Lab3

Yifan Ding

Environment A

What has the agent learned after the first 10 episodes ?

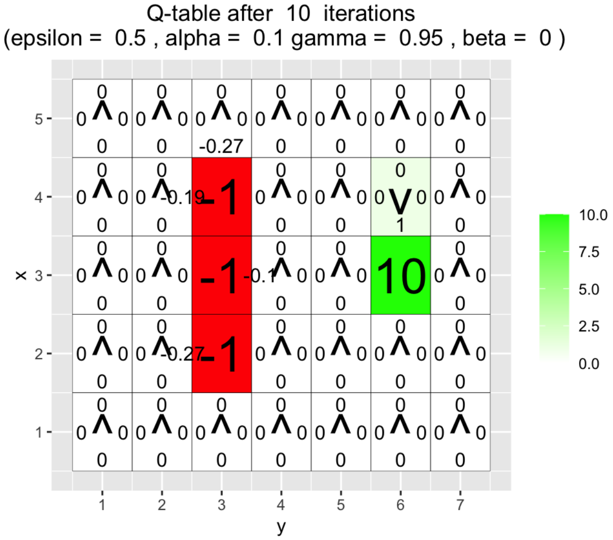


Figure 1: Q-table 10 iterarion with position (3, 1)

Answer: As in figure 1, the agent learned a few transitions, such as when agent is above on 10, then move to 10 (positive q-value), or when agent close to -1s, agent would avoid to move to -1s (negative q-value).

Is the final greedy policy (after 10000 episodes) optimal for all states, i.e. not only for the initial state ? Why / Why not ?

Answer: As in figure 2, it is not optima for all states. In state (1, 3), the best policy is go right and then up to the final state with reward 10. But the agent will choose to go left and then walk a long way to reach the final state. This is because the initial position of agent is (3,1), then the agent can only have a small possibility to walk to (1,3), and therefore the Q-table at state (1, 3) can not be well optimized. This can be found in figure 3 below, we get the best optima in (1, 3), but also create other sub-optima in the up-left corner.

Do the learned values in the Q-table reflect the fact that there are multiple paths (above and below the negative rewards) to get to the positive reward ? If not, what could be done to make it happen ?

Answer: No, since the agent will choose the action with the largest value in Q-table, so there is only one path to go. The agent will slip to other position if we do not use a deterministic transition_model, that is, give beta a probability between (0, 1) instead of 0.

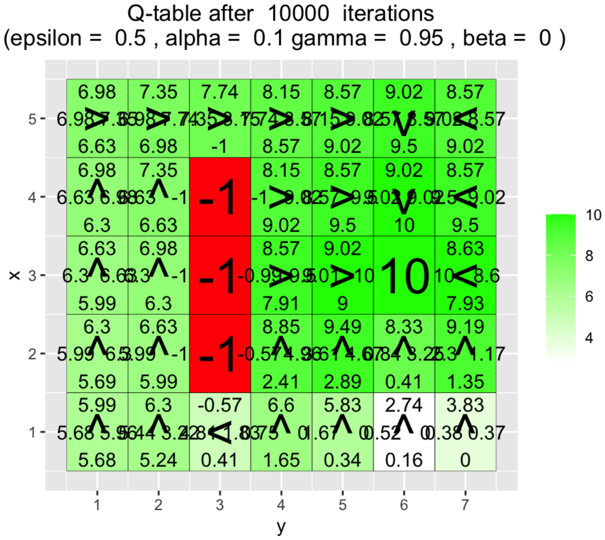


Figure 2: Q-table 10000 iterarion start with position (3, 1) (Default setting)

