csoc-Rl

Sebastian K Sabu

June 2025

1 Introduction

In this week we tried out 5 algorithm they are as follow

1. Monte Carlo learning
2. Temporal Difference Learning (Value state)
3. Q-learning
4. SARSA(state action reward state action) Learning
5. Double Q-Learning

These algorithms were used on the Frozen Lake map of size 100X100 which proved difficult for most algorithms to.Q-learning is the algorithm that found an optimal path out of these

2 Environment

I used a custom version of the Frozen Lake Environment with size 100,50,10 with an expanded action space of 8 which includes the original movement and diagonal movement. First I tried to implement this by simply passing in parameters like size but implementing action space was not possible so I went through the source code of the class Frozen Lake within in Library this gave me access to the generate map function which generates map of given size and that are valid. next was to implement the Action space which i did by overwriting the step() function adding Diagonal movements were done using max() to check for grid overflow. I was also able to implement the reward within the class but the transition probabilities for slippery = True is not implemented but for False is done but this doesn't show up much since we are using model free learning in most case and Q(s,a) is estimated mostly and the test I have done for simplicity is mostly with slippery Off we could do the test with similar effect by choosing the action with some randomness.

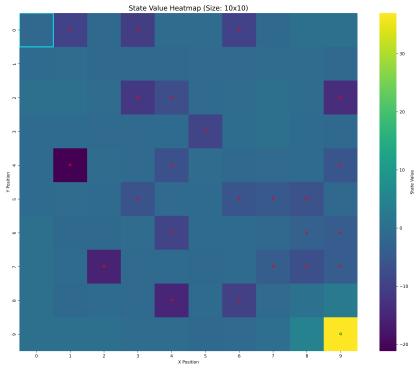

Figure 1: Value states

3 Q-learning

This was one of the most successful algorithm.It took around 400k episodes to train the 100x100 env it used and epsilon greedy for training this made sure the model had enough chances to explore.The best reward for this was Hole = -2 Goal = +10 Frozen = -1 Action that made the agent go out of the board was penalized with -0.01 This was done to ensure the actions don't get stuck in loops.The reasons for this can be seen in the V(s) grid at edges.

If you look closely at point (2,0),(2,1) we notice that both the squares have almost same expected return and but the one on the edge has slightly higher this leads to the agent jumping of the grid giving it better reward compared to going inside the map.This issue only arises in case of 100x100 for smaller maps the effect of reaching goal is carried to almost all cells therefore such valleys doesn't form the example I have used here is from TD(2) learning since q values cant be represented like this but the concepts remain almost the same

Convergence time for 10x10 = 2s

4 SARSA

In case of SARSA i was unable to converge for larger 100x100 but for smaller maps like 10x10 it was possible generally sarsa is slightly slower but in this begin a small map they were also identical we may be able to reduce the episode of the q learning making it even faster.But sarsa is generally used in case were overestimation bias of q-learning comes in.This is due to max() used in the q-learning which tends to build up error if the wrong action gets slightly more reward

by chance it will keep taking that action again and again for a while but eventually it will converge. Sarsa also takes the most cautions path generally suboptimal. Its and on policy model the effects of which I will explain in the cliff walking experiment.

Convergence time for 10x10 = 2s

5 Double-q-learning

This models choose between two Q(s,a) this reduces the overestimation bias that comes when using 1 q values.The final action can be decided by taking sum of Q - values or by randomly switching between them.This model also performed well in 10x10 and 50x50

convergence time for 10x10 = 2s

6 Monte Carlo

This was the most ineffective algorithm out of them all since it went all the way till the end of an episodes and also was unable to converge even after 1 million episodes. This is due to many factors.First the reward was simple 1 for goal,0 for frozen,0 for holes.This resulted in most V values to stay 0 since only the ones close to the goal had any values in them.Then I tried adding -1 for goals.This was also futile since almost all the case ended in hole the overwhelming number of episodes that ended in goals and the long steps made the effect of reaching the goals almost insignificant.My next idea was to add a potential based reward to make the agent move forward closer to the goal.This for some reason didn't seem to work I need to check more into this but since in this case the model doesn't learn in most case unless end of episode the episodes are mostly inefficient.

7 Bonus Task - Cliff Walking

I used two algorithm in this that are

1. Q-learning
2. SARSA-learning

7.1 Cliff learning Environment

A simple 4x12 map with cliff on the bottom row.Map stored as String array with G:ground,H:hole,E:end with simple up,left,down,right movement. Reward was -1 for G and S(start) , -100 for H,+10 for E.Both

the model was trained using epsilon greedy policy.Q-learning produced the optimal actions for the optimal path which if follwed by greedy policy. On the other hand SARSA produced the optimal action for the policy used in the training which is epsilon greedy.Therefore when testing we can see both the models get perfect score when using greedy policy for testing but in case of SARSA it is no the optimal policy since it was not trained on greedy policy.But when testing the model on epsilon greedy policy the sarsa has a success rate of almost 90 % while Q-learning has around 50 - 70 % chance of success.This all boils down to the fact that Q learning is Off policy and SARSA is On policy.

8 Conclusion

Q-learning,double-Q-learning and SARSA seems to be the best options out of them monte carlo is good for small maps.