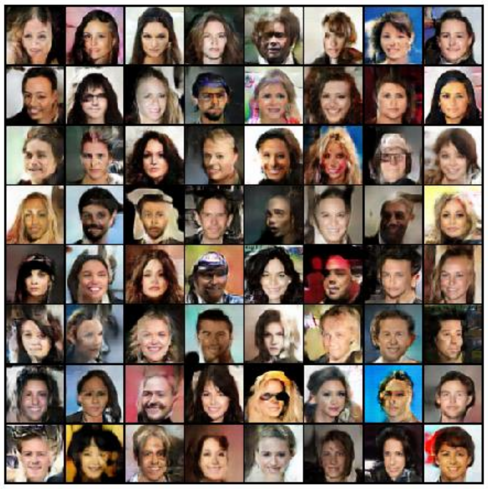
1 Generative adversarial network (GAN)

1. data preprocessing

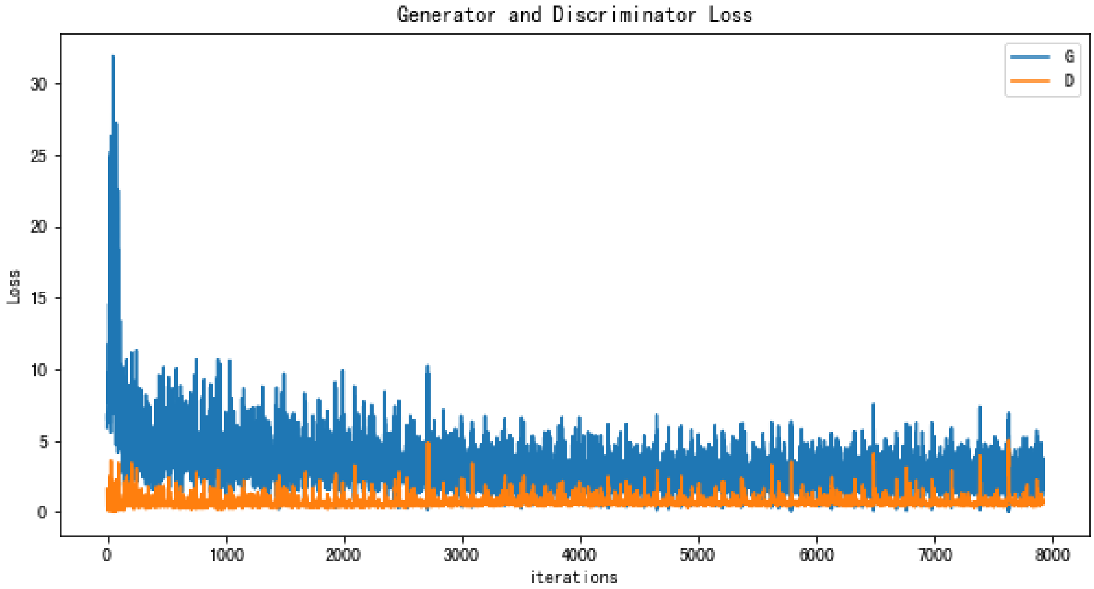
我將圖片統一壓縮成 64*64*3，轉成陣列之後再將其標準化(mean = (0.5, 0.5, 0.5), std = (0.5, 0.5, 0.5))

2.