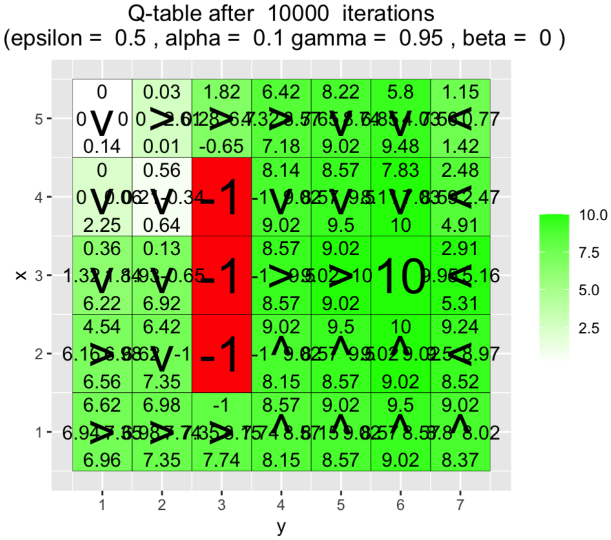
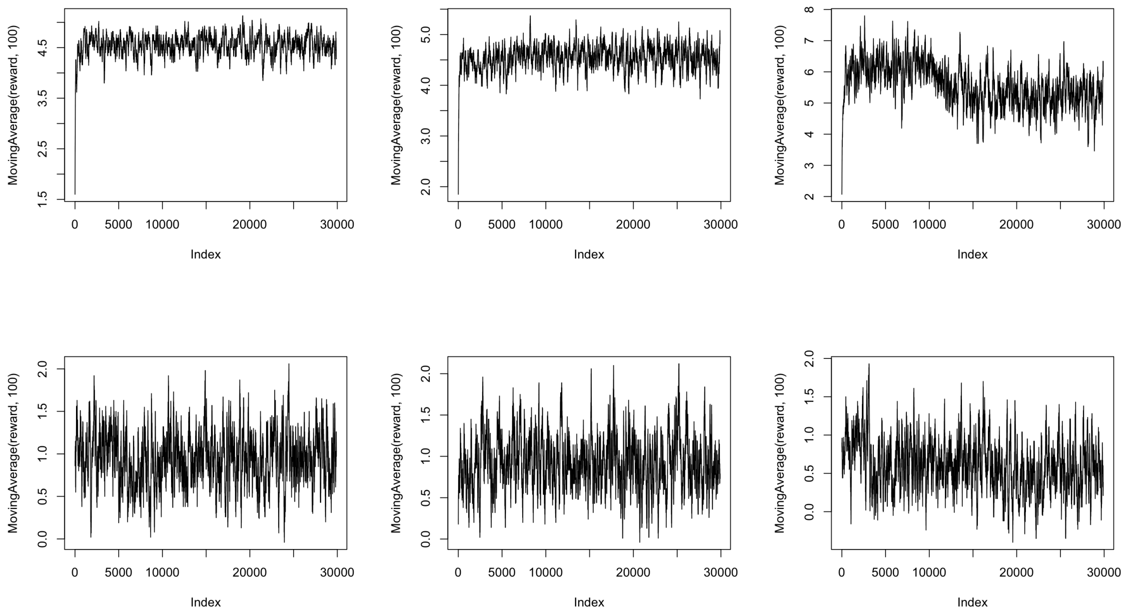


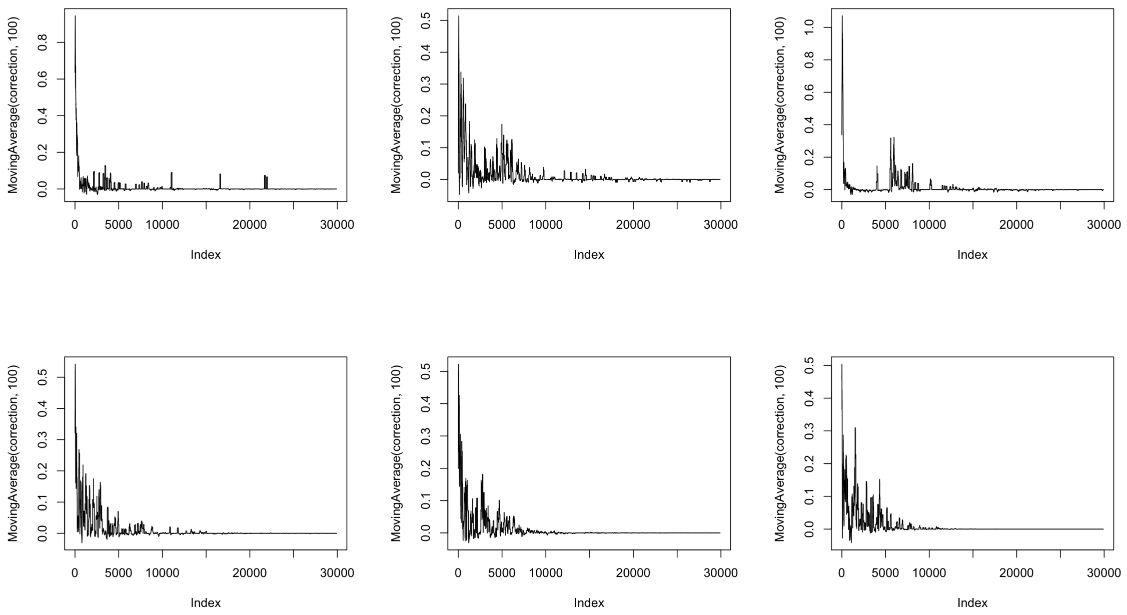
Figure 3: Q-table 10000 iterarion start with position (1, 3)

