Homework-6, Multi Agent Systems

Ankur Satya

March 3^{rd} 2023

1 Monte Carlo Simulation

1.1 Optimal Batch Size

We ran simulations for the following values of infection probability, p, [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1] to determine the optimal value of k, the batch size. For every value of p, we tried multiple values of k but since k can vary from 1 to 10^6 (the population size), we only chose selected values of k to decrease the computation time. What values of k were chosen and why they were chosen will be explained in the following paragraph. For every value of k, 10^3 iterations were simulated so as to achieve the Monte Carlo Simulation process.

If we were to check for every value of k, i.e 10^6 values then the number of iterations would have been of the order 10^9(including 10^3 iterations for each value of k). So to counter this, only those values of k were chosen which were factors of 10^6. This enabled us to make a matrix of the size (N/k, k) where $N = 10^6$ and using this matrix we could check for the presence of the infection in all the batches faster with the help of the numpy package.

Figure 1 shows the num of tests conducted vs the batch size for different values of p. We can see from the figure that the optimal batch size decreases with the increase in the infection probability p. This makes sense because with the increase in the number of infection probability the number of cases will increase and hence their chance of being present in a batch which leads us to re-testing the whole batch. Table 1 shows the values of the optimal batch size for different values of p.

Metric/Probability	0.0001	0.0005	0.001	0.005	0.01	0.05	0.1
optimal batch size	100	50	32	16	10	5	4
workload reduction(%)	98	95.53	93.72	86.04	80.44	57.38	40.61

Table 1: Optimal batch size and workload reduction for different infection probability

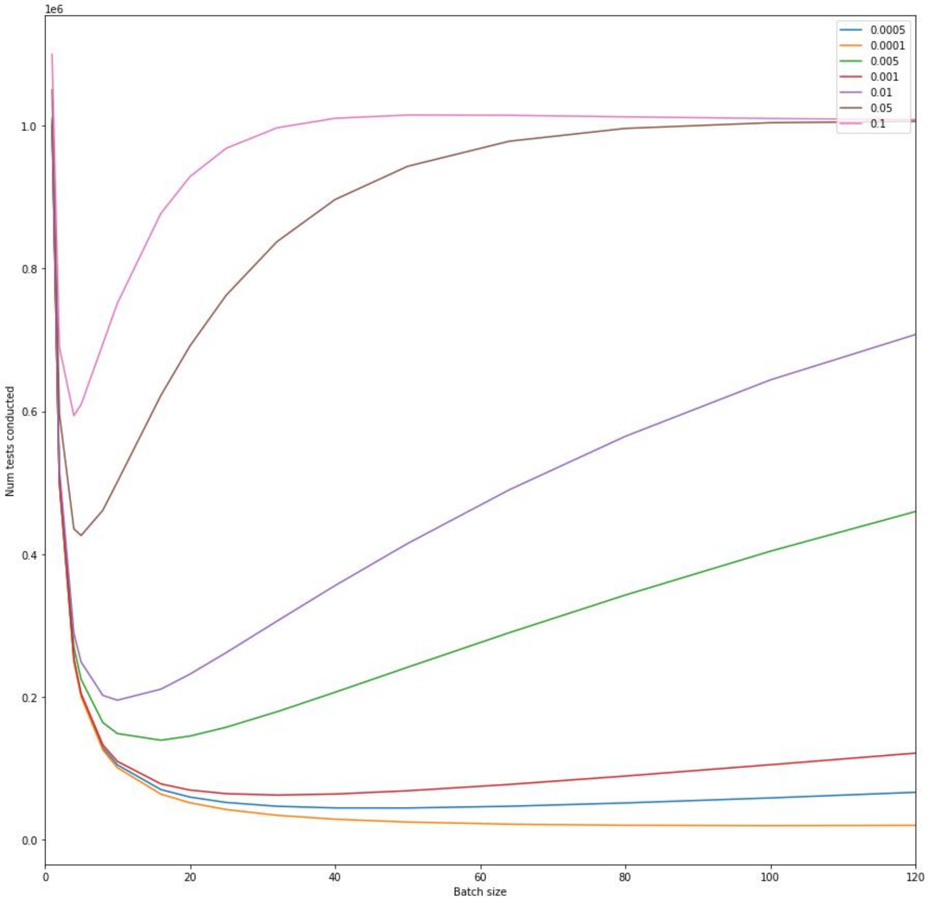

Figure 1: Batch size vs number of tests conducted for different value of infection probability.