原圖



生成圖

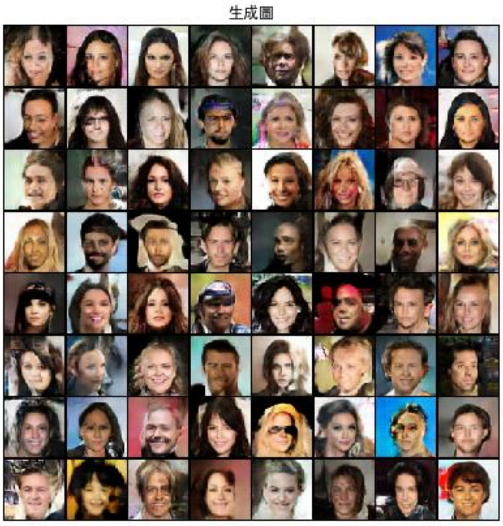


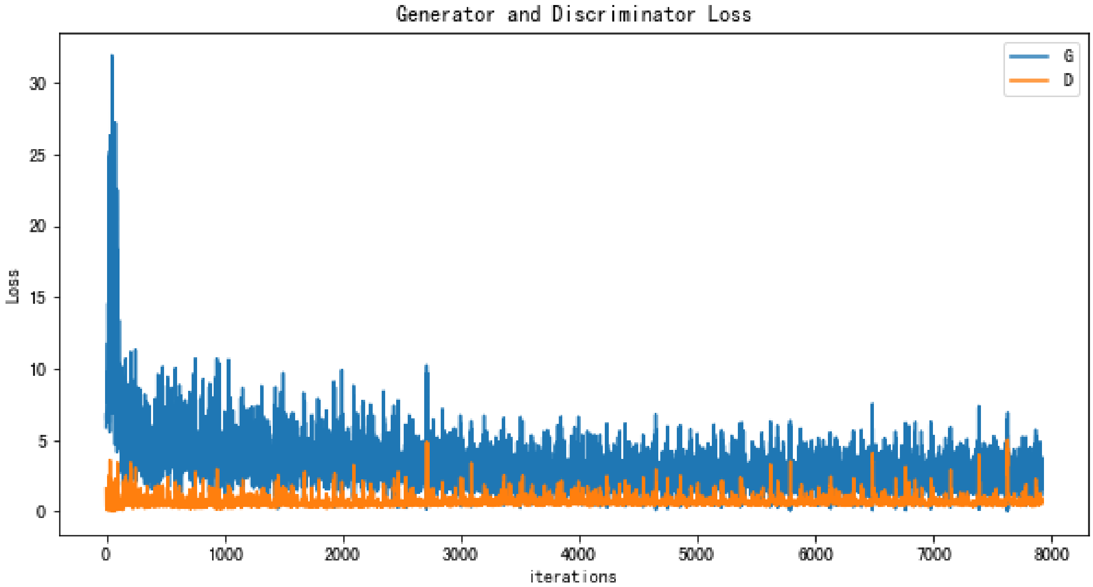
3.

(c)

Main()&train()&Visualization is in hw3_1.py

(d)





(e)

過程中我有條過幾個參數，但效果並沒有比較好，所以我就繼續沿用原本設定好的值

2. Deep Q Network (DQN)

1.