Environment B

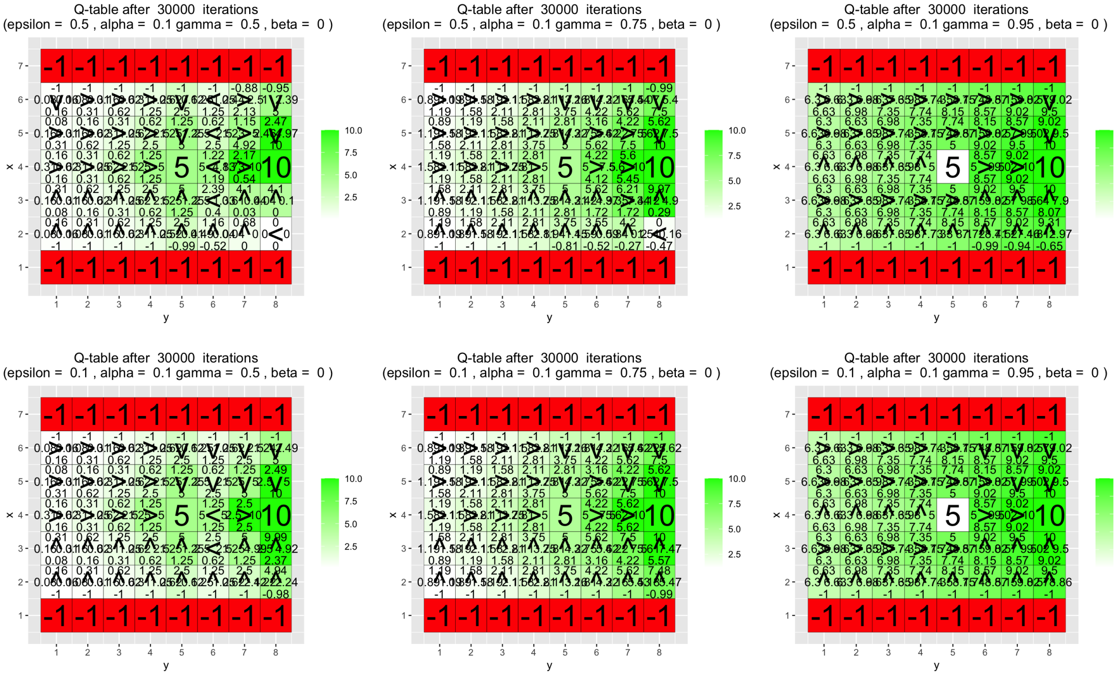
Moving average of reward with $\epsilon = 0.5, 0.1$ and $\gamma = 0.5, 0.75, 0.95$ respectively.



Moving average of correction with $\epsilon = 0.5, 0.1$ and $\gamma = 0.5, 0.75, 0.95$ respectively.



Q-Table with $\epsilon = 0.5, 0.1$ and $\gamma = 0.5, 0.75, 0.95$ respectively.