1.2 Workload Reduction

The reasoning behind making batches and then testing them for infection was to decrease the overall amount of testing of the population. Table 1 shows the reduction in the workload as a percentage of the population i.e for $p = 0.0001$, the number of tests to be conducted decreased by 98%. Again, it makes sense that the workload reduction decreases with the increase in the infection probability as the number of batches to be made for testing increases.

2 Reinforcement Learning: Cliff Walking

2.1 SARSA vs Q-Learning

Figure 2 and figure 3 show the learned policy for SARSA and Q-learning respectively for $\epsilon = 0.1$. It is evident from these figures that the algorithm SARSA prefers the safe path as the learned policy takes the path that is as far as possible from the obstacles(which are present in the bottom row). On the other hand, the algorithm Q-learning prefers the optimal path which takes it through the row just above the obstacle row. This is the expected behaviour as mentioned in the text[1]. This difference in behaviour can be attributed to the fact that Q-learning always chooses the action for the new state by maximizing the Q-values over the action space.

Figure 4 represents the sum of rewards obtained per episode for both the algorithms. It is evident that SARSA has a better value of return per episode which makes sense since the policy for Q-learning chooses the actions that are very close to the obstacles resulting in occasional interaction with the obstacles.

2.2 Influence of ϵ

To check the influence of ϵ we used three values, [0.1, 0.3, 0.6]. Figure 2 to figure 4 show the SARSA policy, Q-learning policy and the sum of rewards per episode respectively for $\epsilon = 0.1$. Figure 5 to figure 7 show the same for $\epsilon = 0.3$ and figure 8 to figure 10 show the same for $\epsilon = 0.6$.

It is evident from the sum of rewards vs episodes figures that the convergence time increases as the value of ϵ increases since the probability of the greediest action being chosen decreases. This means that the algorithm will explore more but its exploitation tendency decreases.

From the policy figures for all the values of ϵ, we can conclude that the final learned policy remains the same for both the algorithms irrespective of the value of ϵ.

2.3 Modified Cliff Walking

For this part, another obstacle was added in the top row and in the center most column. This obstacle also has the same penalty of -100.

Figure 11 shows the policy learned by SARSA with $\epsilon = 0.1$. We observe that the learned path is still the safest path as the policy first suggests to go straight up from the start position. It then keeps going towards the right but then changes its path by coming down one row by still keeping the safe distance from the snake pit obstacle. After crossing this obstacle, the policy suggests going up again so as to be safe from the obstacles in the bottom most row. Rest of the policy is same as before.

Figure 12 shows the policy learned by Q-learning with $\epsilon = 0.1$. We observe that the learned path still remains the same, meaning the learned policy is unaffected by the presence of the new snake pit obstacle.

Figure 13 shows the sum of rewards per episode for both the algorithms when snake pit obstacle was also introduced to the grid. If we compare this figure with the figure 4, we can see that there is no significant difference between the sum of rewards for Q-learning across the two figures. On the other hand, there is a significant change in the sum of rewards for SARSA across the two figures. The presence of the snake pit obstacle in the top row pushes SARSA to find a balance between staying away from the bottom row of obstacles and the snake pit obstacle itself. And in an attempt to do so, SARSA sometimes interacts with the snake pit obstacle and sometimes it interacts with the bottom row of obstacles which leads to the generation of these fluctuations and a higher convergence time.

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: an introduction*. Cambridge, Massachusetts; The MIT press, 2014.

