COL 778 A2 Report

Kartik Krishna 2022AIZ8598, Samanyu Mahajan 2019CS50446

March 1st 2024

1 Part A

1.1 Tabular Q learning

We implement Tabular Q-learning according to the algorithm as shown in Figure-1.

We use multiple exploration vs exploitation strategies and we finally use $\epsilon = 0.2$ and conduct experiments for both large environment and small environment.

1.2 Small Environment

For small environment, we perform the Q-learning for 1000 episodes with $\alpha = 0.3$, here α is the learning rate and we set γ to 1. 1.

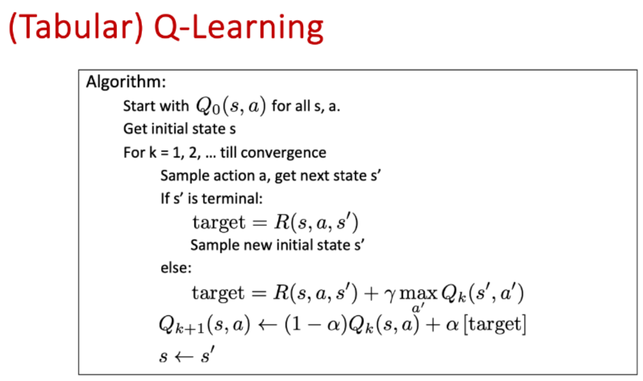
Figure-2 shows the optimal policy for the small environment. Because of the lower ϵ values a lot of states are unexplored but we reach the goal state quickly and hence we get a high reward quickly. Figure-3 shows the plot of the initial state vs the episodes plot, We can see that initially because of exploration, the start state has negative values but as soon as the learning algorithm reward at the goal state, the value of initial state quickly reaches the optimal value.

1.3 Large Environment

For large environment, we perform Q-learning for 5000 episodes with same hyperparameters as we had in the small environment. Here the policy obtained may not be completely optimal because of a lot of the states are not visited during the learning stage. Figure-4 shows the converged optimal policy and Figure-5 shows the intial state vs the reward plot. Here running the policy for larger number of iterations can result in better estimation of other states too because Q-values of other states would be estimated too.

1.4 Comparison with Value iteration

In value iteration we are in a deterministic world where the exact model of the MDP is known and since there is no exploration involved, we get value estimates of all the states and hence it can converge to better policy but in Q-learning



(Tabular) Q-Learning

```
Algorithm:
    Start with $Q_0(s,a)$ for all s, a.
    Get initial state s
    For k = 1, 2, … till convergence
        Sample action a, get next state s'
        If s' is terminal:
            $\text{target} = R(s,a,s')$
            Sample new initial state s'
        else:
            $\text{target} = R(s,a,s') + \gamma \max_{a'} Q_k(s',a')$
        $Q_{k+1}(s,a) \leftarrow (1-\alpha)Q_k(s,a) + \alpha\,[\text{target}]$
        $s \leftarrow s'$
```