Epsilon greedy:根據當前狀態 S 選取一個動作執行，執行完後觀察 reward 和新狀態 S。

$Q(s_t,a_t)$ 是 old value

α 為學習率

r_{t+1} 為 reward

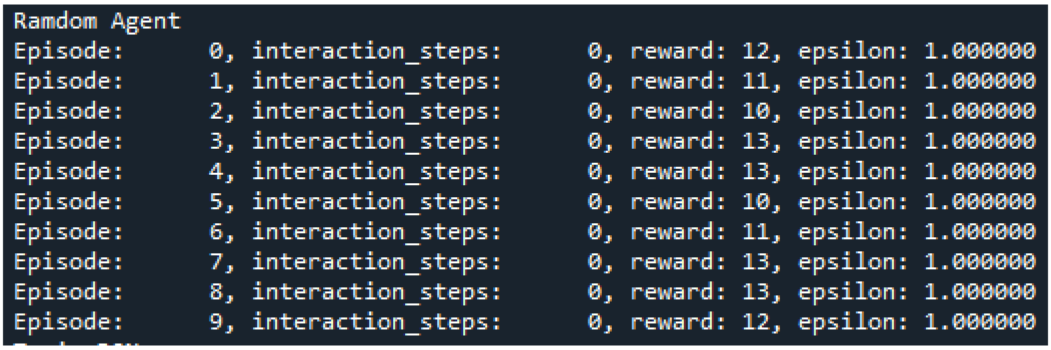
γ 為 discount factor

$\max_a Q(s_{t+1},a_t)$ is the estimate of optimal future value

而 $r_{t+1}+\gamma \bullet \max_a Q(s_{t+1},a_t)$ 為 learned value

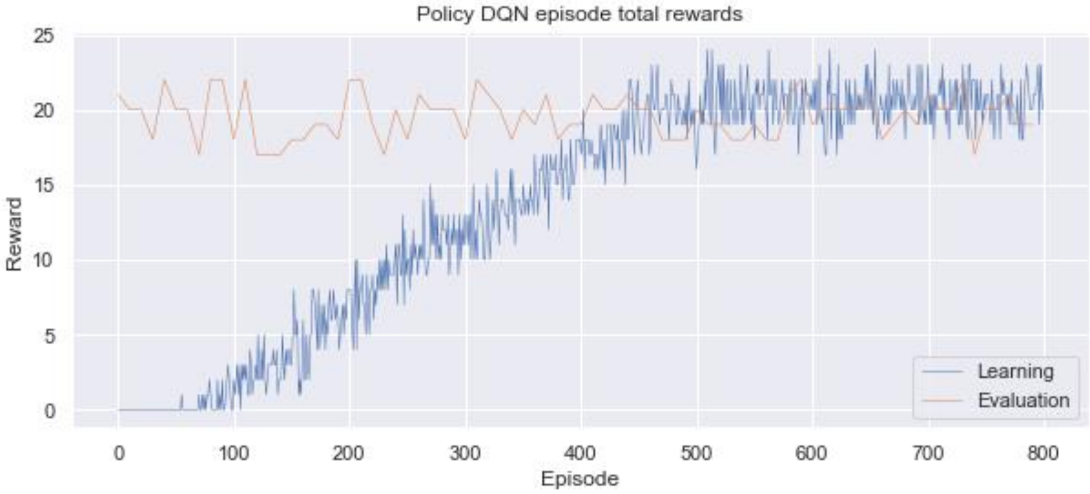
τ 為 target network update period: 整個 trajectory 展開，求機率 P($\boldsymbol{\tau}|\boldsymbol{\theta}$)機率，展開來後從第一項開始：環境初始狀態 p(s_t)，在 state(s_t)狀態下，基於$\boldsymbol{\theta}$所以採取的行動(a_t),接著基於 a_t ，stale1(s_t)過渡到 state(s_{t+1})，中間所產生的 reward(r1)

2.

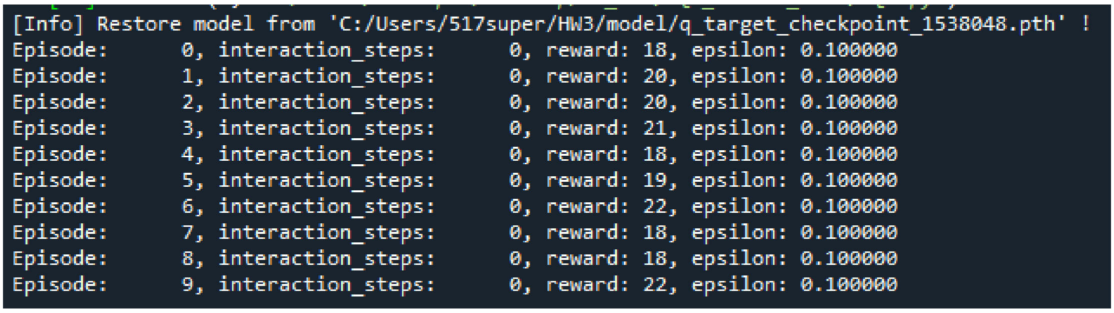


```
Ramdom Agent
Episode:      0, interaction_steps:      0, reward: 12, epsilon: 1.000000
Episode:      1, interaction_steps:      0, reward: 11, epsilon: 1.000000
Episode:      2, interaction_steps:      0, reward: 10, epsilon: 1.000000
Episode:      3, interaction_steps:      0, reward: 13, epsilon: 1.000000
Episode:      4, interaction_steps:      0, reward: 13, epsilon: 1.000000
Episode:      5, interaction_steps:      0, reward: 10, epsilon: 1.000000
Episode:      6, interaction_steps:      0, reward: 11, epsilon: 1.000000
Episode:      7, interaction_steps:      0, reward: 13, epsilon: 1.000000
Episode:      8, interaction_steps:      0, reward: 13, epsilon: 1.000000
Episode:      9, interaction_steps:      0, reward: 12, epsilon: 1.000000
```

3.



4.