A Reinforcement Learning: Cliff Walking

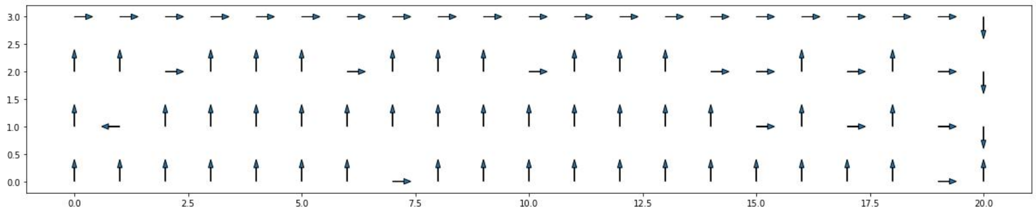

Figure 2: Learned policy for SARSA with $\epsilon = 0.1$

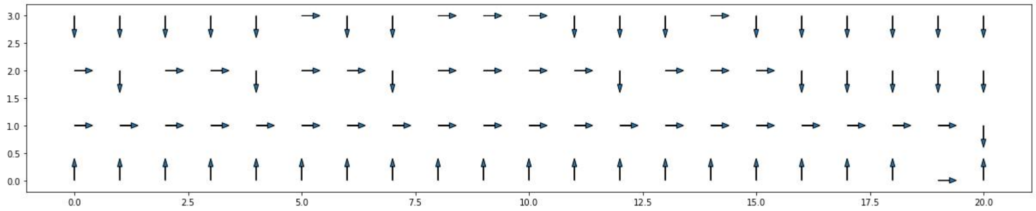

Figure 3: Learned policy for Q-learning with $\epsilon = 0.1$

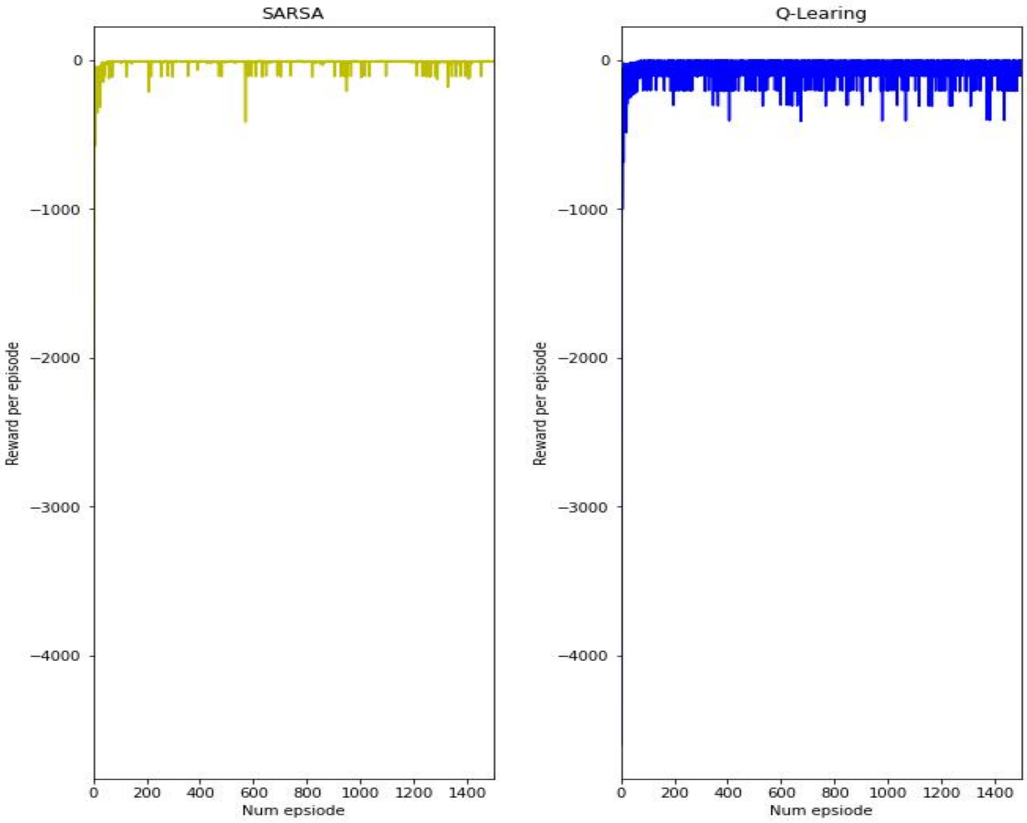

Figure 4: Sum of rewards obtained per episode, SARSA vs Q-learning with $\epsilon = 0.1$

