ADL HW3-Game Playing Report

b03705012 資管四 張晉華

1.Model description

- Policy Gradient(Pong)
  - 將畫面裁剪掉上方比數的部份（變成 160*160 的矩陣），並且去除顏色，改成板子和球為 1 其他背景部份為 0 的畫面，再 resize 成 80*80 的矩陣作為 state。
  - 不斷的進行遊戲，並記住每個 state 變化、相對應的動作和經過 running discount 的 reward，一場遊戲結束後拿來 fit model。input 是 state 相減來呈現 state 的變化，output 則是原先預測各動作的機率再加上 learning rate、gradient(這次做的動作為 1 其他為 0 – 原先預測各動作的機率)、running_discount_reward(用得（失）分來決定梯度方向，時間愈前面加權愈低)相乘為 output。
  - 由於 OpenAI 的 Pong 動作(0,1),(2,4),(3,5)動作一樣，因此只取動作 1,2,3 來預測。
- DQN Learning(Breakout)
  - Hyperparameter：
    memory size = 50000 步
    gamma = 0.99

    online network 更新頻率 = 1 次 episode

    target network 更新頻率 = 100 次 episode

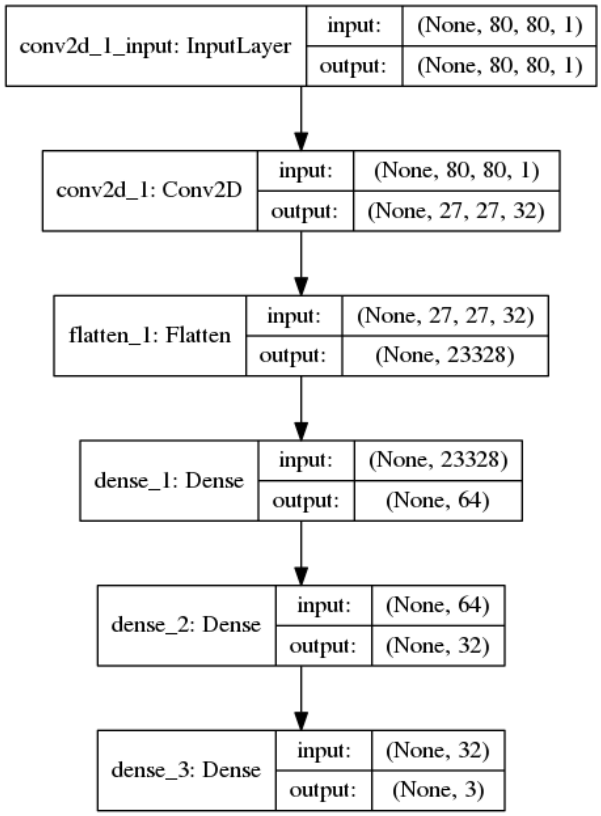
    總次數 = 50000 次 episode

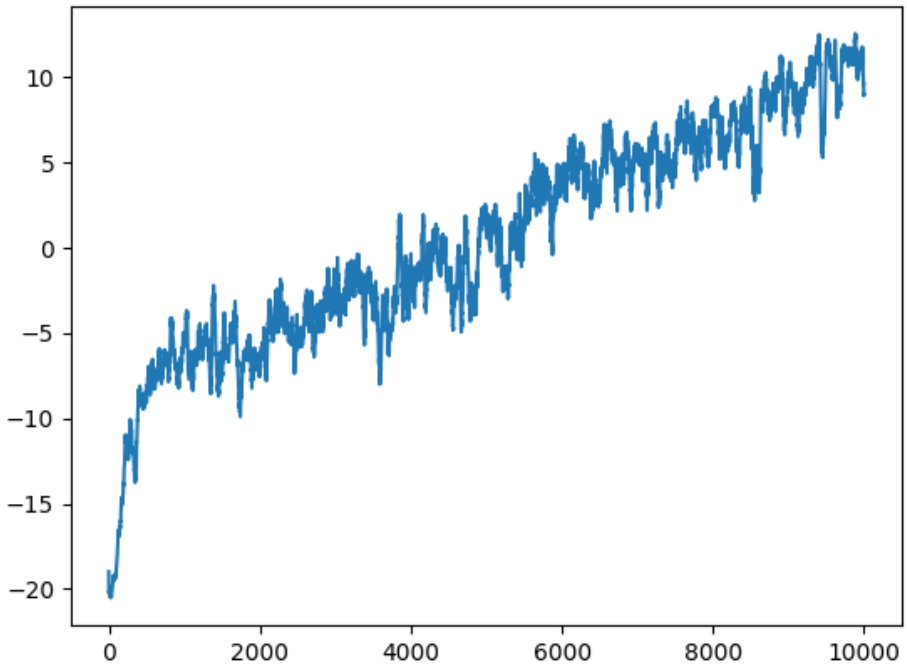
    replay_batch_size = 64 筆紀錄

    epsilon = 1.0 到 0.05（前 2%的 episode 線性下降）
  - 不斷的進行遊戲，並記住每個 state、相對應的動作、reward 和所造成的下個 state，然後依照 online network 更新頻率定期從 memory 中隨機選出 replay_batch_size 筆紀錄 replay 來 fit model。Input 是每筆紀錄的 state，output 則是 reward + (1-done)(如果結束了不會有下個動作的期望值)*gamma*max(target_network_predict(下個 state,每個可能的 action))

