ADLxMLDS Hw3 Report

B03902059 紀典佑

Basic Performance

1. Model Description:
   I. Policy Gradient: 這次實作 policy gradient 的 model 使用了兩層的 CNN 以及兩層的 Fully Connect，各層之間的 activation function 是使用 relu，CNN 的參數入下圖所示:

```
self.conv1 = nn.Conv2d(1, 16, kernel_size=8, stride=4)
self.conv2 = nn.Conv2d(16, 32, kernel_size=4, stride=2)
self.linear1 = nn.Linear(2048, 128)
self.linear2 = nn.Linear(128, 6)
```

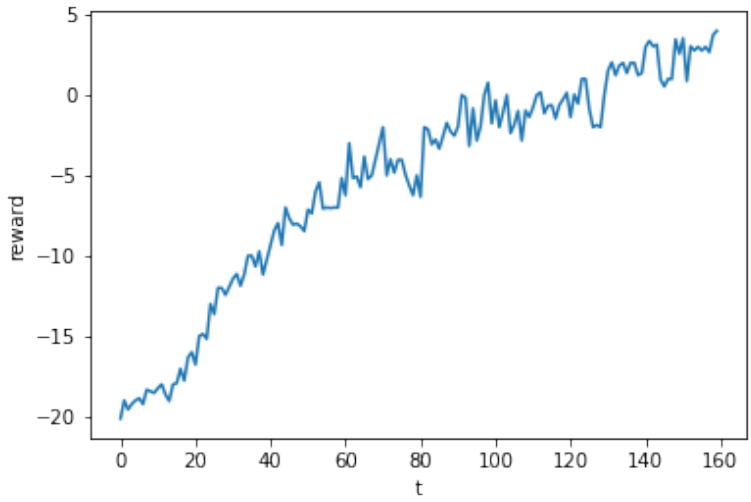
   input 是兩個 environment observation 之間的差 ($S_t - S_{t-1}$)，最後一層再通過 softmax 作為 output，取得各個 action 的機率，將此機率當作 distribution 去 sample 出下一個 action。

   II. DQN: 實作 DQN 的 model 則是使用了三層 CNN 以及兩層 Fully Connect，除了第一層 Fully Connect 的 activation function 使用 Leaky relu，其他曾皆採用 relu，CNN 的參數如下圖所示：

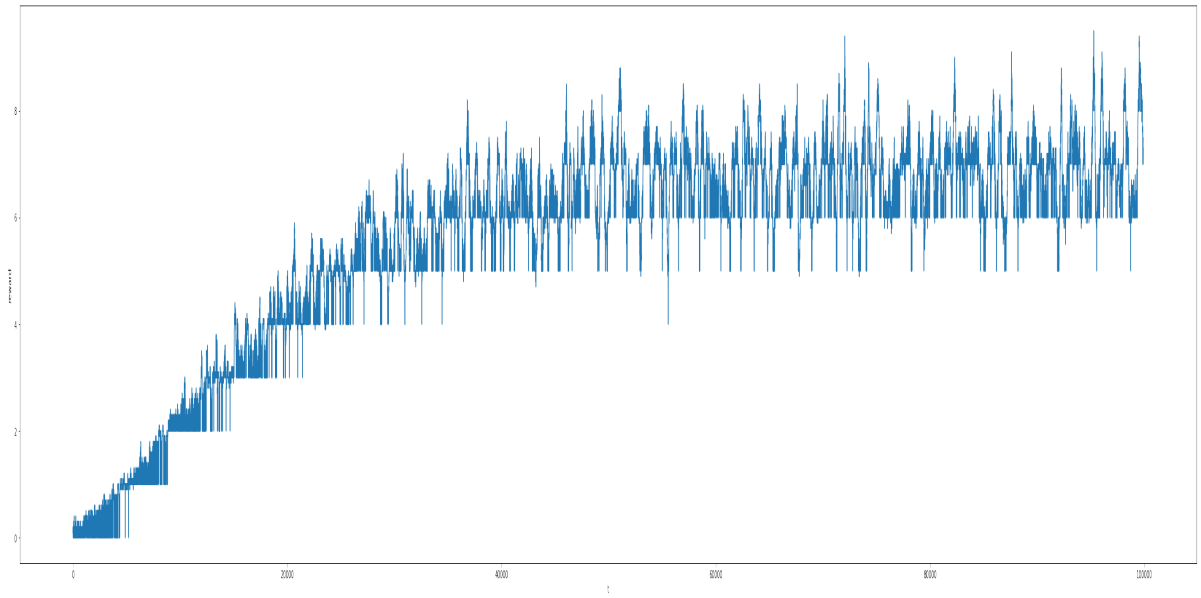
```
self.conv1 = nn.Conv2d(4, 32, kernel_size=8, stride=4)
self.conv2 = nn.Conv2d(32, 64, kernel_size=4, stride=2)
self.conv3 = nn.Conv2d(64, 64, kernel_size=3, stride=1)
```

