Report

CS 747: Programming Assignment 3

Name : Arun Babu PT

Roll No : 213050059

Date: 02/11/2021

1. Introduction

This assignment deals with the implementation of algorithms SARSA and SARSA with function approximation for "mountain task problem". This problem has been adopted from the *Mountain Car* task given as Example 9.2 by Sutton and Barto (2018). The first part deals with the direct implementation of the SARSA(0) and the second part involves implementation of SARSA(0) with function approximation.

2. Task 1

In this task we need to complete 3 functions given in the template code.

1. get_table_features to return a feature vector corresponding to a tabular representation
2. choose_action to select a action
3. sarsa_update to update Q values

get_table_features :

Each state is represented by a tuple (position, velocity).

As per the implementation of the environment in OpenAI Gym

position $\in$ [-1.2,0.6] and velocity $\in$ [-0.07,0.07].

Offsets -1.2 and -0.07 is added to position and velocity respectively to convert domain from 0 - max_value.The input state given by environment is also added the offset and find out indices of the bins to which each value in input array belongs.After that the corresponding indices for position and velocity is returned ad tuple

choose_action:

Action is selected with respect to epsilon using a greedy algorithm.With a epsilon probability exploit and else explore.

In task 2 tile coding is used as function approximation and the function get_table_features is created in a form that can also be used in tandem with linear function approximation. So maximum of state values of sum of each corresponding tile is used for exploit.

sarsa_update :

For sarsa update the algorithm discussed in the class (SARSA(0)) is implemented.

final weights for Task 1 is saved in T1.npy

The values of hyper parameters used by the algorithm is:

epsilon = 0.4

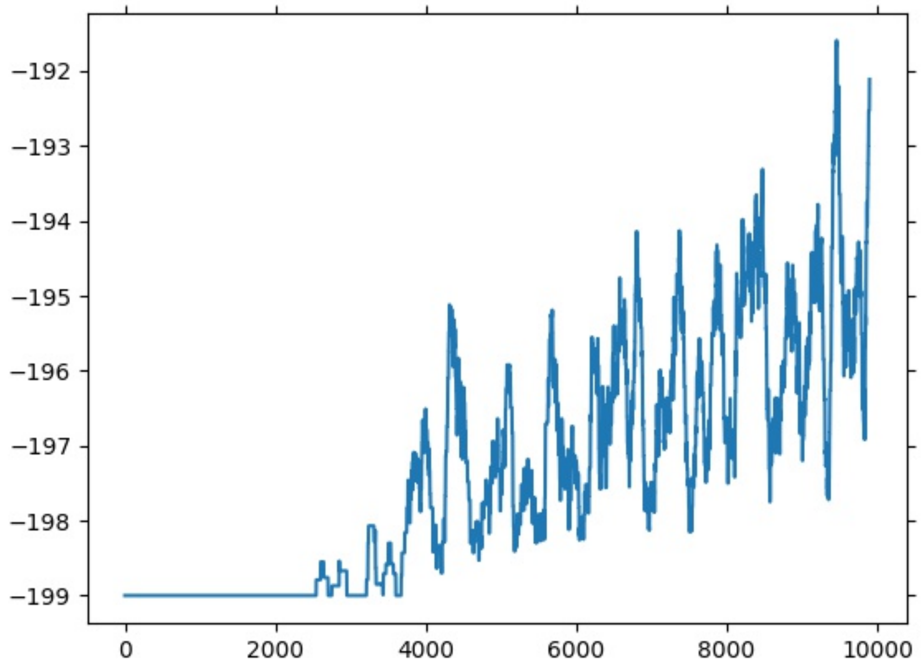
learning rate = 0.35

no of bins in position, velocity = 100, 80

weights are initialised by zeros

Maximum reward given by the algorithm for 100 episodes is = -146.16

Graph



The reward is less than -199 upto around 3000 and reward is increasing after that.But the reward is fluctuating when increasing the episodes.But we can observe that the average reward of a small interval of episodes is increasing when the number of episodes increases.Maximum reward is getting before reading 10000 episodes.

