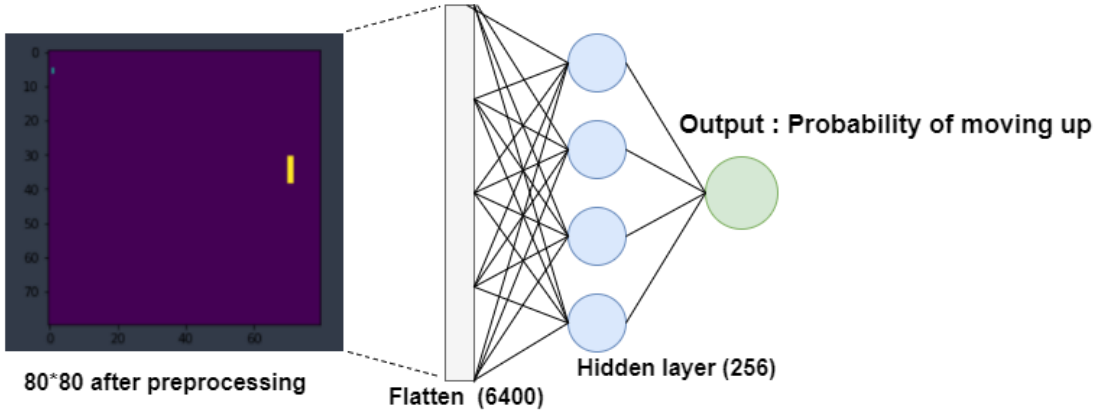
R06946003 湯忠憲 Game Playing

Basic Performance :

Describe your Policy Gradient & DQN model

Policy Gradient:

本次作業的 Policy Gradient 是實作在 Pong 這個遊戲上。在 Policy Gradient 中，model 的輸入是遊戲的畫面 pixel，輸出的部分則是 action，也就是基於$\pi(a|s,\theta)$選擇輸出動作，*其中，a 是 action，s為觀察到的pixel，θ是模型參數*。在 Pong 遊戲中，我們需要考量的 action 有 up 跟 down，所以輸出可用 sigmoid，大於一定機率時給定 up。Loss function 是用 log likelihood，我們希望增加那些 reward 高的 action 的出現機率，並且反之減少那些會讓比賽輸掉的動作的出現機率。而最終的目標函數則是希望能夠最大化(decay)reward 的累加期望。Policy Network 的部分我使用兩層全連接層，單元個數分別為 256 和 1，整體架構如圖一。



圖一、Policy Network

模型相關參數:

Optimizer : Adam(learning_rate=0.0005)
Hidden size : 256 or 200
Discount factor : 0.99
Batch size : Update network every episode

在進入模型前先將圖片做預處理以降低學習難度和過濾不必要的資訊。可以把計分板和上下的空白處拿掉，並把三原色模式變成單一色域。所以原本(210,160,3)的畫面變成 (80,80) 的正方形。由於需要觀察兩張 observation 的變化才能知道球是飛過來還是打過去了，需要將連續兩次接收到的 observation 相減。接著再將它壓平成 6400 維的向量進入神經網絡。這邊也可以不壓平然後進入捲積神經網路，不過我先用一般的神經網路，發現也訓練得起來。訓練過程如圖二。大概跑兩天 reward 可以超過 7。

圖二、Policy Gradient 訓練圖

DQN:

