Reinforcement Learning Tutorial 4, Week 8

—

Value Function Approximation / Eligibility Traces

Pavlos Andreadis, Sanjay Rakshit

March 2020

[with special thanks to **Ross McKenzie** for introducing a first version]

Overview: The following tutorial questions relate to material taught in weeks 5 and 6 of the 2019-20 Reinforcement Learning course. They aim at encouraging engagement with the course material and facilitating a deeper understanding.

Problem 1 - Semi Gradient Monte Carlo

An AI controlled orchard needs to decide when to harvest its trees. To do this it measures the concentration of three chemicals in the air. Each day the orchard can choose to wait or harvest. Waiting costs one credit in operating costs while a harvest ends the process. Once a crop is harvested, packaged and sold, the orchard is told the profit or loss of that harvest. Most experts agree that the function mapping the chemical concentrations to the profit is approximately linear.

The orchard has several samples of the profits from other harvests. as seen in the table below:

Concentration of A (ppm)	Concentration of B (ppm)	Concentration of C (ppm)	Profit/Reward (credits)
4	7	1	3
10	6	0	-15
20	1	15	5
4	19	3	21

Begin to approximate the function that maps the state feature vector to Q(state, harvest) using a Monte Carlo target, doing a gradient decent step on each sample

(using linear function approximation). Solutions will be provided for a learning rate of 0.01, but feel free to try any value.

Problem 2 - Semi Gradient TD(λ)

The orchard also has the following record from its own last harvest:

Concentration of A (ppm)	Concentration of B (ppm)	Concentration of C (ppm)	Action Taken	Profit/Reward (credits)
6	7	2	Wait	-1
0	5	2	Wait	-1
3	8	4	Harvest	19

Continue to approximate Q(state, harvest) and start approximating Q(state, wait). Run through this episode using the True TD(λ) algorithm, with $\lambda = 0.1$. Instead of choosing your actions in an ϵ-greedy way, assume the ϵ-greedy procedure chose the actions in the table above.

Problem 3 - Discussion

Part a

Considering a Reinforcement Learning algorithm in general, what is the overall effect of increasing the learning rate? What happens when you set it too high? What happens when you set it too low?

Part b

Is the discount factor γ:

1. Part of the definition of a Markov Decision Process? That is, a part of the definition of the problem to be solved; or
2. Is it external to the problem? That is, a hyperparameter for training the model.

When a discount factor is close to 1, we end up with a long horizon problem. That is, we plan for long-term gains. Assume we were training a Reinforcement Learning agent for a long-horizon problem. Could you think of a reason for which a method using short-horizon targets (cut-off at some horizon h) might outperform a method using long-term horizon targets on this problem?