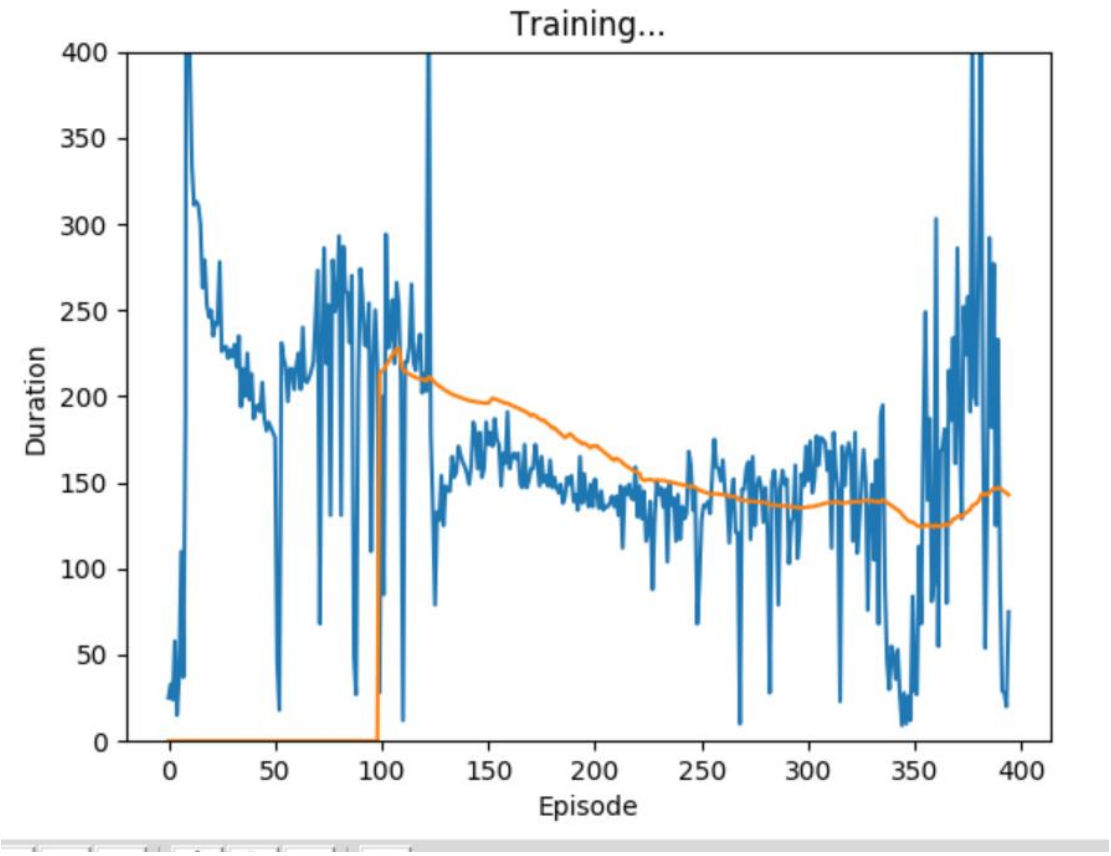
清華大學資工系

大二 學號: A053095 林柏淵

1000+400 episode 前面 1000episode 沒有截到圖所以再 train 400 episode 再截下來



Describe your implement of network structure & Loss function

我用的 loss function 是 adam。然後 network structure 就跟講義上一樣兩層 fully connected，其分別為第一層(4, 32)加上 RELU 和第二層(32, 2)，其中 input size 是 4 由 Cart Position, Cart Velocity, Pole Angle, Pole Velocity at Tip 組成，再來 2 個 output 為向左和向右的指令。

Describe how you implement the training process of deep Q-learning

先 select action 然後 environment 再根據 action 做出 next state，再把狀態全部存到 memory 裡，再來會有兩個 network 一個負責 current q-value 一個負責 next q-value 然後更新其中一個 network 等到 50 個 episode 一個循環再把另一個 network 做更新。

Describe the way you implement of epsilon-greedy action select method

```
def select_action(state):
    global steps_done
    global EPS_START
    sample = random.random()
    #eps_threshold = EPS_END + (EPS_START - EPS_END) * math.exp(-1. * steps_done / EPS_DECAY)
    eps_threshold = EPS_START*EPS_DECAY
    EPS_START = eps_threshold
    if eps_threshold<EPS_END:
        eps_threshold=EPS_END
    steps_done += 1
    if sample > eps_threshold:
        with torch.no_grad():
            return Action_Net(Variable(state).type(FloatTensor)).data.max(1)[1].view(1, 1)
    else:
        return LongTensor([[random.randrange(2)]])
```

紅框這邊會判斷說此次的 select 會交給 NN 判斷 action 還是會隨機判斷 action 的決定。

Describe how the code work (the whole code)

首先會先 import openAI 寫好的環境 gym 再來指定要 CartPole-v0，再來設定 Replaymemory 的 class 用來記錄狀態;
Network 本身上面提過的有兩層;
負責畫圖的 class 用來顯示每個 episode 總共撐過多久的時間的 plot_durations;
負責選擇要往右還往左的動作的 select_action;
接下來是實際 run 的部分一開始會先讓 environment 重置，再來進入一次遊戲回合，迴圈裡面每一個 time step 都會生成 frame 而時間流逝後連續的 frame 就會形成滑車的動畫，接下來就是上面敘述過的 training 過程(select action 等.....)了。