CS6700 : Reinforcement Learning
Programming Assignment 3
Report

Rahul V
ME16B171

July 31, 2020

1 SMDP Q-Learning

SMDP Q-Learning and Intra option Q-Learning are implemented on four-room gridworld. Both the algorithms are trained for 1000 episodes and the results are averaged over 50 independent runs for both goals G_1 and G_2 and for both starting conditions(random state in room-1 and fixed state in room-4). ϵ - greedy policy with an $\epsilon = 0.1$ and a $learning\_rate(\alpha)$ of 0.25 arae used while training in all runs.
Since for any state in a room, the agent can only choose from 4 primitive actions$(Up, Right, Down, Left)$ + 2 options available in that room, We can generalize the option space for each state into 6 options (4 primitive actions + 2 hallway options). In the Implementation, these options are numbered as (0-3) for primitive actions(in the order $\{Up, Right, Down, Left\}$), 4 for $Clockwise\,hallway\,option$(Option that leads to hallway which is in clockwise direction to the agent at the current state) and 5 for $AntiClockwise\,hallway\,option$

1.1 Learned Value functions with SMDP Q-Learning

The values of a state(V) or state-option pair(Q) are visualised by drawing a circle in the state grid. The value is proportional to the radius of the circle and is equal to 1 if the circle just touches the square.

1.1.1 Goal- G_1 with Random start state in room-1

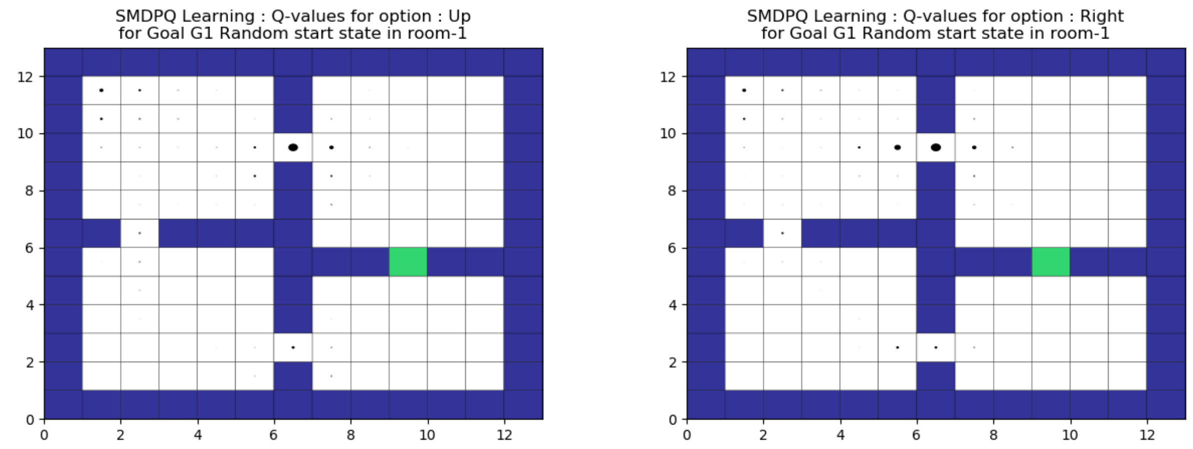


Figure 1: Q-Values for G_1 with random start state for option a)Up, b)Right.

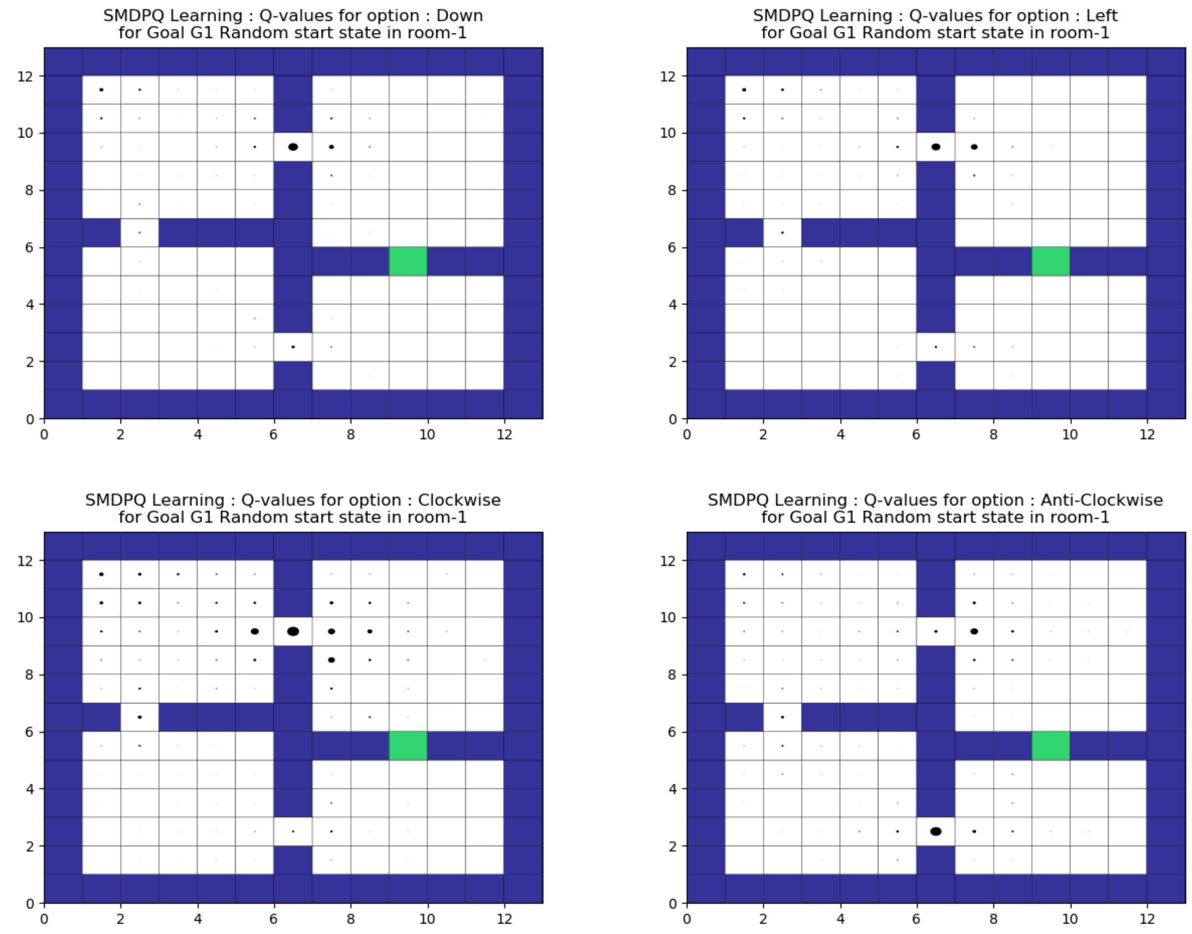


Figure 2: Q-Values for G_1 with random start state for option a)Down, b)Left, c)Clockwise, d)Anti Clockwise.

