Assignment 1 – Coding Part

Majid Ghasemi

October 2024

1 Maze Coding

This is a report on the coding part of the first assignment. For all the tasks, to handle exploration, I used the epsilon-greedy approach. More specifically, I used decaying epsilon for this manner. I also used decaying for learning rate. Here is a set of hyperparameters and their values that are used for all the methods, and also can be seen from Fig. 1:

```
def epsilon_decay(self):
    self.epsilon = max(self.min_epsilon, self.epsilon * self.decay_rate)

def learning_rate_scheduler(self):
    self.lr = max(self.min_lr, self.lr * self.decay_rate)
```

Figure 1: Epsilon decay & learning rate scheduler

- learning rate = 0.1
- gamma (γ) = 0.9
- initial epsilon (ϵ) = 0.2
- minimum learning rate = 0.001
- minimum epsilon = 0.01
- decaying rate = 0.99
- number of episodes = 2000

γ is set to 0.9 because of the objective we have and the environment. We want to end up in the good state in a quickest time. Also, the initial epsilon is heavily exploratory but as I used 2000 episodes, I thought it is a good idea to start with a high epsilon and decay it with the rate of 0.99 to give the agent freedom to explore to find the best actions.

Also, the same thing is true for the learning rate (α).

The minimum numbers for epsilon and learning rate do not let ϵ and α go below that boundary.

Let us see thee results of the algorithms in each task.

1.1 Task 1

Fig. 2 shows that even though all of these algorithms tend to improve over time, Expected SARSA and SARSA have more stable and consistent learning patterns. Also, their rewards converged a little bit faster

because they have less fluctuation, whereas Q-Learning and Double Q-Learning had more changes before it stabilized around zero. Double Q-Learning performed worst.

The graph of path length also showed that all methods converged to reduce path lengths over time. Again, Expected SARSA and SARSA are superior to the rest. The path length was shorter and extremely consistent in both those methods. Double Q-Learning did some initial progress in shortening the paths but ended up converging towards the same level as Q-Learning relatively.

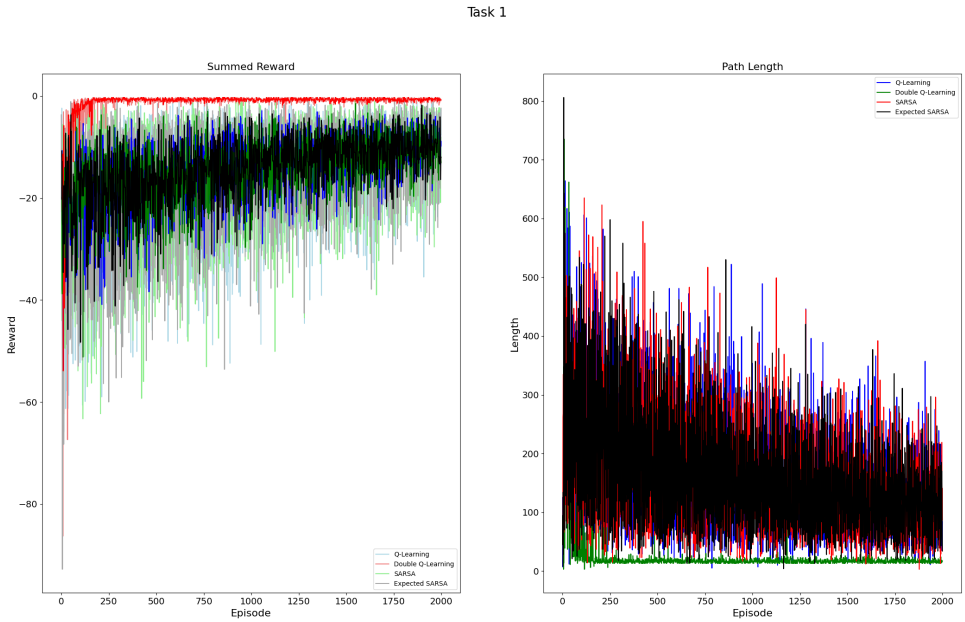

Figure 2: Task 1 Comparison

In summary, SARSA-based methods, particularly Expected SARSA, exhibit better stability and faster learning. In contrast, Q-Learning-based methods are less stable and take longer to reach convergence, making SARSA-based approaches more effective for this task.