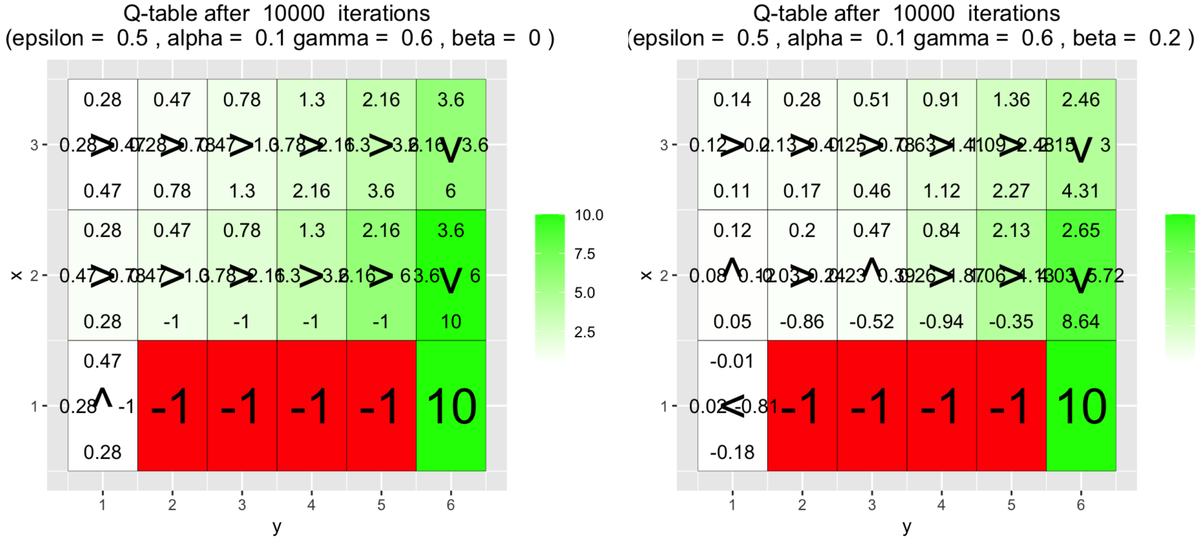
Explain your observations.

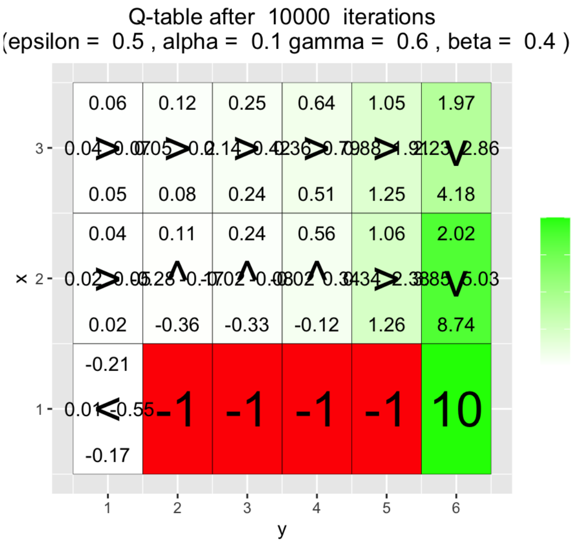
Answer:

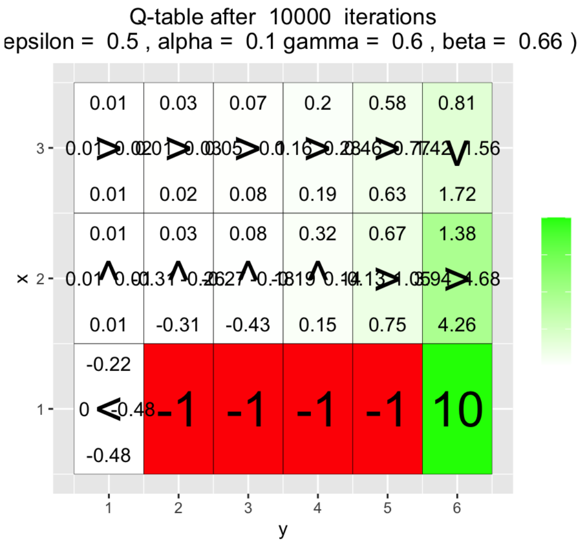
ϵ is the probability of acting greedily, therefore when ϵ is smaller the more possible the agent will randomly explore more rather than follow the temporal best policy (Q-Table), we can also see this from Q-table above, plots with smaller ϵ have more green area. Since the smaller ϵ explore more, therefore the moving average of reward is also smaller.