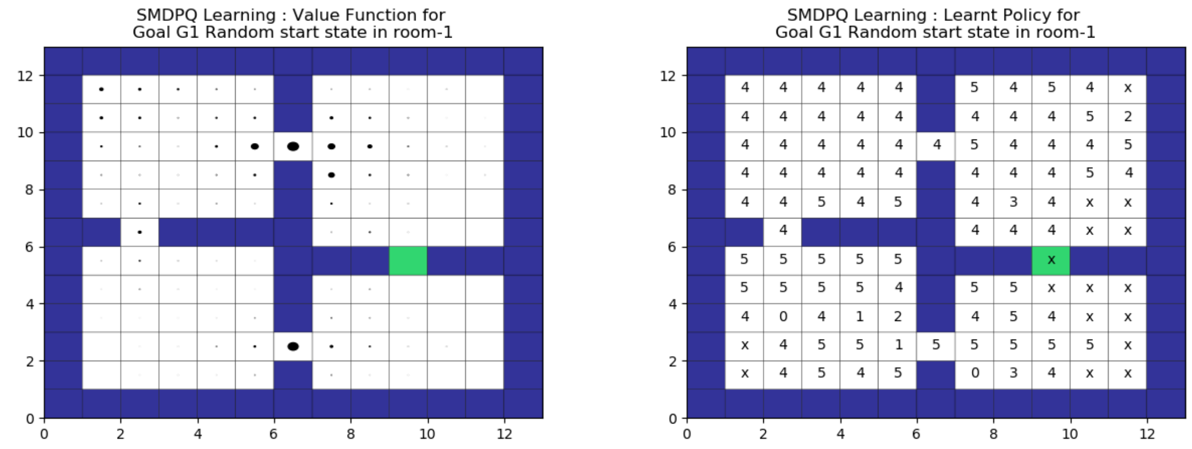


Figure 3: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_1 which is marked as green with random starting state

1.1.2 Goal- G_2 with Random start state in room-1

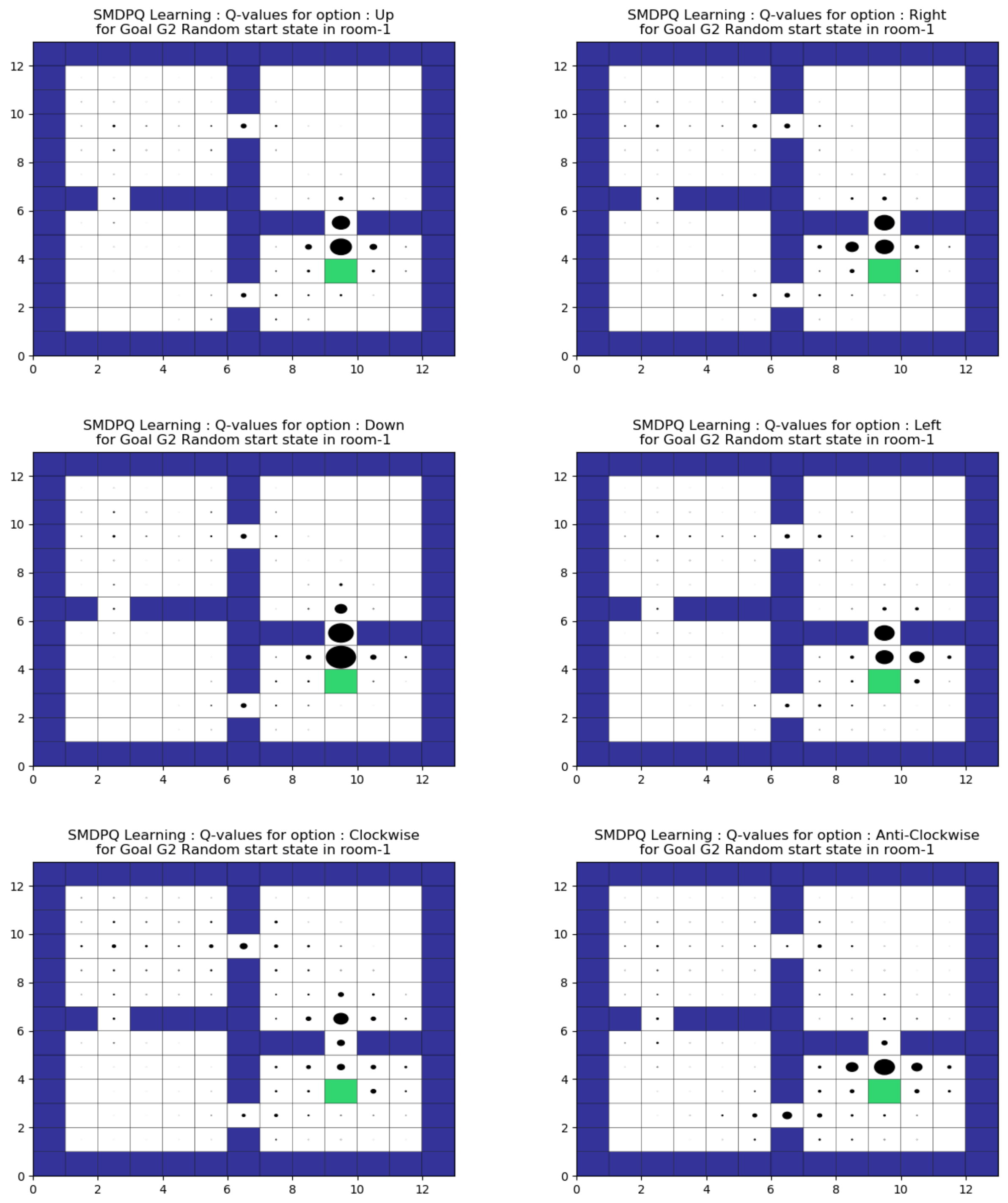


Figure 4: Q-Values for G_2 with random start state for option a)Up, b)Right, c)Down, d)Left, e)Clockwise, f)Anti Clockwise.