Figure 1: Vannila Value iteration for small map heat map and directions

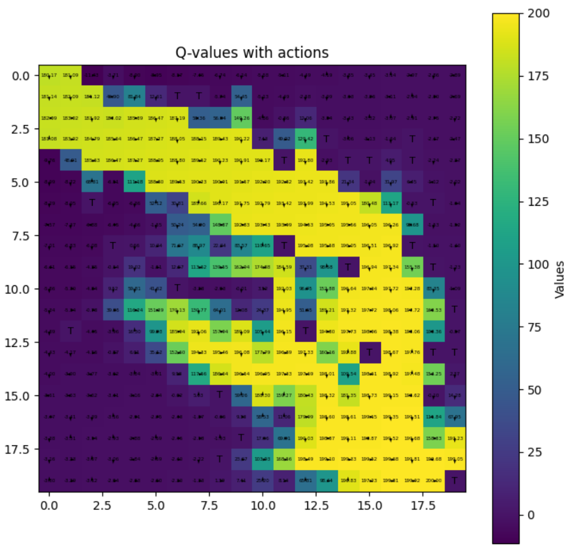


Figure 2: Optimal Policy Small environment

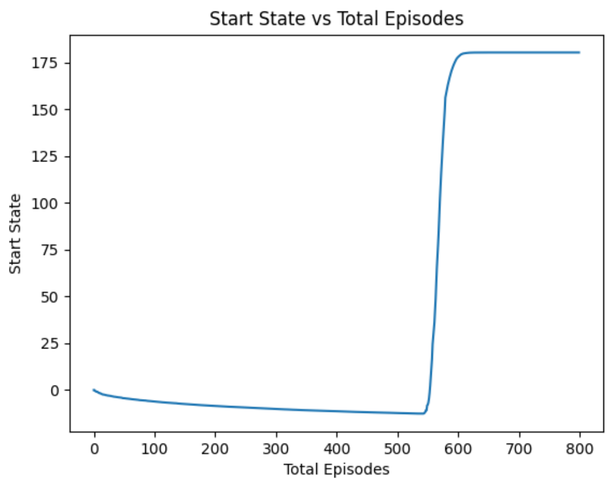


Figure 3: Start State vs Episodes Small environment

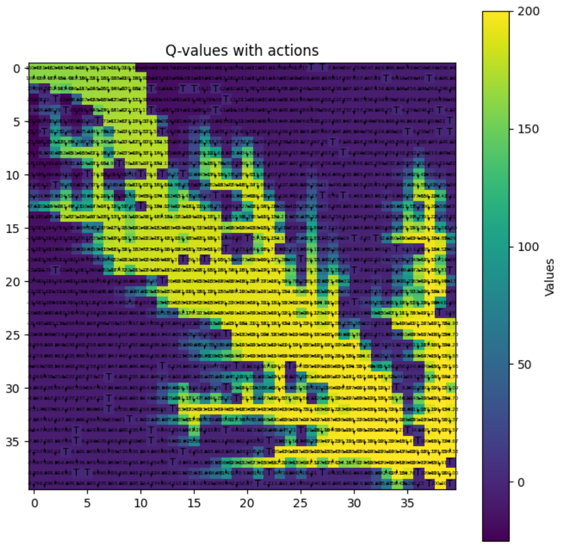


Figure 4: Optimal Policy Small environment

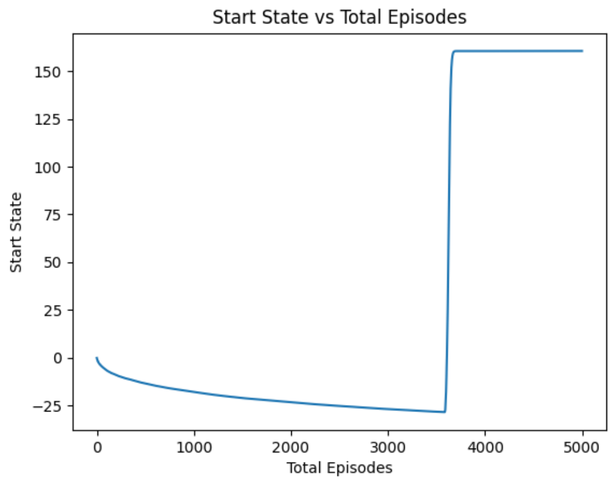


Figure 5: Start State vs Episodes Large environment

since the underlying model of the MDP is not known, it may not always lead to the exactly optimal policy but with more number of iterations it can get there.

2 Deep Q Learning

In deep Q learning approach instead of maintaining a table, we learn a function which would give us an estimate of the Q values of all the states. Here we employ a neural network to learn the Q values. The architecture of the neural network is that it has two layers which takes an input hidden state and gives the final Q-values for all the actions for a given state.

Here we also maintain a replay buffer to replay the experiences from the past and we use smooth L1 loss, with epsilon decay. We also employ a batch size of 128 and run the DQN for 800 episodes We use one hot encoding because it gives more information about the nature of the state in a grid map than simple raw integer.

The equation for epsilon decay is given by:

$$\epsilon = \epsilon_{\text{end}} + (\epsilon_{\text{start}} - \epsilon_{\text{end}}) \times e^{-\text{step\_decay}} \tag{1}$$

2.1 Small Environment

Figure-7 and Figure-6 shows the reward vs episodes plot and the start state value vs the episodes plot for the small environment. We can see that initially the estimates of the states are very bad and reward is very large and negative but subsequently the model learns to estimate the state better and hence the reward in the subsequent states saturates to the maximum discounted reward.