Here is also the results achieved (output):

Experiment Setup:

- episodes: 2000
- sim_speed: 0.008

```
[Q-Learning on Task 1]
  max-rew = -0.900
  med-3 = -7.600
  var-3 = 9.016
  max-episode-len = 664.0

[Double Q-Learning on Task 1]
  max-rew = -0.300
  med-3 = -0.800
  var-3 = 0.016
  max-episode-len = 735.0

[SARSA on Task 1]
  max-rew = -0.700
```

```
  med-3 = -8.900
  var-3 = 4.809
  max-episode-len = 635.0

[Expected SARSA on Task 1]
  max-rew = -0.700
  med-3 = -5.700
  var-3 = 15.860
  max-episode-len = 806.0
```

1.2 Task 2

In the reward plot, the algorithms gradually improved over time. Expected SARSA and SARSA are most stable and converge the fastest - much like in the previous task. Changes across all algorithms are less noticeable this time, each method converged close to zero (even though with a different pace). The SARSA-based algorithms in general outperformed Q-Learning approaches again. Q-Learning approaches had more changes in early episodes but eventually got stabilised (to some extent I guess).

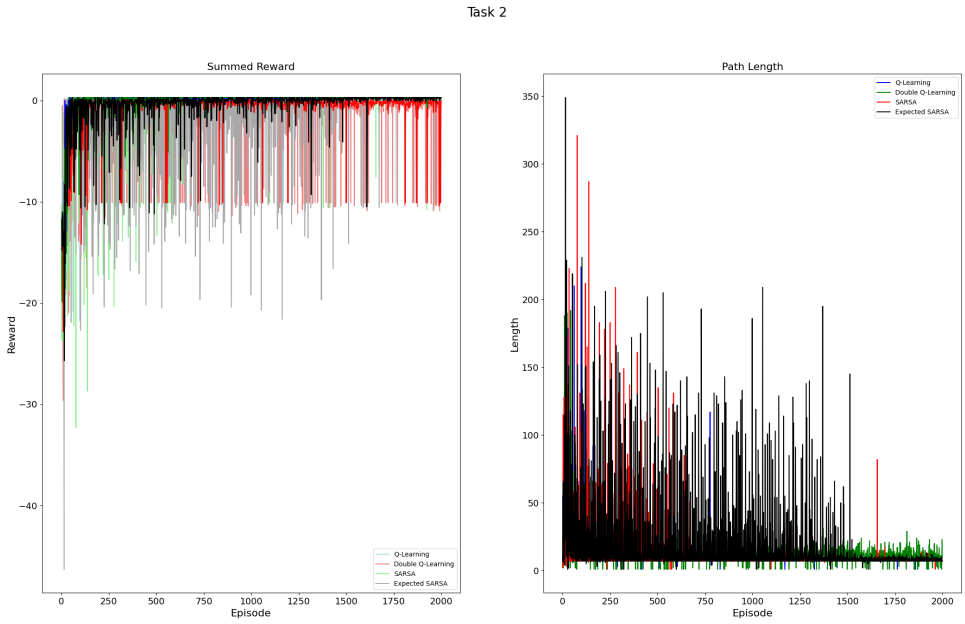

Figure 3: Task 2 Comparison

Experiment Setup:

- episodes: 2000
- sim_speed: 0.008

```
[Q-Learning on Task 2]
  max-rew = 0.300
  med-3 = 0.300
  var-3 = 0.000
  max-episode-len = 224.0

[Double Q-Learning on Task 2]
```

```
  max-rew = 0.200
  med-3 = -0.200
  var-3 = 0.327
  max-episode-len = 192.0

[SARSA on Task 2]
  max-rew = 0.300
  med-3 = 0.300
  var-3 = 0.020
  max-episode-len = 321.0

[Expected SARSA on Task 2]
  max-rew = 0.300
  med-3 = 0.300
  var-3 = 0.000
  max-episode-len = 349.0
```

As can be seen from the output, Double Q-Learning had a way higher variance compared to other algorithms, and that is the reason it did not perform good in this task (path length).