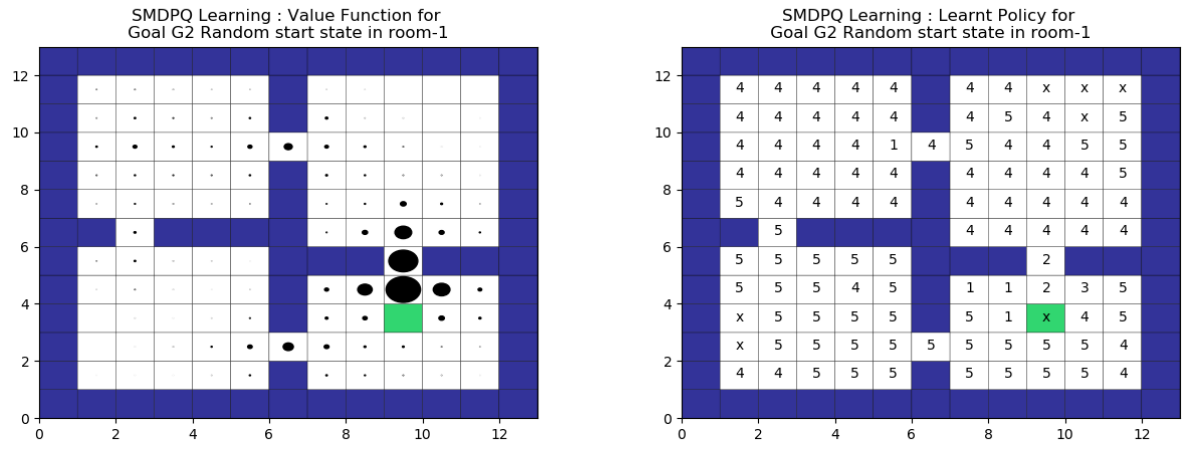


Figure 5: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_2 which is marked as green with random starting state

1.1.3 Goal- G_1 with Fixed start state in room-4

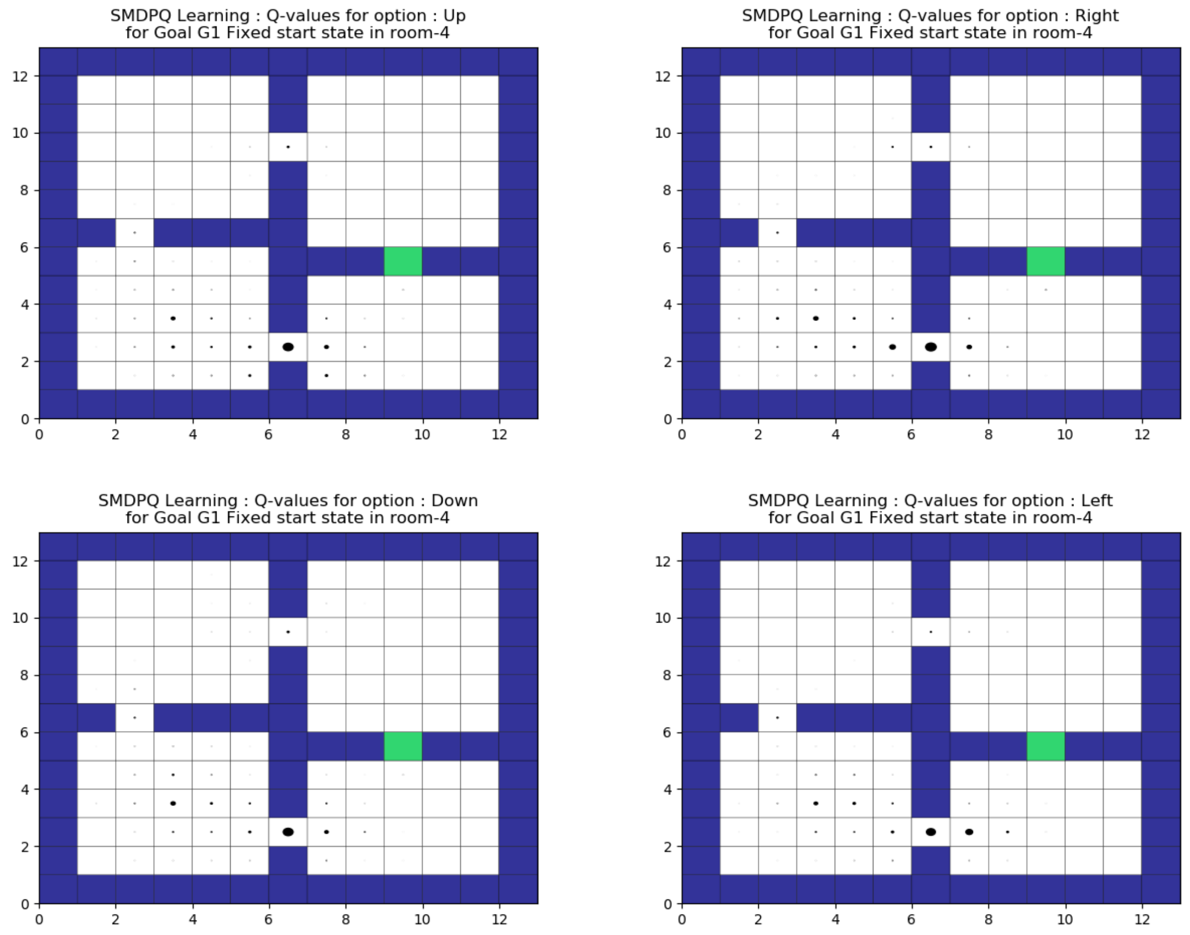


Figure 6: Q-Values for G_1 with fixed start state for option a)Up, b)Right, c)Down, d)Left

