Policy Evaluation (warm up)

This task is about evaluating policies given a finite Markov Decision Process (MDP) with a stationary transition probabilities. Specifically, your task will be to implement a method that does Policy Evaluation for the commonly used Grid World Problem.

For background on Policy Evaluation, please read Chapter 4.1. of Sutton, R. S., & Barto, A. G. (2018) - Reinforcement Learning: An Introduction. (For background on Markov Decision Processes, Policies and Values, see Chapter 3 of the same book.)

Implementation

In the first task, you will implement the algorithm for evaluating a policy. You will implement all the methods of this assignment in the partially complete template notebook **GridWorld.ipynb**. Fill in the code in the marked places, do not change anything else.

Setup

1. First create and activate an environment in conda. Run the following commands:
```
$ conda create -n aml python=3.7 jupyter
$ conda activate aml
```
2. Then, install the requirements. On the home directory run
```
$ pip install -r requirements.txt
```
3. Start a jupyter notebook server.
```
$ jupyter notebook
```

Preparation

Make yourself familiar with the rllib library and especially the data structures `TabularValueFunction`, `TabularPolicy`, and `EasyGridWorld` and its superclasses.

Task 1.1

Initialize the value function and policy variables by filling in the code in the method `init_policy_and_value_function()`.

Task 1.2

The value function satisfies the equation $V = r + \gamma PV$. Rearrange it to get an expression of V that you can solve in one step and implement it in the method `linear_system_policy_evaluation(environment, policy, gamma, value_function)`.

Task 1.3

Run policy evaluation and save the plot as part of your submission in a permanent format (png/pdf/...).

Task 2 - Policy Iteration

In this task, you will discuss and implement Policy Iteration as an approach to planning in the Grid World Problem.

For background on Policy Iteration, please read Chapters 4.2 and 4.3 of Sutton, R. S., & Barto, A. G. (2018) - Reinforcement Learning: An Introduction.

Theory Question 1 - Linear convergence of Policy Iteration

Important: *Please type up your answers, handwritten notes will not be accepted.*

Policy iteration works as follows:

1. Start with an arbitrary policy π.
2. Compute the value function V_π with respect to policy π
3. Compute a greedy policy π_G for V_π and use π_G as the policy π for the next iteration.
4. Repeat 2. and 3. until convergence.

Consider the policy iteration algorithm for infinite horizon MDPs with a discount factor $\gamma < 1$. Let π_t and π_{t+1} be the policies at time steps t and $t+1$ respectively, where π_{t+1} is the greedy policy based on the value function V_t.

The goal of this exercise is to prove linear convergence of policy iteration. For simplicity, we assume that the policy $\pi(x)$ is deterministic.

(a) Prove

$$BV^{\pi_t}(x) \geq V^{\pi_t}(x) \forall x \in X \tag{1}$$

where $B : \mathbb{R}^n \to \mathbb{R}$ is the Bellman operator defined as

$$BV^{\pi}(x) = \max_a \left[r(x,a) + \gamma \mathbb{E}_{s'|s,a}[V^{\pi}(x')] \right] \tag{2}$$

(b) Prove

$$V^{\pi_{t+1}}(x) \geq BV^{\pi_t}(x) \forall x \in X \tag{3}$$

Hint: Use induction over t with the fixed point iteration algorithm.

(c) Using the above inequalities, and the fact that $V^* \geq V^\pi \forall \pi$, prove the convergence of the Policy Iteration algorithm.

Hint: Use induction over t, the fact that B is a contraction, and the fact that $BV^* = V^*$.

Bonus points: show the rate of convergence is linear.

Implementation

Now you will implement the Policy Iteration algorithm. Fill in the code in the method `policy_iteration(environment, gamma)` at the marked locations. Do not change any of the already present code.

Task 2.1

Update the value estimate for each transition, i.e. each (state, action) pair.

Task 2.2

Check if the policy is stable, i.e. if any of the actions of the policy have changed since the last iteration.

Task 2.3

Run the policy iteration algorithm, and save the plots as part of your submission in a permanent format (png/pdf/...).

Task 3 - Value Iteration

In this task, you will use a different approach to obtain the optimal policy, namely Value Iteration.

For background on Value Iteration, please read Chapter 4.4 of Sutton, R. S., & Barto, A. G. (2018) - Reinforcement Learning: An Introduction.

Implementation

Fill in the code in the method `value_iteration(environment, gamma, eps=1e-6, max_iter=1000)` in the marked location. Do not change anything else.

Task 3.1

Initialize a variable that tracks the value of taking different actions.

Task 3.2

Calculate the value estimate for each transition and update the overall value estimate of the action.

Task 3.3

For each state, calculate the error between the last iteration and the current iteration, and set the value function of the state to the new value estimate.

Task 3.4

Run value iteration, and check if you get the same result as with policy iteration. How many iterations does it take?

Theory Question 2

In your own words, compare and contrast policy iteration and value iteration in a few sentences. I.e. what are the advantages and disadvantages of either? What are the types of problem to which they can be applied? Which is faster under which circumstances?

Task 4

Please describe the tasks and topics of the assignment, including your personal contribution, in **1000-1500 characters (you will lose points if you do not meet this limit!) Submit this description as a .txt file**.

Each team member must explain the main aspects of the assignment, and may not discuss this summary with other students.

This summary constitutes a separately and individually graded piece of work.