γ is the discount factor, higher γ means we hope the agent get more reward from future (long-term) instead of reward recent (short term), so as γ increasing the agent prefer to move to 10 and leave a white block in the 5 position.

Environment C







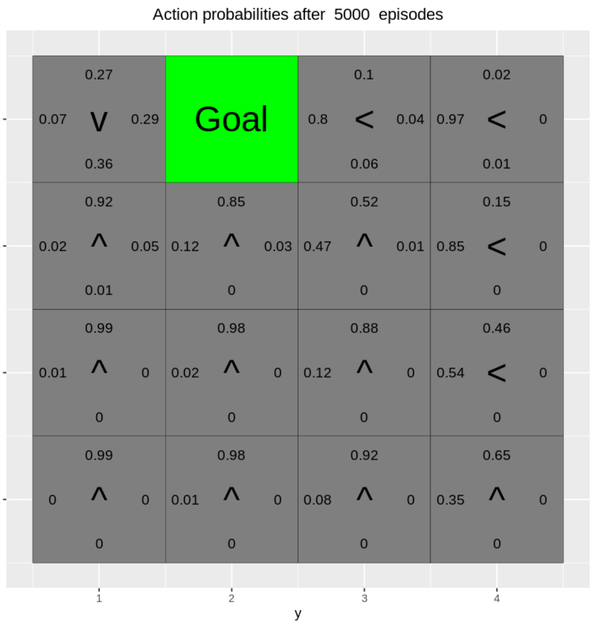
Explain your observations.

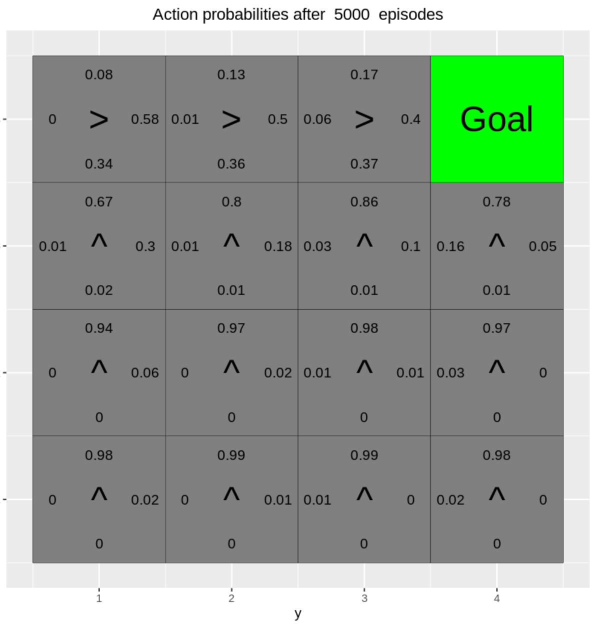
Answer: β is the uncertainty factor of transition model, the probability of actual action given an action, $\beta = 0$ is a deterministic model, the actual action is the given action. With β increasing, we can see the Q-table changes especially in the second row, from right arrows to up arrows. Because of the uncertainty, agent might move to -1 areas (slip right) thus get a negative reward. So agent choose a safety way to go (go up and right then go down to avoid to hit -1 areas).