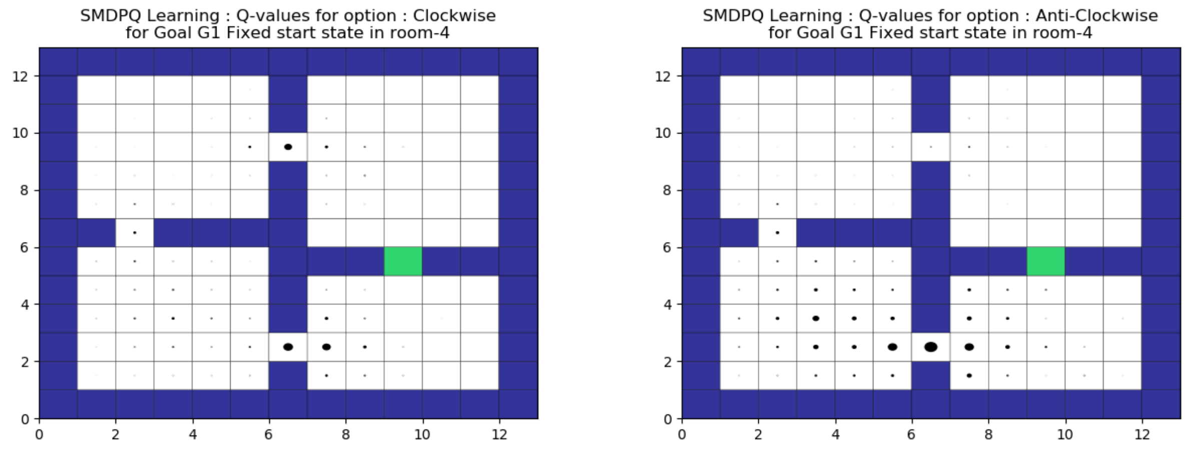
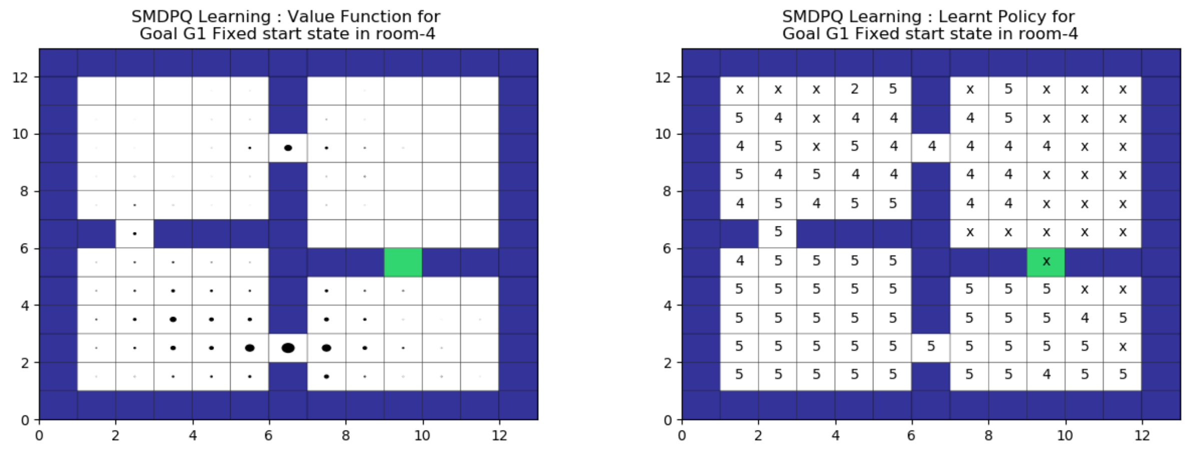


Figure 7: Q-Values for G_1 with fixed start state for option a)Clockwise, b)Anti Clockwise.



x	x	x	2	5		x	5	x	x	x
5	4	x	4	4		4	5	x	x	x
4	5	x	5	4	4	4	4	4	x	x
5	4	5	4	4		4	4	x	x	x
4	5	4	5	5		4	4	x	x	x
	5					x	x	x	x	x
4	5	5	5	5				x		
5	5	5	5	5		5	5	5	x	x
5	5	5	5	5		5	5	5	4	5
5	5	5	5	5	5	5	5	5	5	x
5	5	5	5	5		5	5	4	5	5

Figure 8: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_1 which is marked as green with fixed starting state.

1.1.4 Goal- G_2 with Fixed start state in room-4

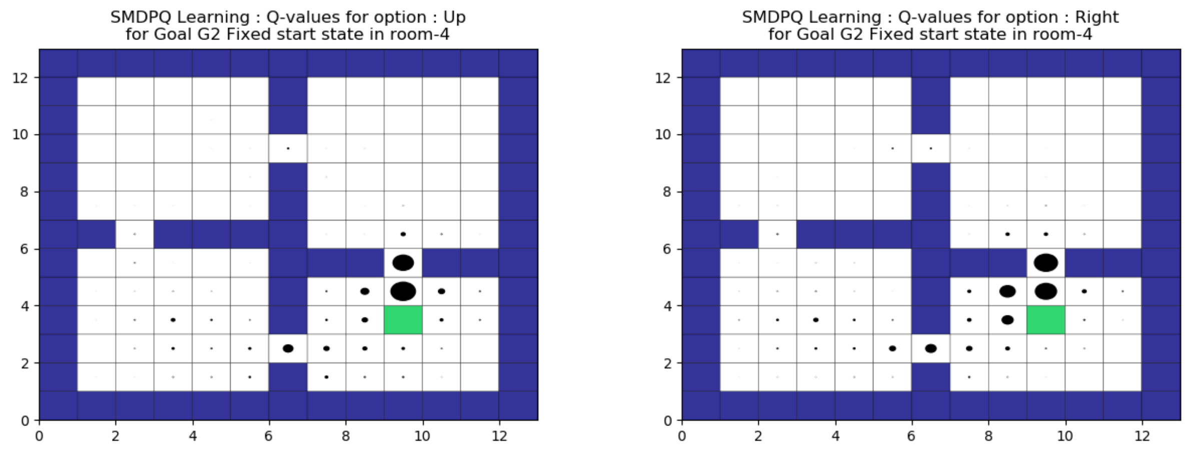


Figure 9: Q-Values for G_2 with fixed start state for option a)Up, b)Right.