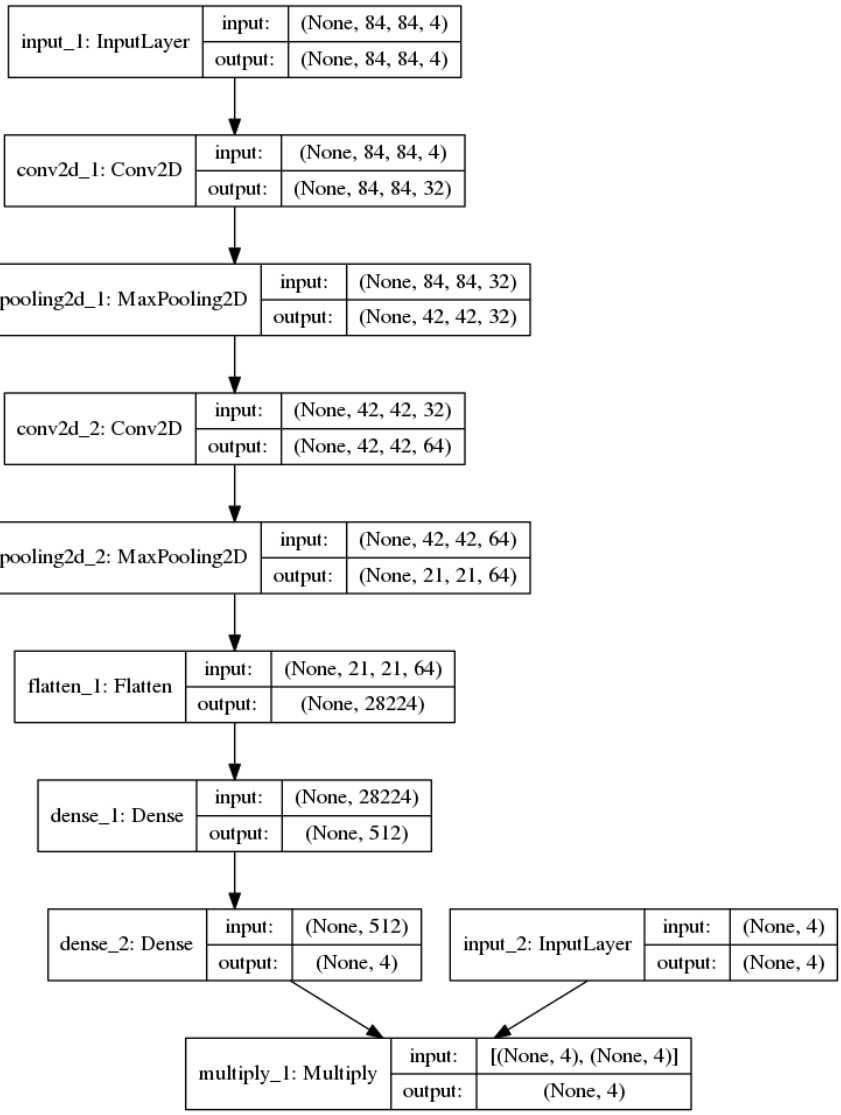
- model 在 output 前多加上一層 action mask，在 make_action 時設全部為 1 來選擇全部的動作中期望 Q-value 最大的，而 fit 時就只考慮 replay action 的 loss 就好。

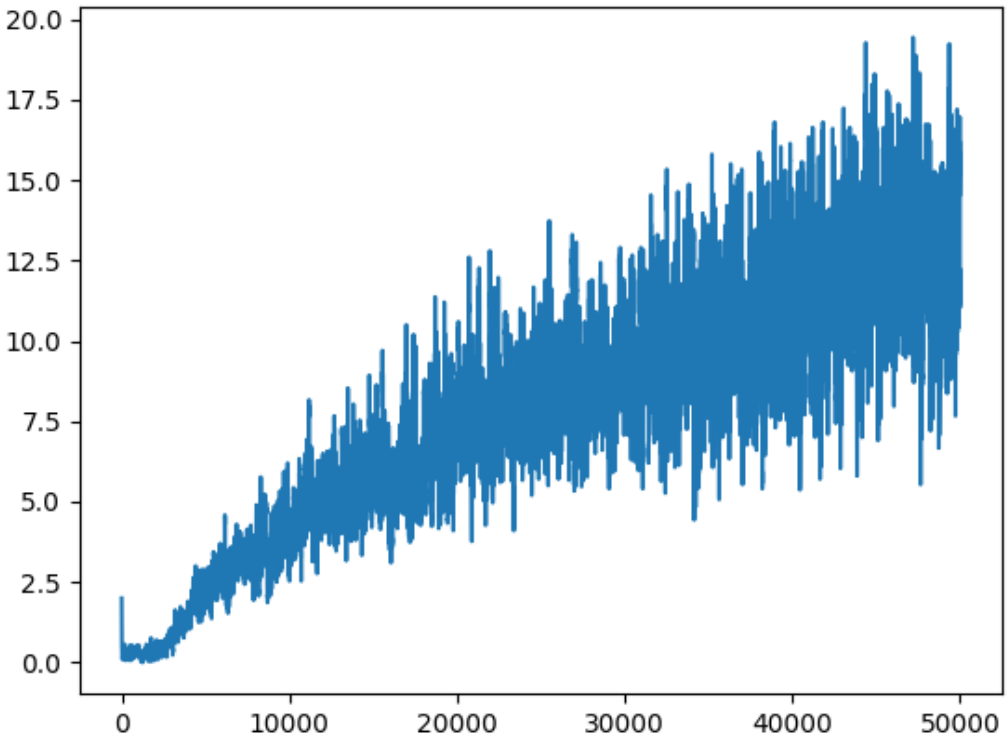
- 模型架構 &
  Learning curve(x,y = episodes, 前 30 episodes 的平均分數)
  - Policy Gradient(Pong)