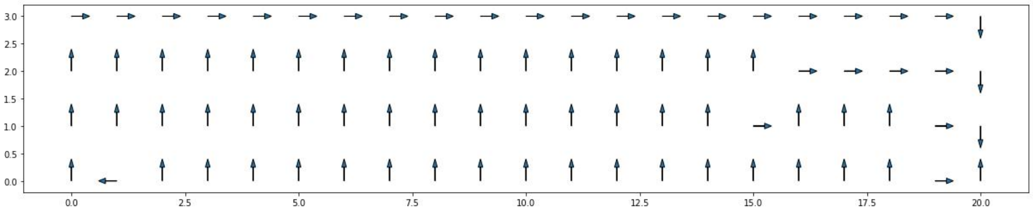

Figure 5: Learned policy for SARSA with $\epsilon = 0.3$

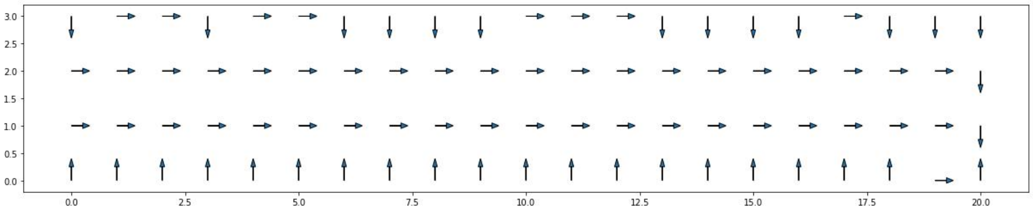

Figure 6: Learned policy for Q-learning with $\epsilon = 0.3$

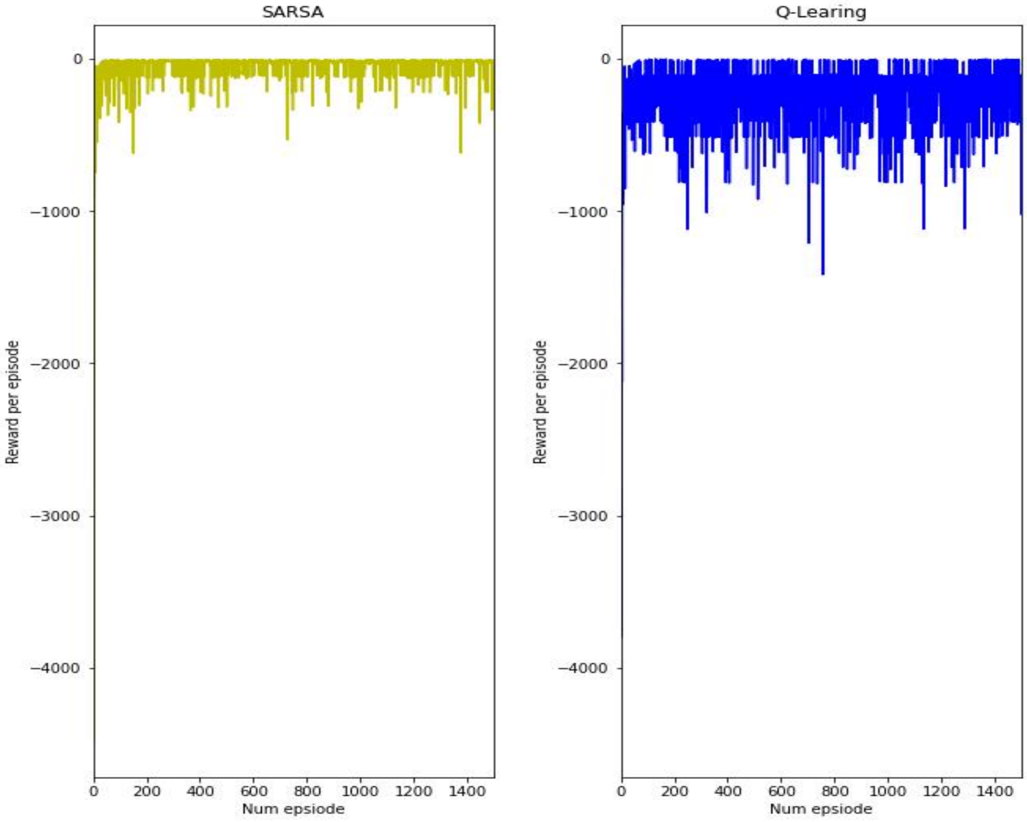

Figure 7: Sum of rewards obtained per episode, SARSA vs Q-learning with $\epsilon = 0.3$