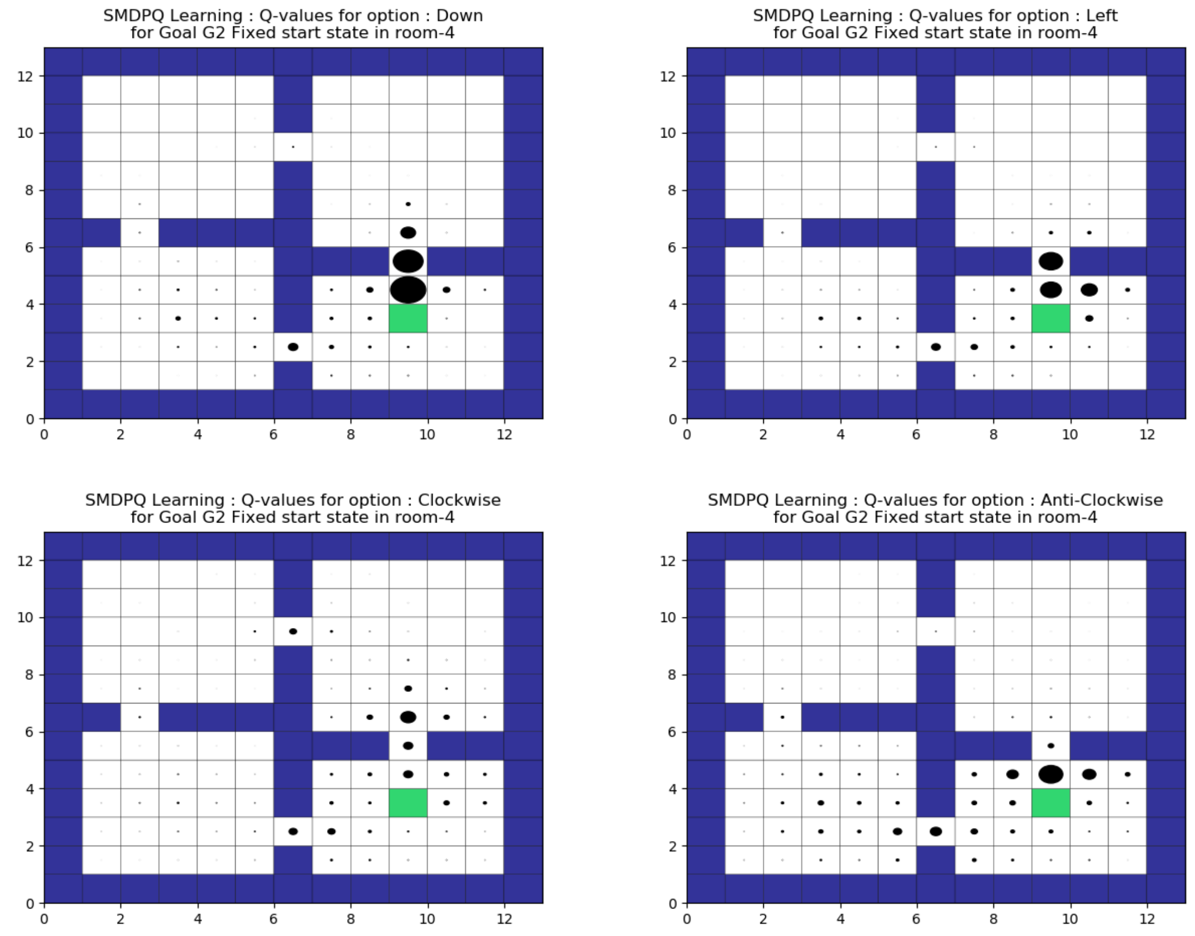
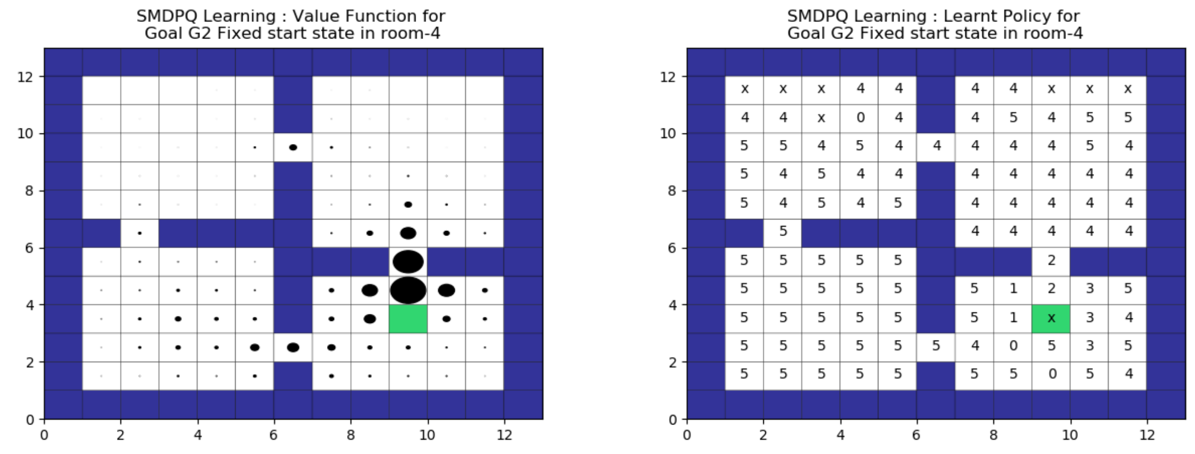


Figure 10: Q-Values for G_2 with fixed start state for option a)Down, b)Left, c)Clockwise, d)Anti Clockwise.



x	x	x	4	4		4	4	x	x	x
4	4	x	0	4		4	5	4	5	5
5	5	4	5	4	4	4	4	4	5	4
5	4	5	4	4		4	4	4	4	4
5	4	5	4	5		4	4	4	4	4
	5					4	4	4	4	4
5	5	5	5	5				2		
5	5	5	5	5		5	1	2	3	5
5	5	5	5	5		5	1	x	3	4
5	5	5	5	5	5	4	0	5	3	5
5	5	5	5	5		5	5	0	5	4

Figure 11: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_2 which is marked as green with fixed starting state

1.2 Learnt Value functions with Intra Option Q-Learning

1.2.1 Goal- G_1 with Random start state in room-1

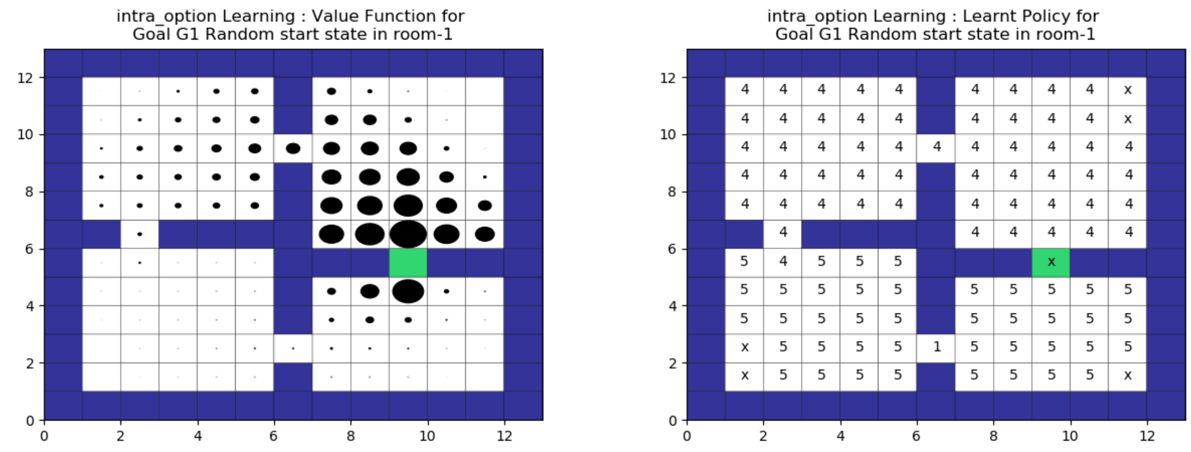


Figure 12: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_1 which is marked as green with random starting state