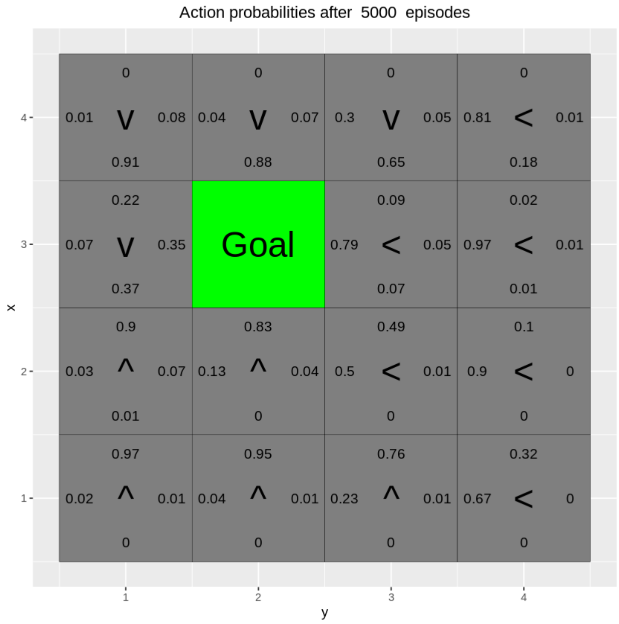
Environment D

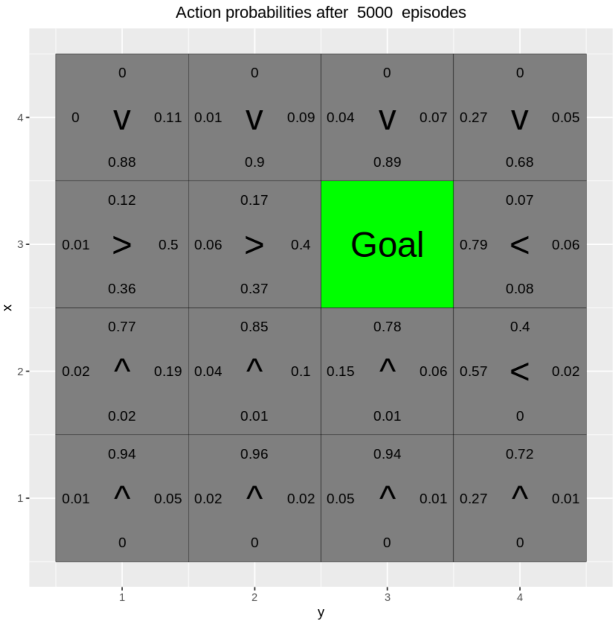
Has the agent learned a good policy? Why / Why not ?

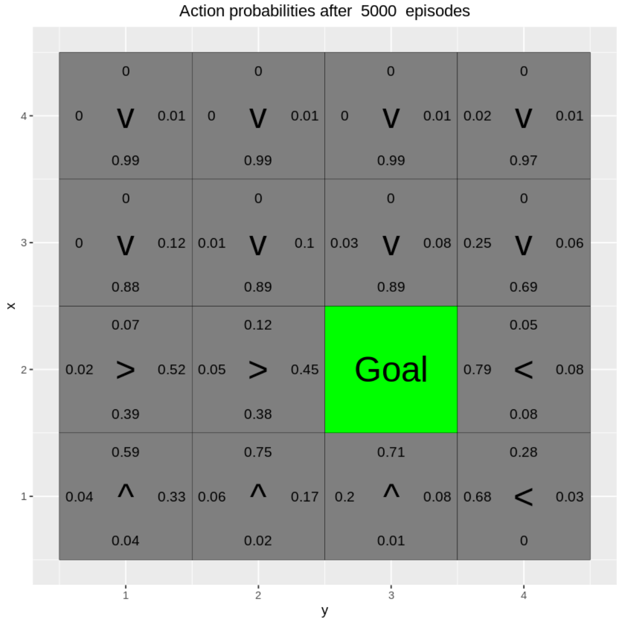
Answer: We think it is an ok policy, check the results after 5000 epoch training:

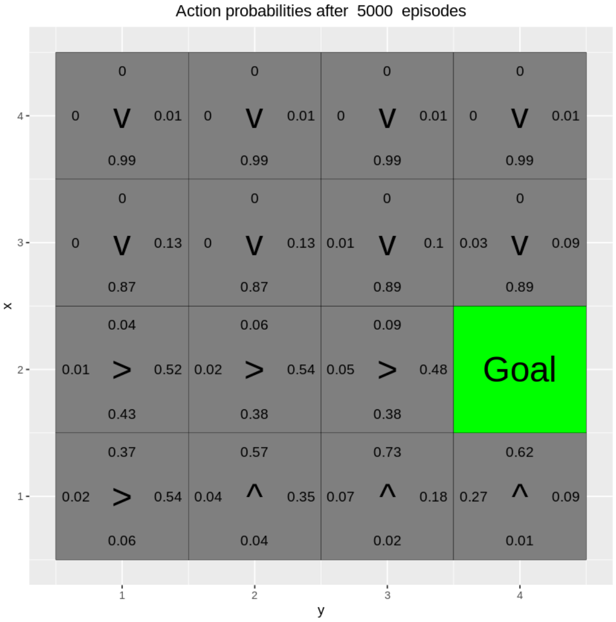


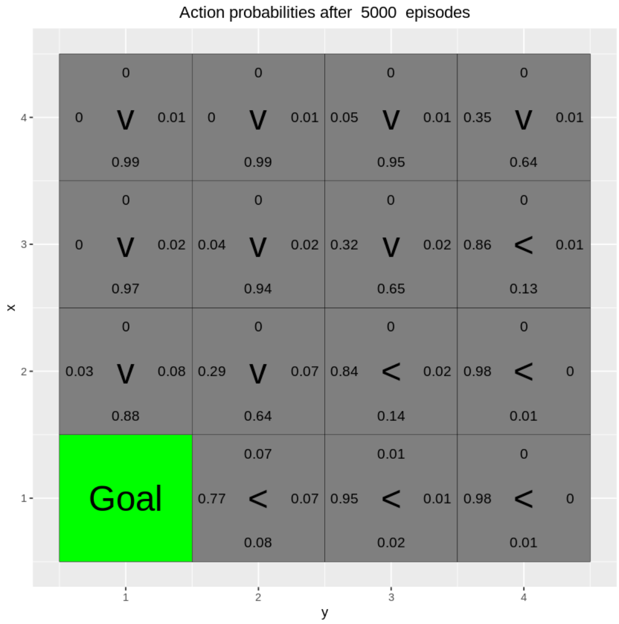


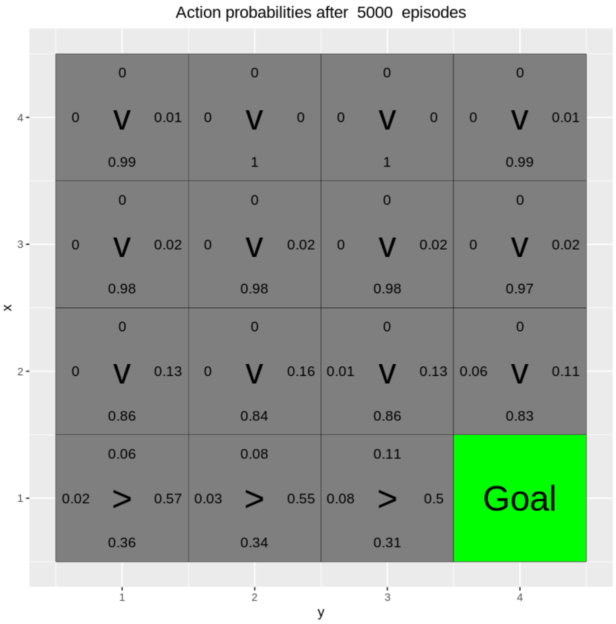

Action probabilities after 5000 episodes
0 0 0 0
0.01 V 0.08 0.04 V 0.07 0.3 V 0.05 0.81 < 0.01
0.91 0.88 0.65 0.18
0.22 0.09 0.02
0.07 V 0.35 Goal 0.79 < 0.05 0.97 < 0.01
0.37 0.07 0.01
0.9 0.83 0.49 0.1
0.03 ^ 0.07 0.13 ^ 0.04 0.5 < 0.01 0.9 < 0
0.01 0 0 0
0.97 0.95 0.76 0.32
0.02 ^ 0.01 0.04 ^ 0.01 0.23 ^ 0.01 0.67 < 0
0 0 0 0
4
3
2
1
x
1 2 3 4
y


Action probabilities after 5000 episodes
0 0 0 0
0 V 0.11 0.01 V 0.09 0.04 V 0.07 0.27 V 0.05
0.88 0.9 0.89 0.68
0.12 0.17 0.07
0.01 > 0.5 0.06 > 0.4 Goal 0.79 < 0.06
0.36 0.37 0.08
0.77 0.85 0.78 0.4
0.02 ^ 0.19 0.04 ^ 0.1 0.15 ^ 0.06 0.57 < 0.02
0.02 0.01 0.01 0
0.94 0.96 0.94 0.72
0.01 ^ 0.05 0.02 ^ 0.02 0.05 ^ 0.01 0.27 ^ 0.01
0 0 0 0
4
3
2
1
x
1 2 3 4
y