1.3 Task 3

In the plot of summed reward, Double Q-Learning was quite unstable during the beginning episodes, with sharp dips into negative rewards, especially within the first few hundred episodes. While this may be the case, despite these reward fluctuations, Double Q-Learning steadily stabilizes and begins converging closer to zero over time. But again, it varied and fluctuated a lot. If we look at the output, the max reward achieved by Double Q-Learning is much lesser and worse compared to other algorithms. Fig. 4 shows the details.

A similar pattern can be observed in the path length plot as well.

All other algorithms performed very well but it must be noticed that Q-Learning's variance is lesser than the others.

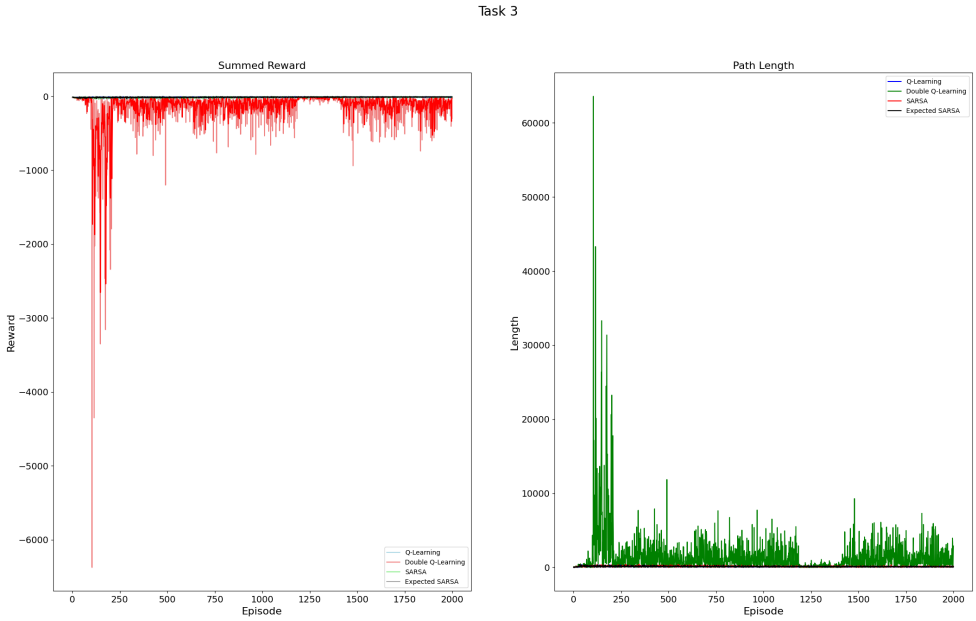

Figure 4: Task 3 Comparison

Experiment Setup:

- episodes: 2000
- sim_speed: 0.008

```
[Q-Learning on Task 3]
  max-rew = -0.800
  med-3 = -9.100
  var-3 = 4.527
  max-episode-len = 400.0

[Double Q-Learning on Task 3]
  max-rew = -5.900
  med-3 = -155.400
  var-3 = 12362.327
  max-episode-len = 63586.0

[SARSA on Task 3]
  max-rew = -1.000
  med-3 = -7.800
  var-3 = 10.916
  max-episode-len = 422.0

[Expected SARSA on Task 3]
  max-rew = -0.200
  med-3 = -5.700
  var-3 = 32.327
  max-episode-len = 403.0
```

1.4 Summary

In summary, Double Q-Learning performed worst in all the tasks among others. SARSA-based algorithms outperformed better than I expected to be honest.

Signed as: Majid Ghasemi , Ocotber 8