1.2.2 Goal- G_2 with Random start state in room-1

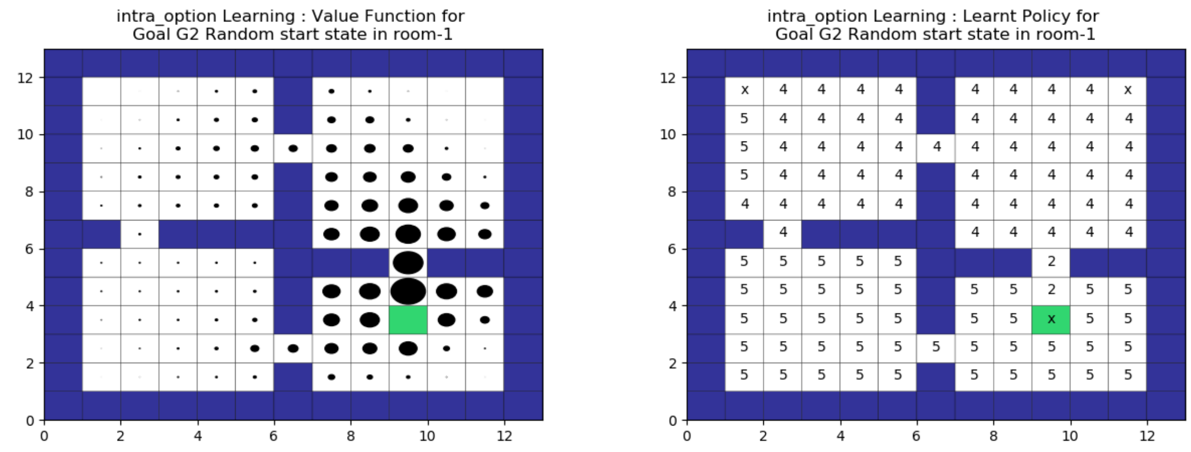


Figure 13: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_2 which is marked as green with random starting state

1.2.3 Goal- G_1 with Fixed start state in room-4

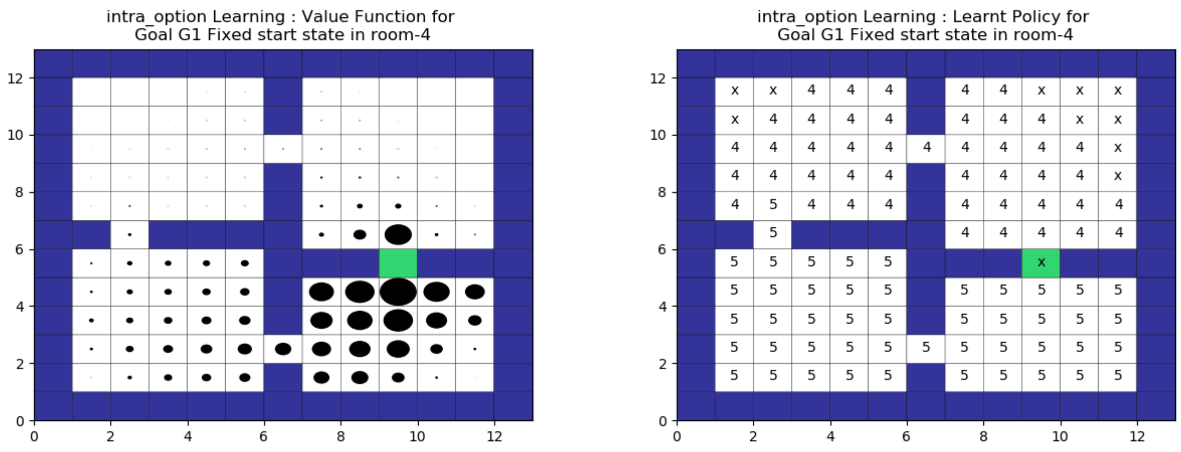


Figure 14: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_1 which is marked as green with fixed starting state

1.2.4 Goal- G_2 with Fixed start state in room-4

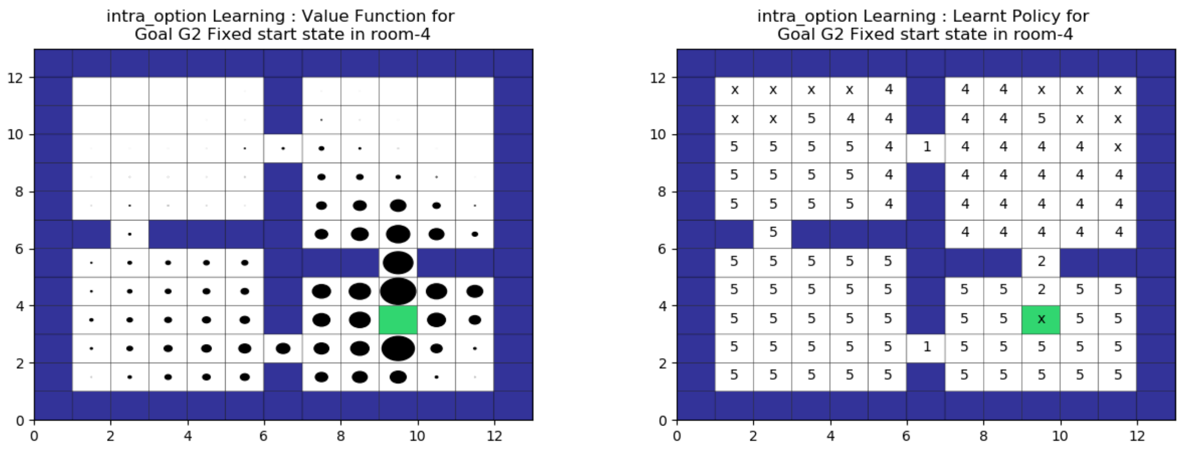
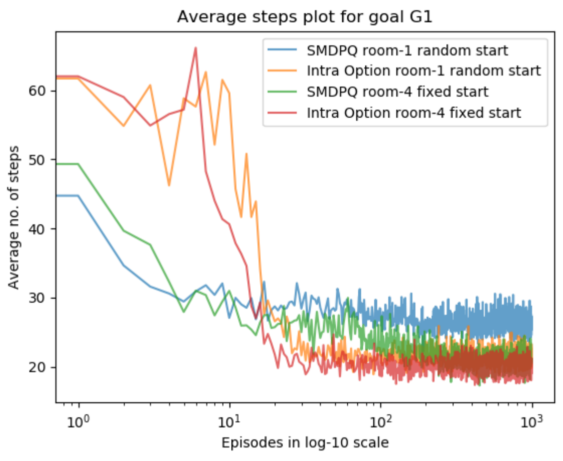