Action probabilities after 5000 episodes
0 0 0 0
0 V 0.01 0 V 0.01 0 V 0.01 0.02 V 0.01
0.99 0.99 0.99 0.97
0 0 0 0
0 V 0.12 0.01 V 0.1 0.03 V 0.08 0.25 V 0.06
0.88 0.89 0.89 0.69
0.07 0.12 0.05
0.02 > 0.52 0.05 > 0.45 Goal 0.79 < 0.08
0.39 0.38 0.08
0.59 0.75 0.71 0.28
0.04 ^ 0.33 0.06 ^ 0.17 0.2 ^ 0.08 0.68 < 0.03
0.04 0.02 0.01 0
4
3
2
1
x
1 2 3 4
y


Action probabilities after 5000 episodes
0 0 0 0
0 V 0.01 0 V 0.01 0 V 0.01 0 V 0.01
0.99 0.99 0.99 0.99
0 0 0 0
0 V 0.13 0 V 0.13 0.01 V 0.1 0.03 V 0.09
0.87 0.87 0.89 0.89
0.04 0.06 0.09
0.01 > 0.52 0.02 > 0.54 0.05 > 0.48 Goal
0.43 0.38 0.38
0.37 0.57 0.73 0.62
0.02 > 0.54 0.04 ^ 0.35 0.07 ^ 0.18 0.27 ^ 0.09
0.06 0.04 0.02 0.01
4
3
2
1
x
1 2 3 4
y



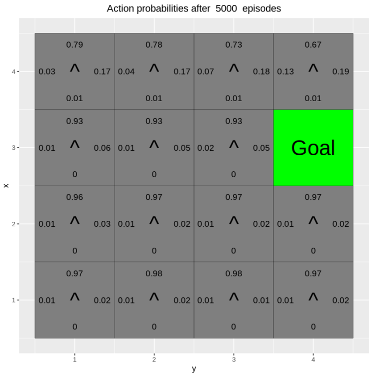


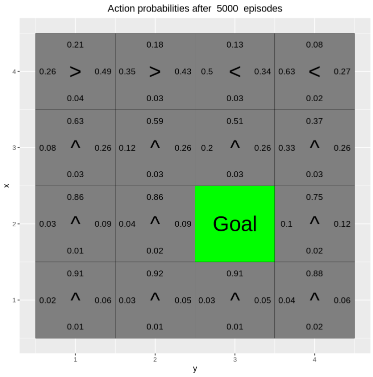
Since those goal actually not exist in training set, but still, the agent can find the goal with extra feature (here is just the position of the goal) in most time.

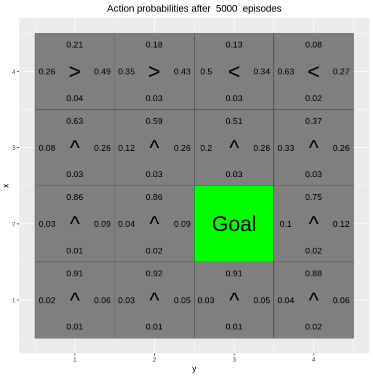
Could you have used the Q-learning algorithm to solve this task ?

Answer: In Q-learning, we only have the fixed Q-table, once we changed the position of goal, we need to train a new table, so it is not possible to do it.

Environment E







Has the agent learned a good policy? Why / Why not ?

Answer Not at all, the agent can not find the final goal in most time. This is because the training sets are only on the top row, but validation sets are on the rest position. There is a huge domain gap between training set and validation set, which means two datasets are not overlapped, or I can say they from two totally different distribution. This model heavily overfitted on training set therefore can not generalize to validation set.

If the results obtained for environments D and E differ, explain why.

Answer: In D, training set and validation set are "overlapped", or I can say the interpolation is possible. But in E, the extrapolation is not possible, it is not only impossible for Deep-Q, it is impossible for any machine learning algorithm. Or as mentioned above, the model totally overfitted in E.