Final Report

Artificial Intelligence, COSE361-02

2014210040

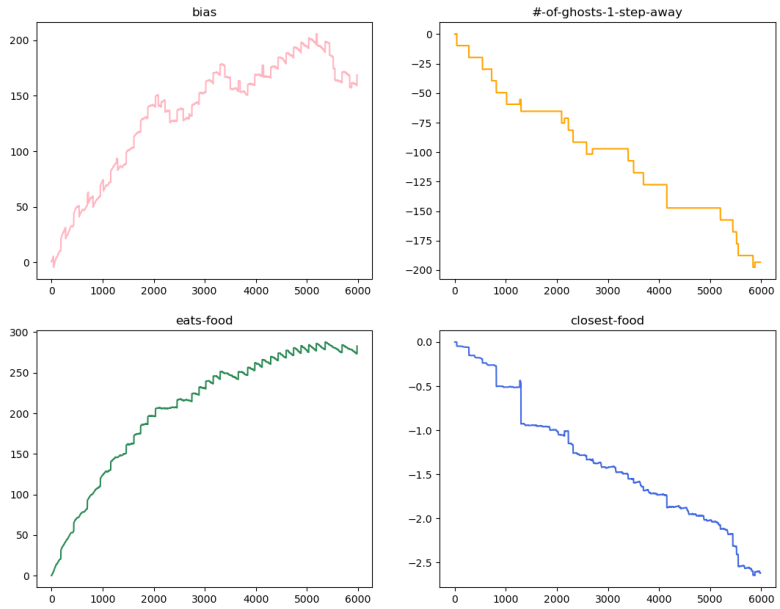
이수호

1. autograder.py result screenshot

```
Finished at 19:39:48

Provisional grades
==================
Question q1: 0/4
Question q2: 0/1
Question q3: 0/5
Question q4: 0/1
Question q5: 0/3
Question q6: 4/4
Question q7: 2/2
Question q8: 1/1
Question q9: 1/1
Question q10: 3/3
------------------
Total: 11/25
```

2. Result of log_Q_weights and discussion of their behavior



We have three different features in `SimpleExtractor` provided and a 'bias', which will be discussed later. Feature `#-of-ghosts-1-step-away` represents count of total ghosts which are 1 step away from agent. Feature `eats-food` is 1.0 if agent eats a food otherwise zero. Feature `closest-food` represents closest food distance divided by dimension of the game map. All of the features are divided by 10, so they never exceed 1.0 in this scenario.

First thing we can notice is that direction of the weight moving among iteration. Weight of the feature `eats-food` increases, while weights for `#-of-ghosts-1-step-away` and `closest-food` decreases. Moving toward negative value means that $Qdiff < 0$ which also means following reward is negative (did nothing or the agent died) or not sufficient. Weights which are moved to negative value means that higher value of those feature affects negatively to Q-Value, or total score for the game. We can intuitively find out that decision that makes those higher can make a negative effect to the game.

One thing to concern is that doing nothing also makes the score worse. The only way to constantly gain score is to constantly find and eat food. This can explain that why only weight of `eats-food` is increasing *constantly*. Also, there is an area where features are not updated through iteration and it simply indicates that the feature $f_i(s, a)$ was zero at that point.

Next thing to concern is the absolute value of the weight. The absolute value of the weight for feature `closest-food` is far smaller than others because the feature is derived from $dist/(width * height)$, which has a smaller value than the others.

Notable thing here provided is the bias. Bias is 0.1 for every (state, action) pair and weight of the bias is just the sum of $0.1 * \alpha * Qdiff$. As the agent iterates it can get a better $(Qdiff > 0)$ or worse $(Qdiff < 0)$ score. From the plot, we can see that the weight of bias was decreased around iteration 2500-3000 and 3500-4000. Weight of `eats-food` from the same iteration is not updated drastically and that seems to be the reason why agent was performed worse.

3. Q-value convergence regarding epsilon value

In the epsilon-greedy search, epsilon value defines the agent's tendency to *explore* – act randomly. As a result, higher value of epsilon means quicker convergence of Q-value regarding iteration. I made a graph visualizer to find out those behavior.