Figure 15: a)Value Function Plot b)Learnt Policy(x - not explored; (0-5) option as explained above) for Goal - G_2 which is marked as green with fixed starting state

1.3 Learning Curves and Observations



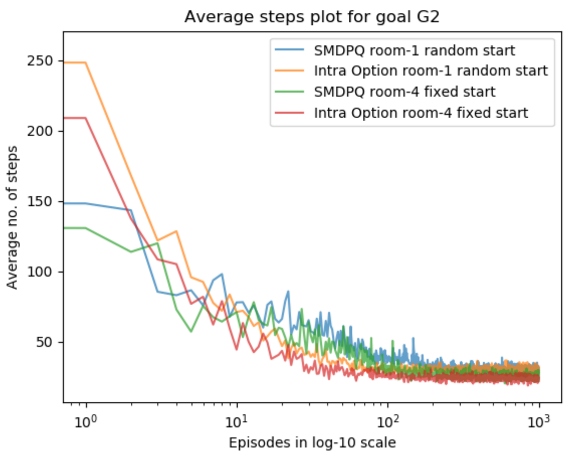


Figure 16: Plot Comparing Average steps taken by the agent with different starting conditions and learning algorithms for goal a)G_1 b)G_2 with epsiodes(x-axis) in log scale.

- In SMDP Q-Learning by changing the start state to fixed position in room-4, the number steps required for both the goals decreased on an average. This is because the start state in room-4 is nearer for G_2. And for G_1 it could be because random start state in room-1 could lead to near wall starting states that the agent might keep bumping into.
- For G_1 using SMDP Q-Learning other than the starting room all the other rooms are explored very little which is evident from the Q-values visualised. This is because the goal state is a hallway state which the agent can reach easily with the help of a hallway option.
- Even though G_2 is not a hallway state, the above analogy applies as only the starting state room and goal state room are explored to some extent. Since the G_2 is not a hallway state on an average the agent took longer to reach it.
- Using intra-option Q-Learning improved the agent's ability to learn the Q-values for more state-option pairs accurately compared to SMDP Q-Learning in the same number of episodes.
- From the average steps plot we can observe that Intra Option Q-Learning initialy(first 10 steps) takes more steps than SMDP Q-Learning for all goals and starting conditions as it explores the environment better. And after about 50 episodes it converges to its optimal behaviour which is only slightly better than SMDP Q-Learning's optimal behaviour but it learns the Q-values much better and faster than SMDP Q-Learning.

2 Deep RL

2.1 Implementation Details

Implementation of DQN algorithm was completed using the base code provided. The best set of hyperparameters obtained by trying out different sets is reported in the next subsection. Adam Optimizer is used to train the Q-network. For exploration ϵ - greedy policy is used with decaying ϵ over time to exploit the environment after training for sometime.

2.2 Results and Best Hyperparameter Values

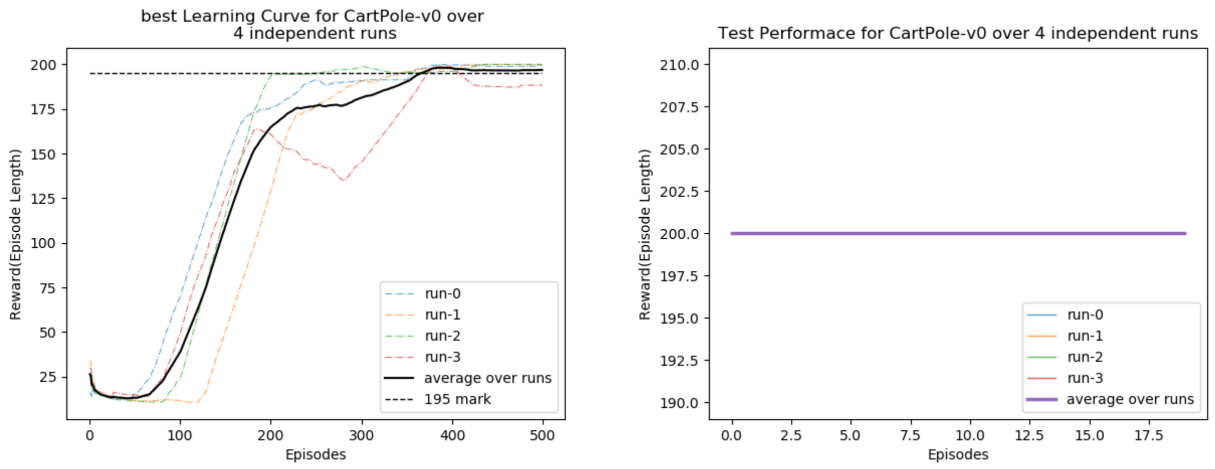


Figure 17: a)Learning Curve for DQN on CartPole-v0 environment averaged over 4 independent runs(different seeds) trained for 500 episodes. b) Test rewards obtained by the learnt agent over 20 episodes.

From the above Learning Curve we can say that on an average the agent "solved" CartPole-v0 with the implemented DQN algorithm for the set of hyperparameter values mentioned in the table below. The agent is also getting reward of 200 for all the episodes and over all runs while testing.

Hyperparameter	Value	Description
REPLAY_MEMORY_SIZE	10^5	number of tuples in experience replay
EPSILON_START	1	initial epsilon of epsilon-greedy exploration
EPSILON_DECAY	0.995	decay multiplier for epsilon
EPSILON_MIN	0.001	minimum value of epsilon
HIDDEN1_SIZE	20	size of hidden layer 1
HIDDEN2_SIZE	20	size of hidden layer 2
HIDDEN3_SIZE	20	size of hidden layer 3
EPISODES_NUM	500	number of episodes to train on.
MAX_STEPS	200	maximum number of steps in an episode
LEARNING_RATE	0.001	learning rate for SGD Optimizer
MINIBATCH_SIZE	32	size of minibatch sampled from the experience replay
DISCOUNT_FACTOR	0.999	MDP's gamma
TARGET_UPDATE_FREQ	100	number of steps (not episodes) after which to update the target networks

Table 1: Final set of Hyperparameters which worked best for the DQN.