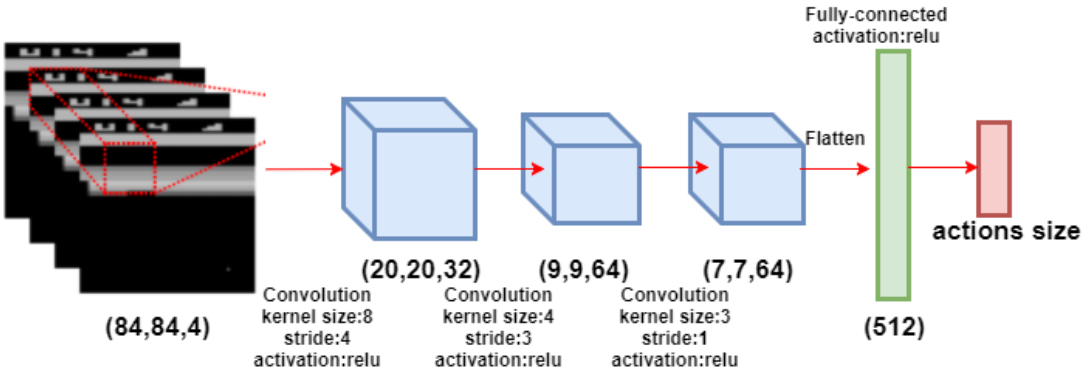
Q-learning 中我們考慮的是接收到一個環境變化後，採取每一種選擇所能得到的回饋(Q 值)。因此，希望能藉由神經網路來更準確的預測 Q 值。Loss function 如圖三:

$$loss = \left(\underbrace{r + \gamma \max_{a`} \hat{Q}(s, a`)}_{\text{Target}} - \underbrace{Q(s, a)}_{\text{Prediction}} \right)^2$$

圖三、DQN loss function[1]

$(\boldsymbol{r}, \boldsymbol{\gamma}, \boldsymbol{s}, \boldsymbol{a})$ 分別是**(reward, discount rate, observation, action)**

$\hat{Q}$是$target\ network$，Q是時時更新的$network$。其中使用 network 來得到 Q 值而不是建立 table 用來搜索 Q 值是 DQN 與一般 Q-learning 不一樣的地方。Network 的部分輸入是遊戲畫面，經過捲積神經網路，輸出 Q 值。整體架構如圖四:



圖四、DQN model

模型相關參數:

Gamma : 0.99
Batch size : 32
Experience replay size : 100000
Epsilon range : (0.1-1)
Target network update frequency : 10000
Online network update frequency : 4
(Total) Time steps to observe before training : 50000

[1] https://www.slideshare.net/carpedm20/ai-67616630

Exploration rate (epsilon decay) : 9e-7 per step after 50000 steps (observe stage)
Optimizer : RMSProp (learning rate = 0.00025)

本次作業的 env 傳進來的已經是處理好的畫面資料，即將原本的(210, 160, 3)處理成(84,84)，並將四張圖疊起來以觀察變化 (在 Pong 的時候是透過 observation 相減得到變化的資訊)。訓練約一天可以達到約 70 的 reward。訓練過程如下圖:

圖五、DQN 訓練圖

Experimenting with DQN hyper-parameters:

本題我挑選 Gamma 作為實驗參數。Gamma 為圖三中的 discount rate，是希望離當下越遠的時間點所估計的回饋能夠隨著距離遞減，使得時間距離拉越長，影響越小。從公式可以看出這個參數對 loss 有直接的影響，因此我認為有實驗的價值。在這邊使用的參數組合為 **(0.99, 0.95, 0.90, 0.80)** 依序遞減。訓練過程如圖六:

圖六、Gamma 參數實驗圖

從上圖可以發現在 Gamma 為 0.99 時可以在最短的時間內 train 起來，再來是 0.95。若調到 0.80 可以發現整條線是平的，沒有起來。會有這樣的結果可能是因為 model 無法看到長期下來回饋的結果(Gamma 小)，所以做出來的動作容易只是局部最佳化的結果，最終造成整個 train 不起來。

Bonus

Improvement of DQN:

1. Double DQN:

在圖三的式子中可以發現在每次都取 Qmax 來做為估計，這樣會使得模型的輸出相對於現實的 Q 值有過估計的現象，並且誤差會比一般估計來的大，因為是取 max。所以 Double DQN 的算法是希望能夠透過另一個神經網路來估計輸出的 Q 值，來降低過度估計所產生的誤差。而 loss function 就會變成入下圖:

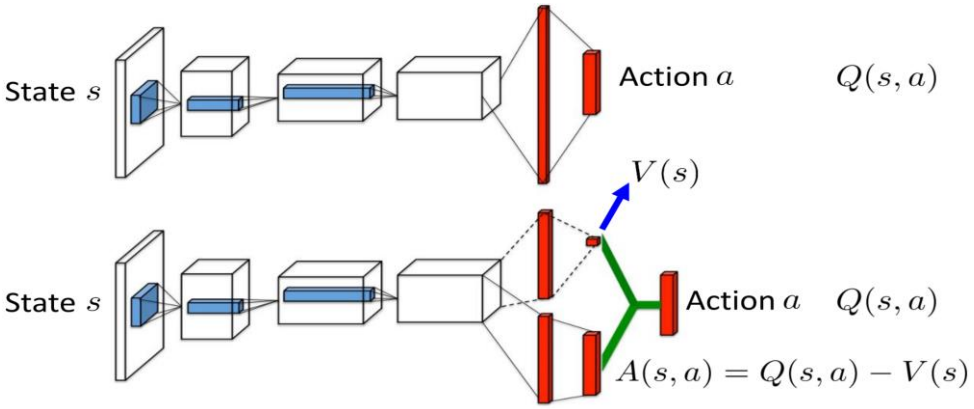
$$loss = \left(r + \gamma \hat{Q}\left(s, \arg\max_{a`} Q(s, a`)\right) - Q(s, a) \right)^2$$

圖七、Double DQN loss function[2]

相較於 natural DQN，這裡使用時時更新的那個神經網路來找出會使 Q 值最大的 action，並將其作為輸入提供給 Q target net。這樣就就能更有效利的用兩個神經網路。

2. Duel DQN:

Duel DQN 將 advantage 和 value 分別考慮。在玩 breakout 時，球一旦從板子彈出去後，我們比較關心的是球的動向(observation/state)。球在上面打的時候，下面板子的動作不影響得分。當球準備回來並且撞到地面時，我們關注的就是選擇動作(action)。因此，在 Duel DQN 中分成兩個 channels 來訓練，以期得到更好的效果。而下圖分別是 DQN 跟 Duel DQN 的架構比較:



圖八、Duel DQN architecture[3]

3. Experiment

這部份我做了兩組的實驗，分別是 Double DQN (gamma = 0.95)與 Nature DQN (gamma = 0.95)的比較和 Duel DQN (gamma = 0.99)與 Nature DQN (gamma = 0.99)的比較。