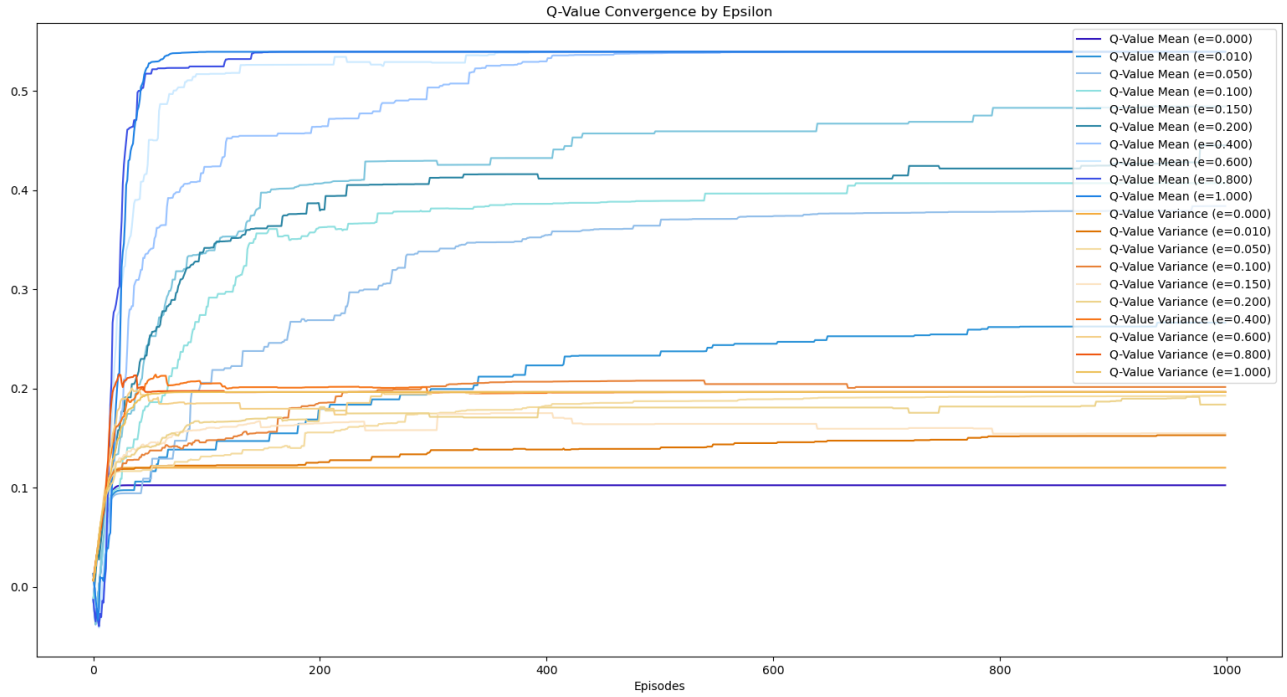


FIGURE 1 DIFFERENT CONVERGENCE BEHAVIOR REGARDING EPSILON VALUE

Mean of Q-value (colored blue) converges to specific value after it accumulates. Note that variance of each Q-value (colored orange) did not change after the mean of Q-value converges. It means that, ultimately, all of the Q-value converges with specific value. In general, iteration with a higher epsilon value generally converges faster, which is as same as the assumption. Agent with no further exploration (epsilon=0) should never exceed Q-Value of 0.1 in this case and the graph also shows that behavior.