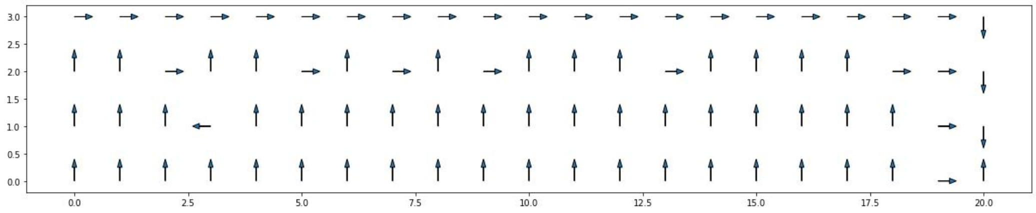

Figure 8: Learned policy for SARSA with $\epsilon = 0.6$

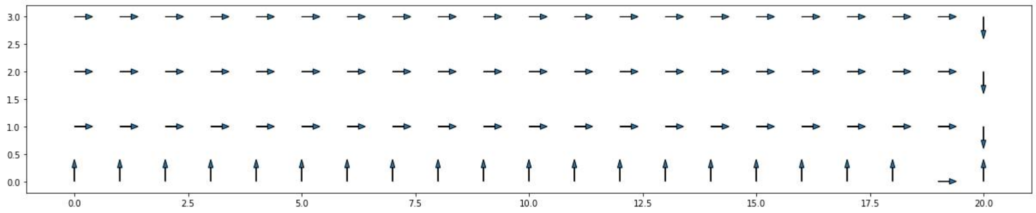

Figure 9: Learned policy for Q-learning with $\epsilon = 0.6$

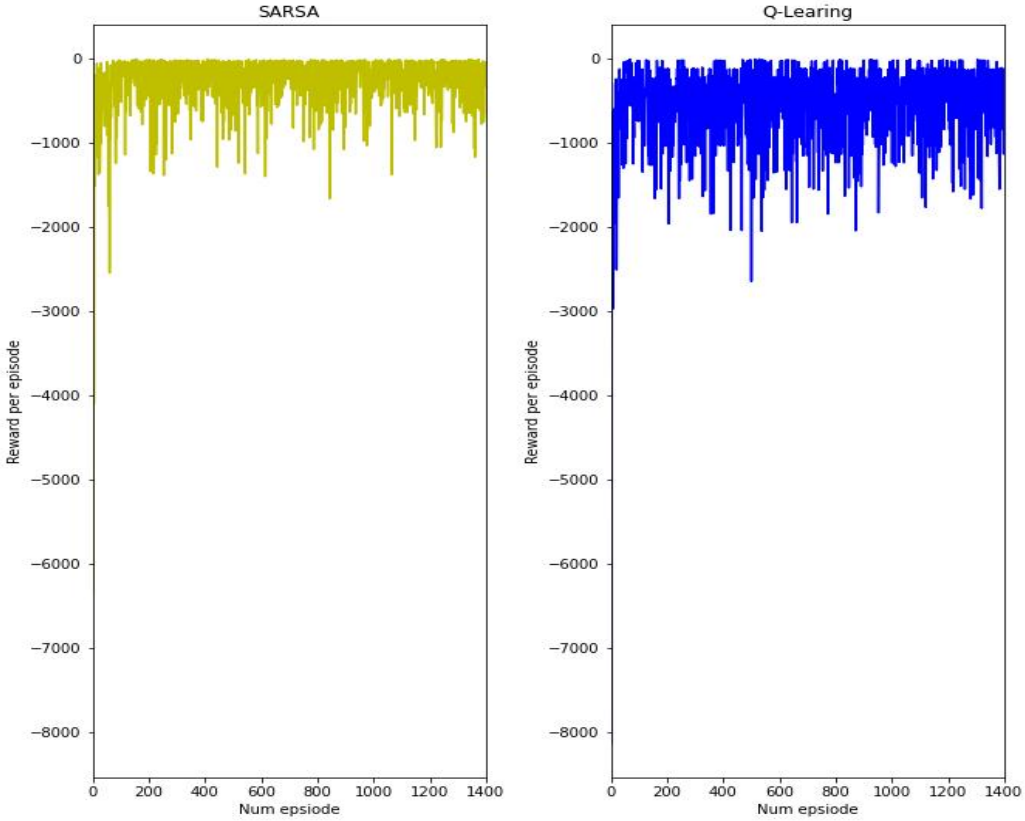

Figure 10: Sum of rewards obtained per episode, SARSA vs Q-learning with $\epsilon = 0.6$

