Orçun Özdemir

54020

COMP341: Introduction to Artificial Intelligence, Homework 6 Report

Answer for Q1:

Agents move according to optimal policy which is summation of values in certain grid in our pacman world. Since, agents only move when they observe the optimal values which are simply calculated numerical values, we can easily say that agents behave like an reflex agents. When it comes to value iteration, agent does not calculate values via experience, we calculate or iterate values before agent starts to choose an action. Since, optimal policies already defined before agents actions, we can say that value iteration is "offline planning".

Answer for Q2:

For the programming Question 2, I set the discount factor as default which is 0.9 and set the noise parameter to 0. I know that I should not decrease discount factor since, discount factor forces agent to achieve the goal as soon as possible and when I decrease the discount factor, I observed that since all the grids that below and above have negative rewards, agent would not decide to move in case of falling into negative reward therefore, I had to force it even if the agent is surrounded with negative rewards. Noise was crucial for finding the right path for the agent since, any disrupted grid value would result agent to decide on choosing the negative reward. Thus, I set the noise parameter to 0 which caused a series of values that directly guide the pacman to the positive reward state.

Answer for Q3:

For 3.a, I chose discount factor as 0.9 because I wanted to move the agent to a reward , noise parameter as 0.1 since even a little noise can discourage the agent to go to the further reward and eventually it will decide on the shorter reward and living reward as -3.0 since I wanted to force the agent to go to and choose the positive reward in a certain iterations.

For 3.b, I choose discount factor as 0.1 because I do not want to risk my agent to fall into negative reward even if its slower to achieve to go to positive reward. Therefore, with low discount factor my agent would not dare to go through more dangerous path, noise parameter as 0.1 because I want my agent to go to the positive reward, if I choose to bigger discount it may believe that any reward grid might be positive one which is quite dangerous and highly risky action to do since there are numerous negative rewards on the path.

For 3.c, I choose discount factor as 0.9 because I want my agent to take a risky action and decide to go to further path, noise factor as 0.0 since even a little noise can manipulates its decision very badly. There are negative rewards along its way and my agent should not disrupted by false Q values.

For 3.d, I choose discount factor as 0.9 because I want my agent to take a risky action and decide to go to further path, noise factor as 0.1 because my agent decides that the noisy values can led its doom which forces it to avoid dangerous path even if its discount factor is high.

For 3.e, I choose both discount factor and noisy parameter as 0.0, since no guidance or beneficial policy to take action with certain values is meaningless. In addition I set the living reward parameter as 15.00 which causes pacman to stay, since living reward is bigger than the positive reward in the terminal state.

Answer for Q4:

After the execution of first command line, there is an offline value iteration process. Values of the grids are calculated before pacman explores them which is highly natural since pacman does not use reinforcement learning thus, it only exploits not explores. If the values would not be calculated before, pure exploitation would not work as well. On the other hand, after the execution of second command line pacman starts with almost zero information about the values. Therefore, it has to gain experience in this case at 100 iterations. Pacman goes to any terminal state does not matter if its negative or positive reward, it must explore them. Since, after numerous exploration the agent exploits what it learns and finds the optimal policy through its experiences. The major difference between, the results of first and second command line is the first one does not learn the values by itself they are given to them then it decides which direction should the agent choose. The latter make pacman to learn itself and decide from what its learned.

Answer for Q5:

I run the following line "python gridworld.py -a q -k 50 -n 0 -g BridgeGrid -e 1" and agent explored at most 4 blocks then completed its 50 iterations. After I altered the epsilon rate to 0 and run the line again, agent did not even explore 4 blocks it just followed a policy and exploited it. Therefore, according to my observations there is not an epsilon and a learning rate for which it is highly likely that the optimal policy will be learned after 50 iterations.

Answer for Q6:

When I run q-learning in mediumGrid, my agent does not work very well, in fact it failed all of its test games and training take more time than smallGrid. Pacman fails on mediumGrid since each board design is a distinct state with distinct Q-values. The agent cannot generalize to encounter a ghost is bad for all positions. Therefore, running on mediumGrid won't scale in terms of our project.

Answer for Q7:

I run Approximate QLearning test line and observed that agent worked even for larger environments, on the contrary It did not work in test of question 6. Thanks to features and their weight, agent can make assumptions about other possible grids in mazes which led the agent to go through all grids in a specific maze with a small data. In addition, I clearly see that my agent made actions and obey its policy according to the features' contents. I observed that the feature that what should agent do in case of ghosts are scared was not in featureExtrator.py, so during the test even though scared ghost is highly close to the agent, agent decided to change its direction instead of eating them. Consequently, if I add a proper feature to the features which made my agent to eat scared ghosts, pacman's behavior would be improved and maybe overall points would be increased as well.