4. Q-value convergence regarding alpha value

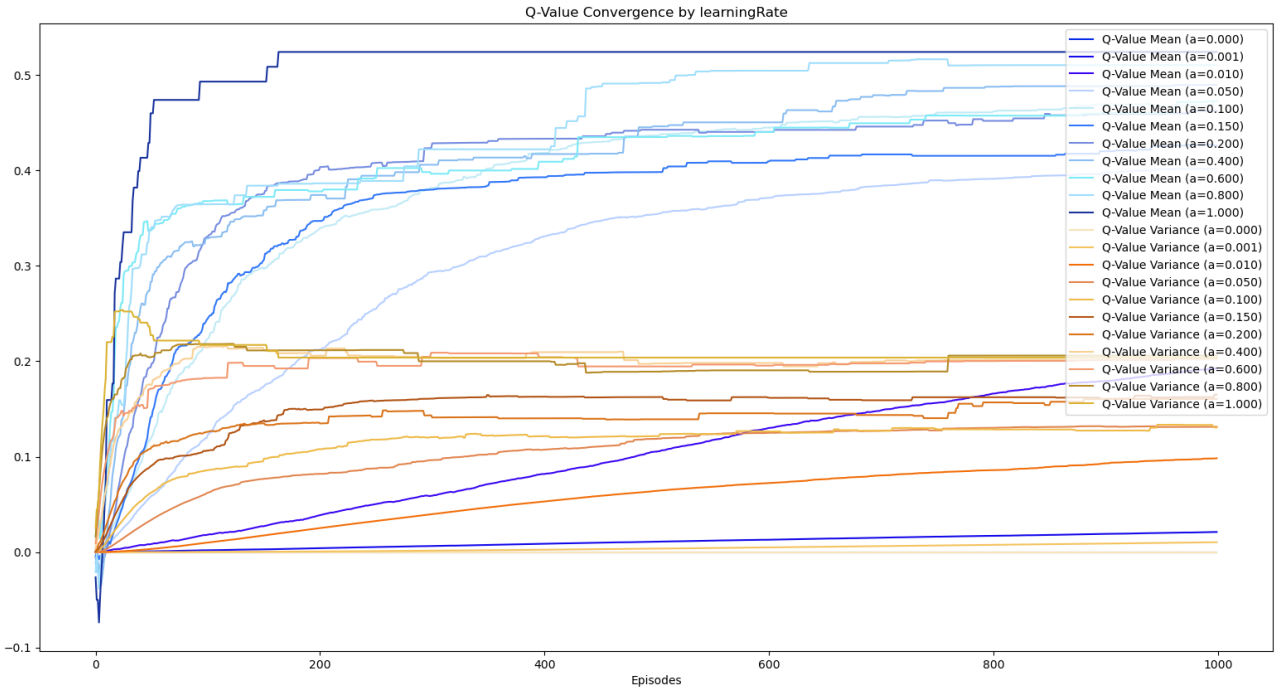


FIGURE 2 DIFFERENT CONVERGENCE BEHAVIOR REGARDING LEARNING RATE

Learning rate affects Q-value convergence more drastically. In general, higher learning rate also results in faster convergence of Q-value. One notable different thing between Q3(considers epsilon value) and this is that the pattern of the graph is different in case of low values. Agent with lower epsilon value has stair-shaped graph while agent with lower learning rate has smoother graph. It means that agent with low alpha value constantly 'exploits' something (with a lower manner), while agent with low epsilon value exploits nothing or something more (sometimes does not explore further).

Agent with zero learning rate does not improve Q-value at all, since it learns nothing from the exploration. This behavior is shown as [Figure 3] below.

I have included the visualizer source code, named gridworld_plot.py, which is fully compatible with gridworld.py without any modification. The source code has some dependencies, such as matplotlib.

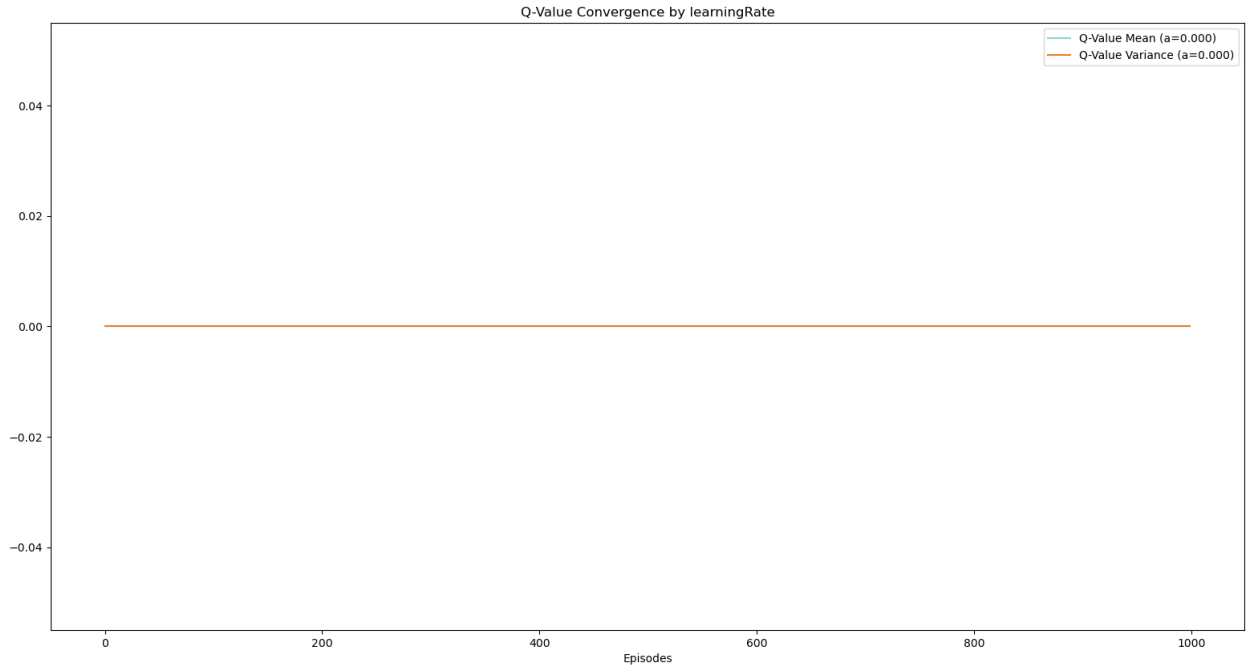


FIGURE 3 AGENT DOES NOT LEARN ANYTHING (LIKE A BAD STUDENT)

5. Discussion of new features which will improve agent

One substantial thing to consider is the scared timer of the ghost. In `SimpleExtractor`, the agent runs away from the ghost without considering the ghost scared or not. Adding this feature enables the agent to consume the ghost in a scared state. Another thing to consider is the proximity between the agent and a capsule, which makes ghost scared. In most case, the capsule is located alongside foods to consume. However, adding this feature might be helpful if the capsule is not located next to the food, and the agent is far from the food.

I have included implementation of the features mentioned above in featureExtractors.py.