   input 是 env.step 產生出來的 observation([84, 84, 4])，output 則為各個 action 的 Q value。

2. Policy Gradient Learning Curve:
   Learning Curve 震盪情形滿嚴重的，但是整體 reward 的趨勢是在上升的，區間下降的原因我猜想是因為那幾次的 episode 剛好跟原本的 distribution 相差有些大，造成了錯誤的學習以至於降低了 reward，但是由於整個目標(π)的設定，掉下去後還是會慢慢升回來。

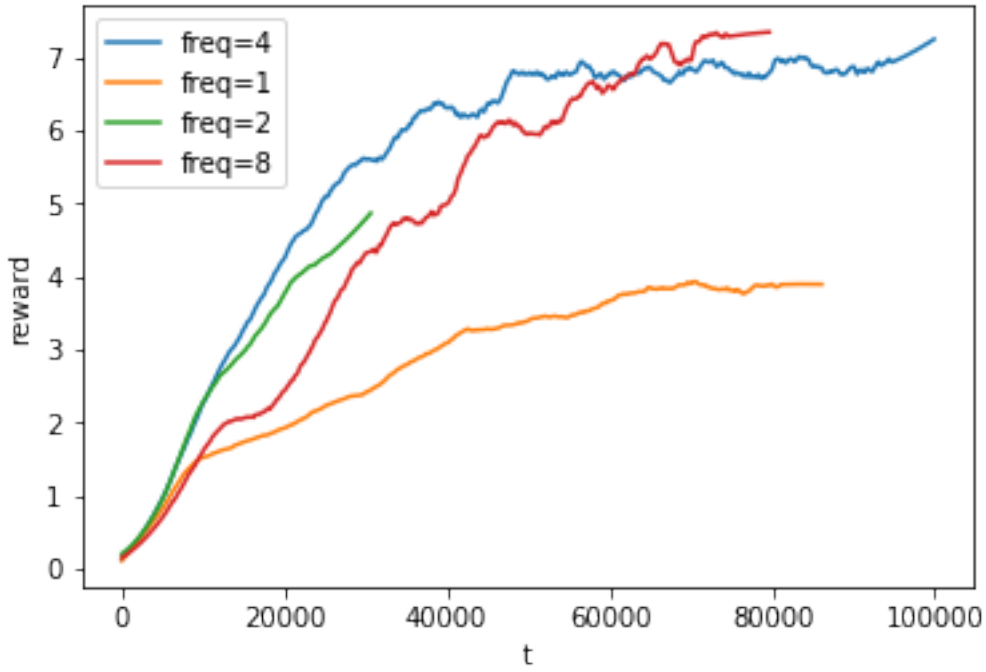


3. DQN Learning Curve:
   從圖片可以看出 reward 從開始 training 時持續上升直到約 4M 個 time step(每一個 action)左右，最後在 7~8 上下左右振盪(clip 過的)。



Experimenting with DQN hyperparameters

1. Experiment Curve:



2. Explain: 我選擇的參數是 learning frequency，也就是每幾個 step 更新一次 Q network(not target)，助教給的參數是 4，而我猜想更新頻率可能影響收斂的速度，因為每一步更新，可以將 memory 裡面的 data 有更多的機會能被 sample 到，原本每 4 步更新一次有些 data 可能存在 memory 內但是從未被 sample 出來過，如果將 frequency 調大就會有更多的 data 沒被 sample 過，因此 frequency 小每次 sample 出來的 data 比較接近但可能會 variance 過小，結果並沒有選 frequency 較大的好。而實際的實驗結果顯示 freq 較小的沒辦法很快 train 到較高的 reward，而 freq 較大的學習的成果跟預期的一樣，比小的好。