2.3 Observations

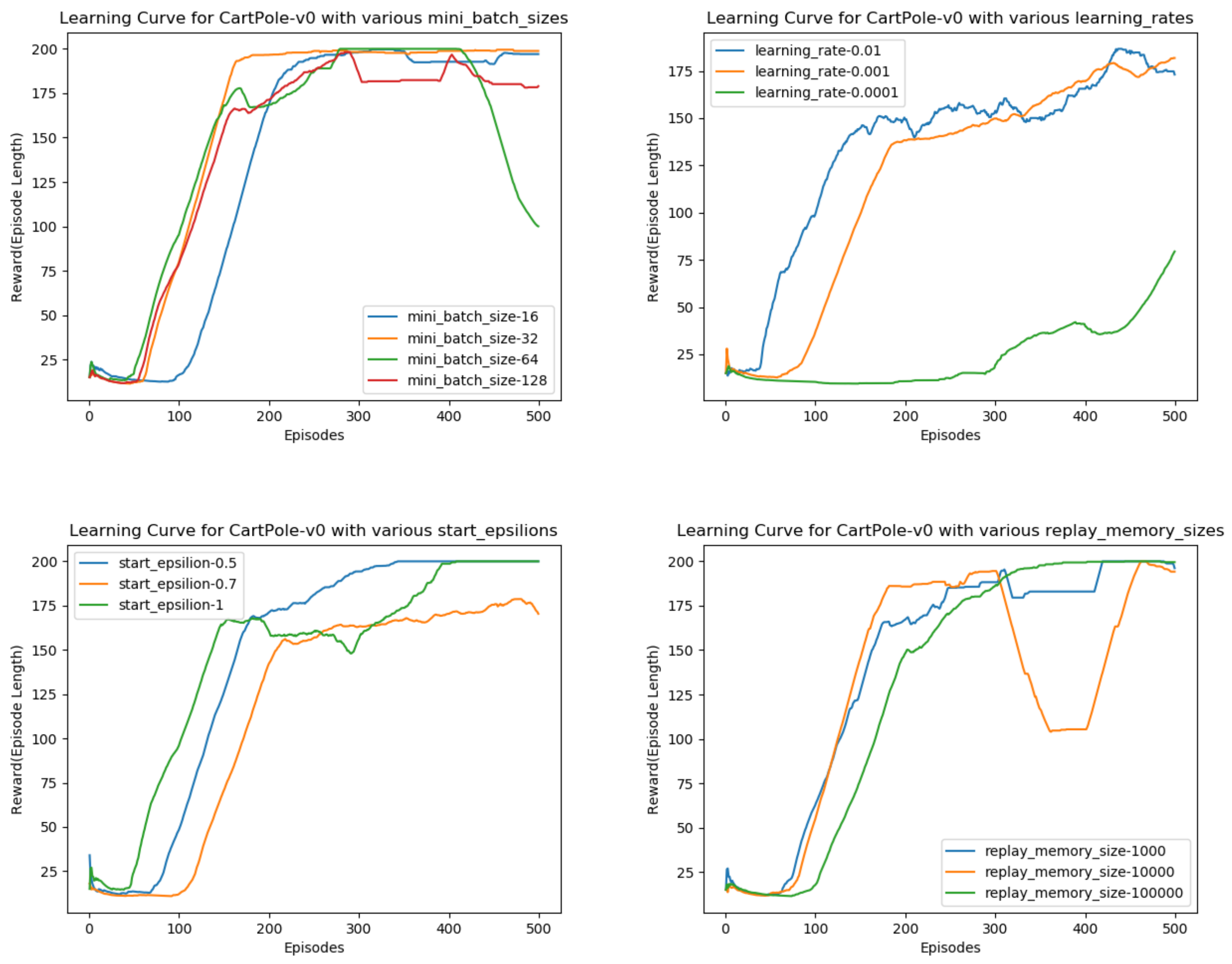


Figure 18: a)Learning Curve for DQN on CartPole-v0 environment trained for 500 episodes with varying a) Mini Batch Size, b) Learning Rate c) Starting Epsilon d) Replay Memory size

- From above figure (a)
- From above figure (b) we can see that high learning rate like 0.01 causes unstablity in the learning process and low learning rate like 0.0001 causes the agent to learn very slowly, So, a learning rate of 0.001 is chosen which from the graph seems to solve the problem.
- From above figure (c) we can see that higher starting epsilon is working better than lower starting epsilon. This is because higher starting epsilon allows the agent to explore more initially and learn the optimal policy which it can later exploit as the epsilon is decayed exponentially with a decay rate.
- From above figure (d) We can see that higher replay memory sizes are more stable as replay memory with less size will cause the sampled mini batch to contain correlated and bad tuples which hinders the learning process.
- Lower epsilon decay multiplier will result in sub optimal policies as the exploration stops earlier and the agent hasn't explored enough.

2.4 Removal of Target Network and Experience Replay

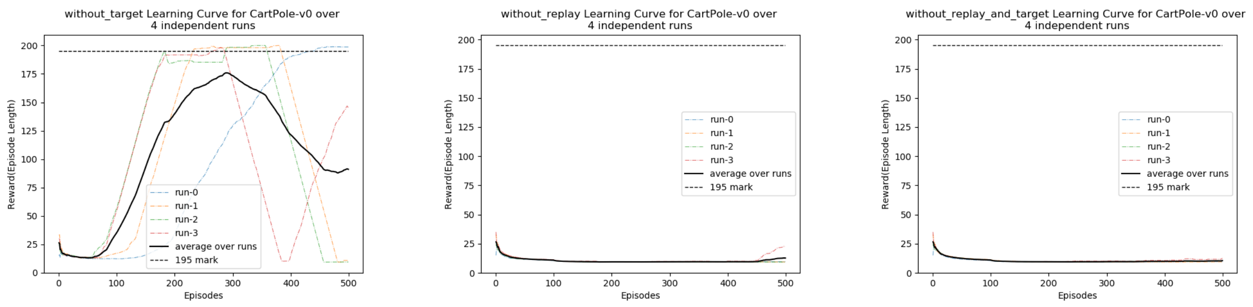


Figure 19: Learning Curve for DQN on CartPole-v0 environment a)without target network b) without replay memory, c) without both replay memory and target network averaged over 4 independent runs trained for 500 episodes.

From the above plots we can see that the agent doesn't learn anything at all when trained without replay memory for the fixed hyperparameters might be because all the samples that it encountered are bad and highly correlated and it can only learn from one sample at a time. So, the gradients might be too small for it to learn anything.
In the case where only target network is removed, the learning process is highly unstable as the agent is chasing non-stationary targets. So, we can see a lot of sharp ups and downs in the reward curve. This is called as "Catastrophic Forgetting", which means the agent which was doing well till now suddenly starts performing bad and the rewards reduce steeply.
From this we can see that Replay memory is much more useful compared to Target network.