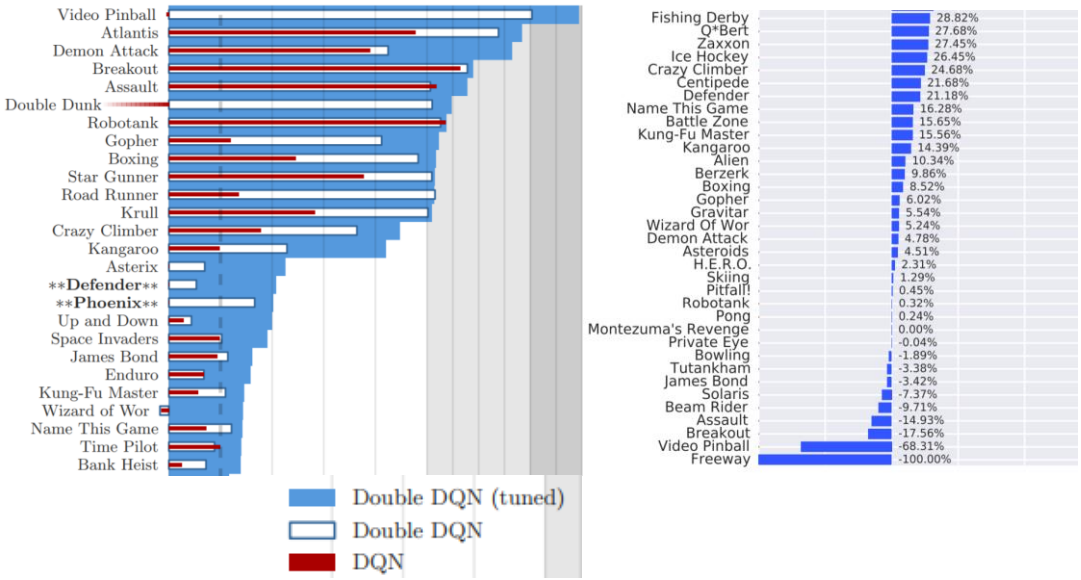
---

[2] https://www.slideshare.net/carpedm20/ai-67616630

[3] https://www.csie.ntu.edu.tw/~yvchen/f106-adl/doc/171204+171207_DeepRL2.pdf

圖九、實驗比較

圖九為他們的訓練過程。第一組的比較可以發現 Double DQN(藍線)沒有特別好，但是線的起伏比較穩定。第二組實驗可以看到 Duel DQN(綠線)比起 Nature DQN(紅線)更早訓練起來，不過值得注意的是到了 9 萬多個 episode 時，average reward 開始往下掉了，而 Nature DQN 依然維持一定的水平。綜合來講，Nature DQN 在 breakout 這個遊戲已經算不錯了，讓其它 improvement 不是那麼的顯著。網路上[4]也有人對 breakout 做更細的 reward 觀察，都可以發現 Nature DQN 表現是還不錯的。



圖十、Double[5](左) \ Duel[6](右) DQN paper 中表現比較

圖十為原論文的實驗比較圖，breakout 在兩種 improvement 中都沒有特別顯著的進步。我去看了一下那些有顯著進步的遊戲，都比較複雜，例如畫面比較豐富或者 action 比較多(兩維以上的移動等等)。因此可能是 breakout 比較單純，讓進步的空間在簡單的修改模型下沒有明顯改善。

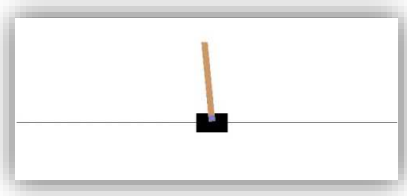
[4] https://github.com/devsisters/DQN-tensorflow

[5] https://arxiv.org/pdf/1509.06461.pdf

[6] https://arxiv.org/pdf/1511.06581.pdf

Implement other advanced RL method, describe what it is and why it is better:

這題我想嘗試 Actor Critic 並與 DQN 做比較，不過玩在 breakout 上有點久所以用 Cartpole 這款遊戲測試，訓練十分鐘就能看到成效了!



```
observation space: Box(4,)
action space: Discrete(2)
reward range: (-inf, inf)
observation: [ 0.04299299 -0.22234843 -0.02266865  0.33373101]
```

圖十一、Cartpole 遊戲畫面及相關資訊

Cartpole 試一個立竿子的遊戲，如圖十一。透過左右控制 (action size = 2)，來維持小竿子的平衡，環境給的 observation 有四個維度，代表的意義分別是 (position of cart, velocity of cart, angle of pole, rotation rate of pole)。Reward 只要在是立著的時候都是 1，所以小竿子立越久分數越高。

Actor Critic:

Actor Critic 是一種結合 policy-based 和 value-based 的模型。我們可以看成是訓練一個 actor (policy-based)和 critic (value-based)。與一般的 policy-based 不一樣的是，critic 的出現可以時時更新 model，而不是玩玩一個 episode 才更新。整個模型需要訓練兩個神經網路，Actor 輸入 state，輸出 action (左或右)。而 Critic 則是評價 action，輸出估計的 value。兩個神經網路的架構如下圖:

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 24)                120
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 50
=================================================================
Total params: 170
Trainable params: 170
Non-trainable params: 0
_________________________________________________________________
```

圖十二、Actor network

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 24)                120
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 25
=================================================================
Total params: 145
Trainable params: 145
Non-trainable params: 0
_________________________________________________________________
```

圖十三、Critic network

因為遊戲蠻簡單的，所以網絡的架構都很簡單。訓練過程如圖十四，並且與 DQN 玩 Cartpole 做比較。可以發現 DQN 依然強悍，在 100 多個 episodes 就能玩到 500 分(max score)。A2C 在這邊收斂的比較慢，我想應該是需要訓練兩個網路使得對資料量的要求較大，要玩比較多 episode。