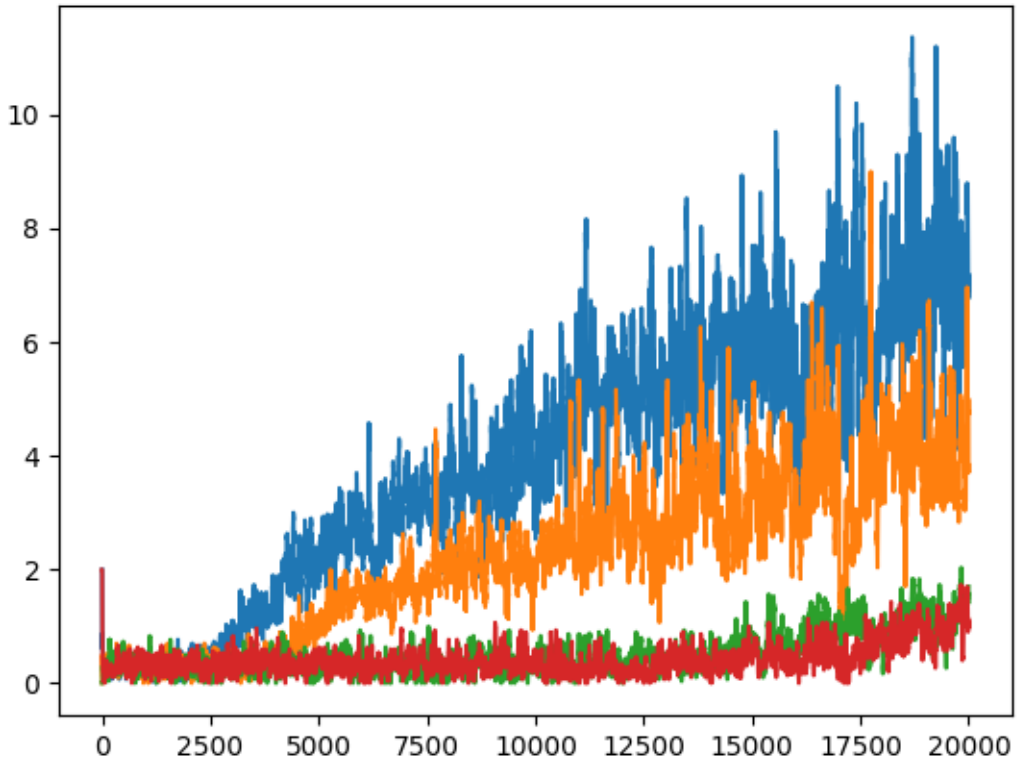
- DQN Learning(Breakout)





2.Experimenting with DQN hyperparameters

- 選擇 online network 更新頻率做實驗，想了解同樣遊戲次數的情況下，更新頻率對學習速度和效果的影響。
- 實驗結果
  （藍 = 1, 橘 = 2, 綠 = 4, 紅 = 8　單位：episodes）
  （x-axis : episodes, y-axis : 30-episodes moving window）



- 在初期分數較低的階段，一個 episode 大約是 25 steps 上下，因此在 batch_size = 64 的情況下，對更新頻率為 1,2 個 episodes 的 model 而言，兩次 replay 的中間所產生的新經驗比一次學習的量還少，因此 exploration 的經驗有被充份的學習，分數上升的速度也較快，且每次除了新的經驗以外還能再重新學習之前學過的經驗，能更趨近於學習的 target，因此學習速度快慢會跟更新頻率有關，學習也較不容易遇到瓶頸，但對於更新頻率為 4,8 個 episodes 的 model 而言則相反，因此開始進步的時間變得很晚，分數上升的速度也變相當的慢，且因為兩次 replay 的中間所產生的新經驗比一次學習的量還多，因此每次的經驗沒有很充份的被學習，學習速度跟更新頻率也沒有太大的相關。