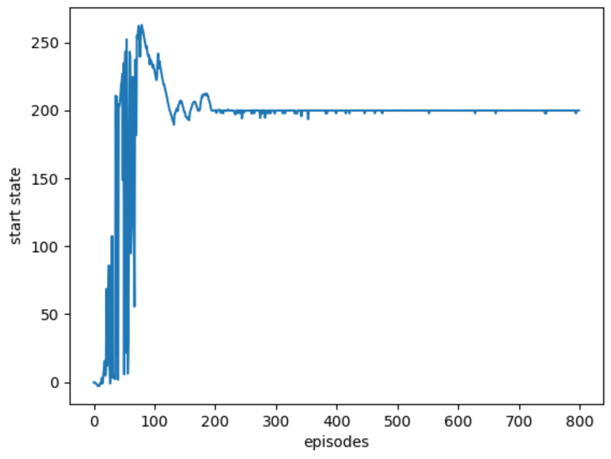


Figure 6: Start State vs Episodes small environment

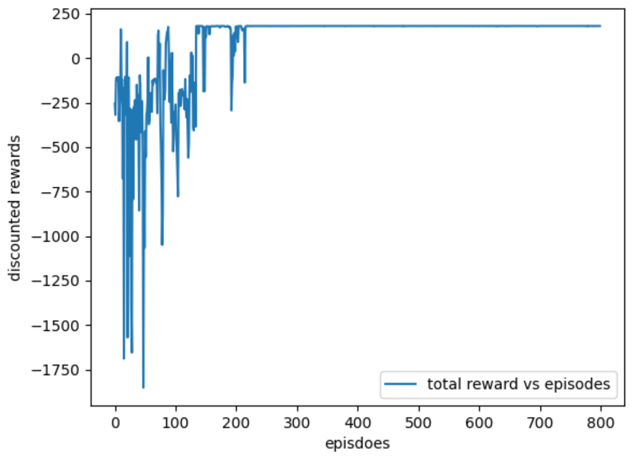


Figure 7: Rewards vs Episodes small environment

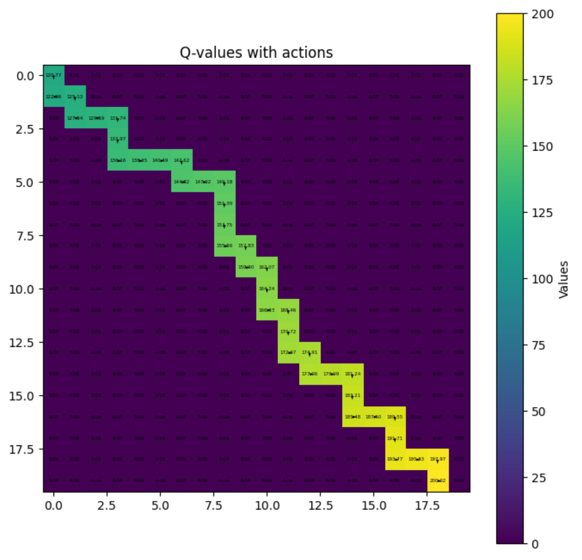


Figure 8: Q-values and optimal policy for DQN in small environment

Figure-8 shows the optimal policy map for DQN in the small environment, For the small environment my DQN takes 16 minutes to completely train to an optimal policy.

2.2 Large Environment

In the large environment our DQN takes 52 minutes to completely train to converge to the optimal policy. We see that from Figure-9 that initially since the environment is large, time taken to identify the goal state is longer and hence the model produces bad estimates of the start state which eventually with training converges to the right values. Similarly Figure-10 shows that initially we have a high negative reward but it converges to the optimal discounted reward eventually.

2.3 Comparison with tabular learning

Compared to tabular learning neural networks are data intensive hence take longer time to converge to the optimal Q-values but eventually the policy learnt by the neural network is comparatively more optimal.

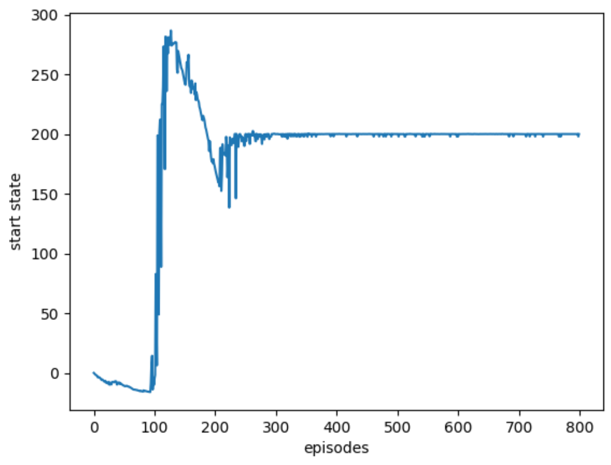


Figure 9: Start State vs Episodes small environment

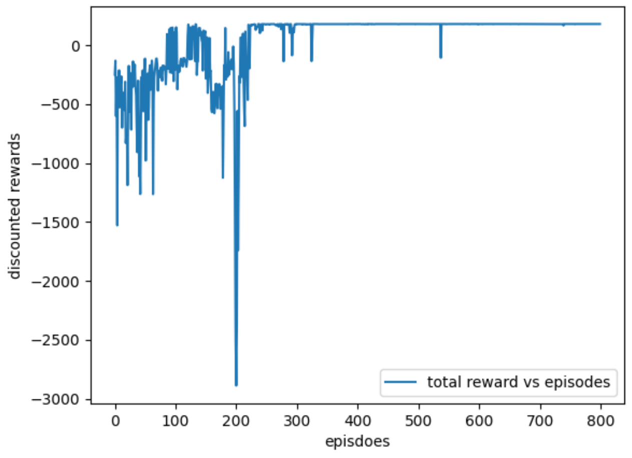


Figure 10: Start State vs Episodes small environment

3 Contributions

All the work was distributed equally