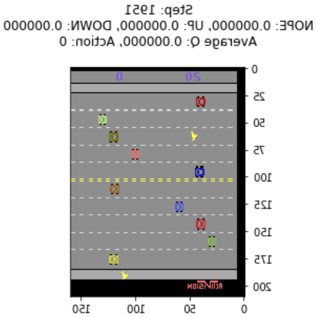


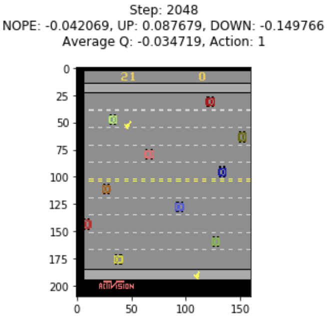
```
[Info] Restore model from 'C:/Users/517super/HW3/model/q_target_checkpoint_1538048.pth' !
Episode:      0, interaction_steps:      0, reward: 18, epsilon: 0.100000
Episode:      1, interaction_steps:      0, reward: 20, epsilon: 0.100000
Episode:      2, interaction_steps:      0, reward: 20, epsilon: 0.100000
Episode:      3, interaction_steps:      0, reward: 21, epsilon: 0.100000
Episode:      4, interaction_steps:      0, reward: 18, epsilon: 0.100000
Episode:      5, interaction_steps:      0, reward: 19, epsilon: 0.100000
Episode:      6, interaction_steps:      0, reward: 22, epsilon: 0.100000
Episode:      7, interaction_steps:      0, reward: 18, epsilon: 0.100000
Episode:      8, interaction_steps:      0, reward: 18, epsilon: 0.100000
Episode:      9, interaction_steps:      0, reward: 22, epsilon: 0.100000
```

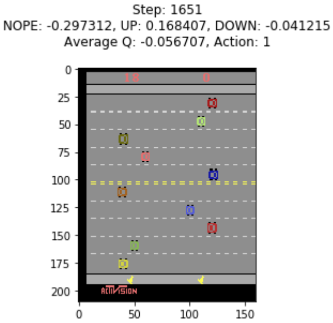
5.

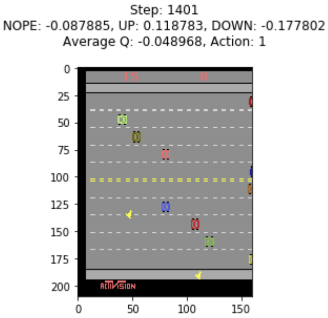
DQN decision in the game is the same as mine

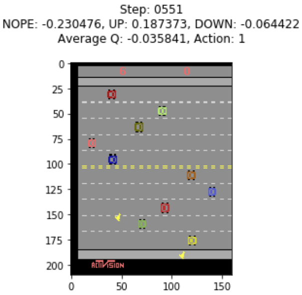
The average fraction that flipped their relative order after projection onto the gaussian bases. This value is somewhat larger than what was seen, but a difference is not unexpected given the radically different distribution of the bases. The average, weighted, percentage of the states for which the optimal action, according to the lookup table Q-Values, was no longer optimal with respect to the approximated values. This value may be smaller than the value we would except if the flipping prob were independent.

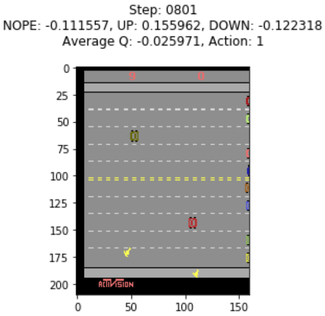

Step: 1951
NOPE: 0.000000, UP: 0.000000, DOWN: 0.000000
Average Q: 0.000000, Action: 0


Step: 2048
NOPE: -0.042069, UP: 0.087679, DOWN: -0.149766
Average Q: -0.034719, Action: 1


Step: 1651
NOPE: -0.297312, UP: 0.168407, DOWN: -0.041215
Average Q: -0.056707, Action: 1


Step: 1401
NOPE: -0.087885, UP: 0.118783, DOWN: -0.177802
Average Q: -0.048968, Action: 1


Step: 0551
NOPE: -0.230476, UP: 0.187373, DOWN: -0.064422
Average Q: -0.035841, Action: 1


Step: 0801
NOPE: -0.111557, UP: 0.155962, DOWN: -0.122318
Average Q: -0.025971, Action: 1