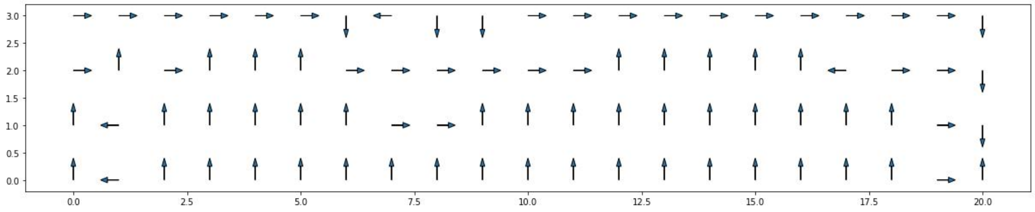

Figure 11: Learned policy for SARSA with snakepit obstacle included and $\epsilon = 0.1$

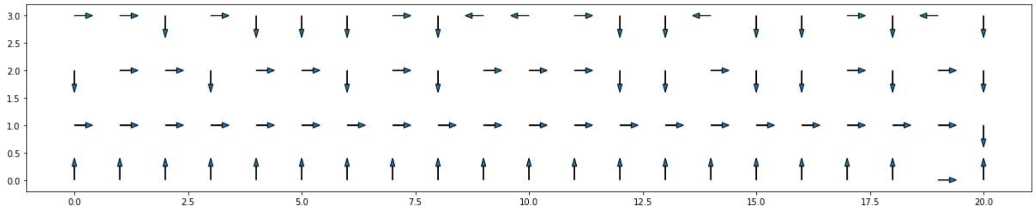

Figure 12: Learned policy for Q-learning with snakepit obstacle included and $\epsilon = 0.1$

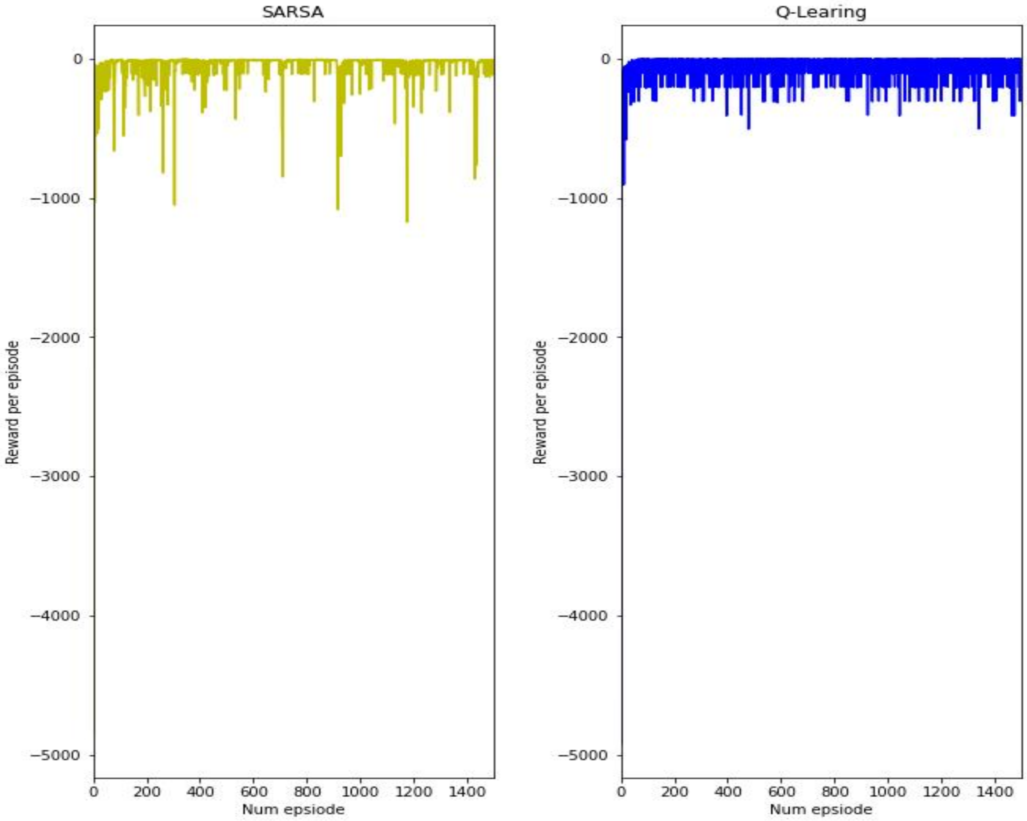

Figure 13: Sum of rewards obtained per episode, SARSA vs Q-learning with snakepit obstacle included and $\epsilon = 0.1$