3. <u>Task 2</u>

For the task 2 function approximation used is Tile coding.

In tile coding, the variable space is partitioned into tiles. Any such partition is called a tiling. The method uses several overlapping tilings and for each tiling, maintains the weights of its tiles. The approximate value of a given point is found by summing the weights of the tiles, one per tiling, in which it is contained. Given a training example, the method adjusts the weights of the involved tiles by the same amount to reduce the error on the example.
[definition from Function Approximation via Tile Coding: Automating Parameter Choice , Alexander A. Sherstov and Peter Stone].

In this task number of tile used is 3. Domain is converted to positive and equally spaced bin is used for position and velocity. The offset used by the algorithm for shifting tiles in the right direction is (0.02, 0.02).
Each state is represent by an array of 3 tuples, each tuple represent where the state lies in each tiles.

The values of hyper parameters used by the algorithm is:

epsilon = 0.5

learning rate = 0.3

no of bins in position, velocity = 80, 55

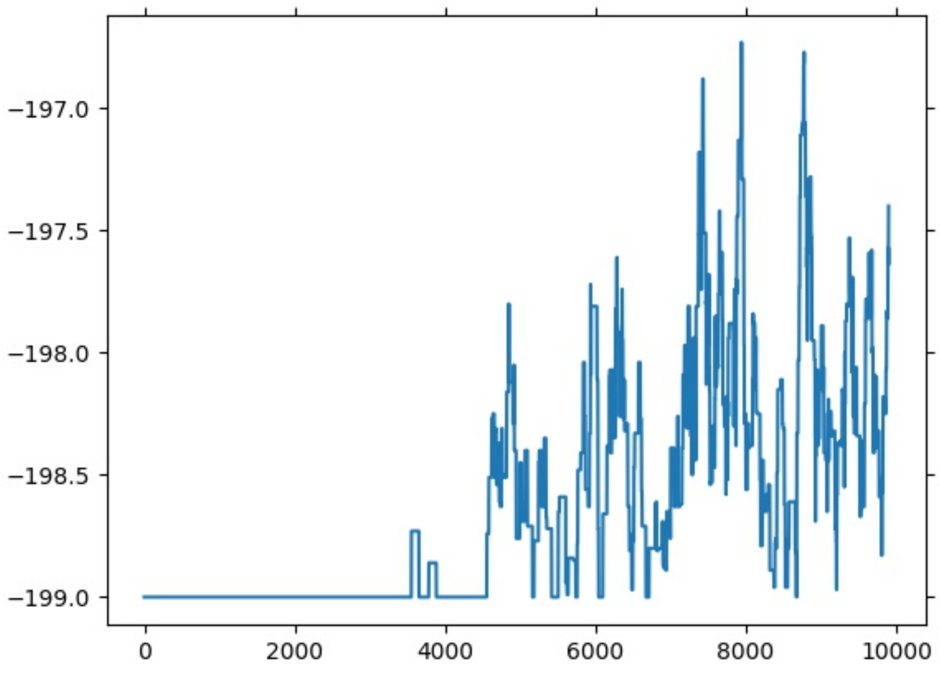
no of tiles = 3

offsets to shift tiles = 0.02, 0.02 (in x and y direction respectively)

weights are initialised by zeros

Maximum reward given by the algorithm for 100 episodes is = -136.37

Graph



The reward is less than -199 upto 3500-3700 range. and reward is increasing after that.But the reward is fluctuating when increasing the episodes.Clearly not increasing gradually. Getting maximum reward while training between 7000-9000 episodes.

Observations

During training the rewards is always less than -199 for some time and mean reward of a small interval is increasing over time.

The reward is more sensitive to the hyper parameters used. Even a small change will affect the average reward for test run.

And seed value is affect the reward . Changing the seed value from 0 to 1 will give a difference of ~30 in reward.

Increasing number of tiles to 4 is actually decreased the performance.