資工四陳柏均
B02902019

Experience replay:
打破在訓練過程中可能會有的重複性,如果在訓練過程中樣本都差不多的時候,model 很容易回不來,所以偶爾拿已經跑過的舊資料訓練可以打破這個循環.
batch = replayMemory.drawBatch(32)
dqn.train(batch, totalstep)
提取之前的記憶來去訓練

Epsilon greedy:
利用 Epsilon greedy 我們可以讓每次做選擇的時候有一定的機率隨機做出選擇.讓訓練的過程中可以不斷的了解每一種選擇的獎勵.

if random.random() > (1 - epsilon):

    隨機走

else:

    根據 model 做出選擇

Clip reward:
為了避免太高分的 q value,和減少 gradients 暴衝或是爆減的機率,我們只要把獎勵的範圍限定再一個空間之內就好了.
quadratic_part = tf.clip_by_value(difference, 0.0, 1.0)

Target network:
避免他訓練的時候會亂走,來回反覆導致最後 model 根本沒有太多變化,解決問題的方法就是我們讓他每過一段時間再改變.

if stepNumber % self.targetModelUpdateFrequency == 0:

self.sess.run(self.update_target)

Q: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)

A:我會把所有可以的選擇做一次 CN 取二,讓 output layer 直接去預測到底哪個組合最適合.