FINAL ARTIFICIAL INTELLIGENCE ASSIGNMENT

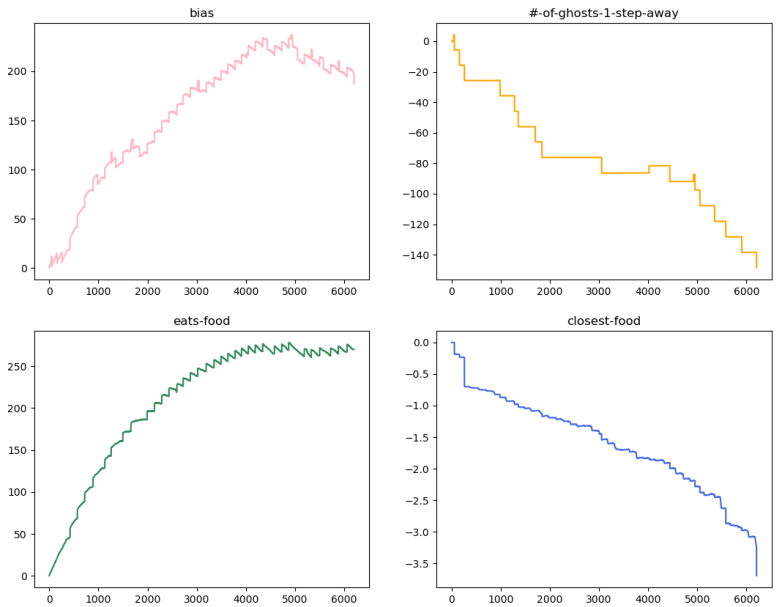
컴퓨터학과 2016320187 홍성윤

1. Screenshot the result of autograder.py in the terminal

```
Finished at 1:28:04

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. Discuss the different behaviors of the weights for each feature in log_Q_weights.png image.



➔ 위의 그래프들은 여러 sample들을 통해 반복하면서 가중치를 최적화하는 과정을 보여준다.

#-of-ghost-1-step-away의 가중치는 0에서부터 점점 작아지다가 -140 이상까지 상당히 작은 값으로 수렴한다. 이와 같은 가중치를 얻음으로써 팩맨은 고스트에게 거의 먹히지 않는다. 또한 #-of-ghost-1-step-away의 가중치는 다른 feature 들에 비해 변화가 계단식으로 급격하게 일어남을 알 수 있다. 고스트와의 거리가 한 칸 이하일 경우, 게임 및 점수에 미치는 영향이 매우 크기 때문에 나타난 현상으로 보인다.

eats-food 가중치는 점점 증가하고, closest-food 가중치는 감소한다. 이로써 팩맨은 가능한한 음식을 많이 먹고 음식과의 거리를 최소한으로 좁혀 나갈 수 있게 되었다. Eats-food 가중치의 절대값이 250이상으로 훨씬 더 크기 때문에 음식에 가까워지도록 움직여 놓고 막상 음식은 먹지 않는 상황은 발생하지 않는다. 또한 eats-food feature의 값은 고스트와의 거리가 1 이하일때는 0이기 때문에, eats-food 가중치가 크다고 음식에만 집중하느라 고스트에게 먹히는 상황은 발생하지 않는다.

Bias feature는 각 state의 feature 값의 편차를 줄이기 위한 값으로 생각된다. 항상 bias는 1이므로 q-value는 기본 bias 값에 feature와 weights의 곱들을 더한 값이 된다. Bias는 약 200에 수렴하고 이로써 적합한 q-value를 얻을 수 있다.

3. In the epsilon-greedy search, discuss the convergence of Q-value according to epsilon (e.g., compare the convergence of Q-values when epsilon is 1 with ones when epsilon is 0).

-> epsilon-greedy search 에서는 epsilon 값을 통해서 exploration 과 exploitation의 비율을 정할 수 있다. 큰 Epsilon에서는 agent가 탐험을 많이 할 것이고, 작은 epsilon에서는 기존의 policy대로 움직일 것이다. 만약 epsilon 값을 1으로 한다면, agent은 항상 랜덤한 움직임으로 학습을 하기 때문에 최적의 값과 거리가 먼 학습을 하게 된다. Epsilon 값이 0 이라면, agent는 기존의 계산된 값을 바탕으로 움직인다. 하지만 초기 값들이 모두 0이기 때문에 랜덤한 움직임을 보인다. 만약 새로운 q-value를 얻었어도, 새로운 탐험 없이 움직이기 때문에 정해진 policy대로만 움직이게 되고 오히려 최적의 값을 얻지 못하는 모습을 보인다. 따라서 최적의 q-value에 수렴하기 위해서는 epsilon 값을 잘 정해야 할 것이다.

4. In the Q-learning, discuss the convergence of Q-value according to alpha (e.g., compare the convergence of Q-values when alpha is 1 with ones when alpha is 0.

-> alpha는 learning rate로서 새로운 sample 이 새롭게 update될 q 값에 미치는 영향을 의미한다. Alpha 값이 너무 크면 새로운 sample의 영향력이 커지고 q값이 자주 바뀌게 된다. Alpha 값

이 너무 작으면 sample을 통해 학습하는 정보가 적어지고, 최적의 값을 얻기 위해서 필요한 반복 횟수 및 sample의 수가 많아질 수 있다. Alpha 값이 0이라면 새로운 sample에서 얻는 정보 및 update 가 없으므로 q 값은 초기값 그대로 남아있는다. 따라서 적합하지 않다. Alpha 값이 1이라면 기존의 q값에 상관없이 새로운 sample의 정보로만 update가 진행되므로 q값이 수시로 바뀌고 원하는 값에 수렴하는 데에 많은 iteration이 필요하다.

5. In Q10, there are three features used in ApproximateQAgent (#-of-ghosts-1-step-away, eatsfood, closest-food). Discuss any new features that might improve ApproximateQAgent and specific situations that the new features might be helpful.

-> 위의 세 feature와 적절하게 얻어진 각각의 weight 값을 통해 팩맨은 비교적 훌륭하게 게임을 수행한다. 그러나 캡슐에 관한 feature가 존재하지 않기 때문에 캡슐을 먹을 수 있음에도 먹지 않는 경우가 생길 수 있다. 또한 고스트가 scared 상태인지 아닌지에 대한 정보도 없기 때문에 고스트가 scared 상태임에도 고스트에 가까이 가지 않는 팩맨의 판단을 확인할 수 있다. 캡슐과의 거리 정보, 캡슐을 먹었는지에 대한 정보 또는 scared 상태인 고스트와의 거리정보를 feature로 추가하면 팩맨이 고스트를 먹고 게임을 유리하게 진행하는 데 도움을 줄 것이다. 또한 고스트가 둘 이상일때 현재 feature 들 만으로는 고스트에게 포위당하는 상황을 방지하지 못하기 때문에 팩맨이 tunnel에 들어가 있는지에 대한 정보를 new feature로 사용할 수 있을 것이다.

Q5. [Extra credit] 코드 첨부하였습니다.