ADL2015 Hw5 Report

姓名：顏毓均
系級：機械五
學號：B01502069

Explain why DQN algorithm need these function and how you implement them:

Experience replay (1%)

Randomizes over the data, thereby removing correlations in the observation sequence and smoothing over changes in the data distribution. [1]
在進行RL的時候，我們的資料的分佈會隨著新演算法的行為而改變，但一般的深度學習輸入的data分布是固定的，因此會造成問題。為了解決這個問題，我們會隨機抽樣上一次的結果，因此利用過去的行為來 smoothing training 的 distribution.

Target network (1%)

Use a separate network to estimate the TD target. This target network has the same architecture as the function approximator but with frozen parameters. Every T steps (a hyperparameter) the parameters from the Q network are copied to the target network. This leads to more stable training because it keeps the target function fixed (for a while). [2] Target network 和 Q network 有一樣的參數，只是他是每 T 個步驟才複製一次，這使得參數的更新更加穩定，因為不會每一個step就更新一次，可能會拿到不好的結果。

1. Avoid oscillations
2. Break correlations between Q-network and target

Epsilon greedy (1%)

Randomly selecting actions, without reference to an estimated probability distribution, is known to give rise to very poor performance. One such method is epsilon greedy, when the agent chooses the action that it believes has the best long-term effect with probability, and it chooses an action uniformly at random, otherwise. [3] 基本上這是一種機率的方法，讓agent有些時候隨機做一些決定，研究者發現這有助於提升整個系統最後的結果。

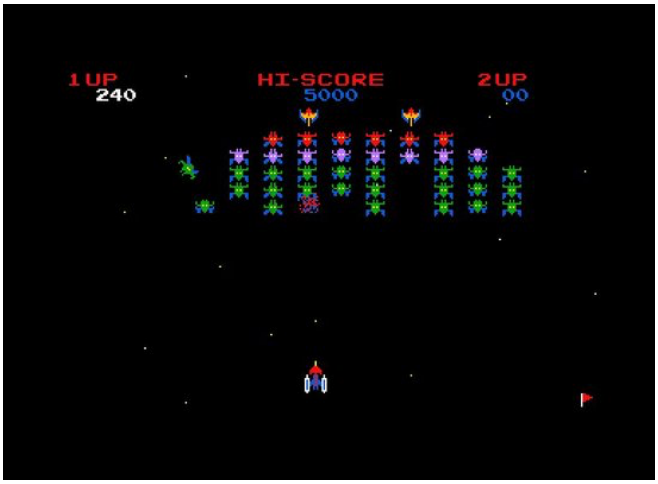
Clip reward (1%)

因為每一種遊戲的reword不一樣，clip reword 就是一個將Reword clips 到[-1,+1]的機制。這可以避免Q-value 太大, 也使得對於不同遊戲 learning rate的調整可以更一致。但壞處是會讓我們喪失精準的reward值。

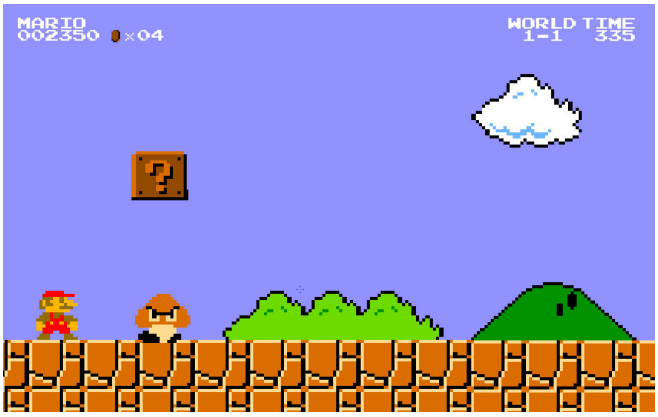
If a game can perform two actions at the same time, how will you solve this problem using DQN algorithm ?(there's no absolute answer) (2%)

可以分成兩種情況：

第一種情況兩個按鍵之間沒有很critical 的關係，例如一個二維開飛機打飛碟的遊戲，左右移動的按鍵可能跟射擊的按鍵沒有很密切關係，我們可以分開來train. 因為射擊可能跟上方飛碟的位置比較有關係。



第二種情況是例如馬力歐，前進後退跟跳躍的按鍵，因為我們可能要邊跳邊往前，不然就會掉到深谷裡去。所以我們可以把兩個指令做排列組合，例如：往前＋跳越，往後＋跳越等等。然後將這些組合集當做可能的單一的指令。再下去用原本單一指令的model來train。因為這兩個指令之間關係密切，所以合在一起train結果會比較好。



（Discuss with teammates)

Reference:

[1] http://home.uchicago.edu/%7Earij/journalclub/papers/2015_Mnih_et_al.pdf
[2] https://github.com/dennybritz/reinforcement-learning/blob/master/DQN/README.md
[3] https://en.wikipedia.org/wiki/Multi-armed_bandit#Semi-uniform_strategies