National Taiwan University Applied Deep Learning
Assignment 5
Implement an agent to play Atari games using Deep Reinforcement Learning
電機工程學系五年級 李佳軒 B01901073

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

Algorithm 1: deep Q-learning with experience replay.
Initialize replay memory D to capacity N
Initialize action-value function Q with random weights θ
Initialize target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$
For episode = 1, M **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  For $t = 1$,T **do**
    With probability ε select a random action a_t
    otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-}) & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters θ
    Every C steps reset $\hat{Q} = Q$
  End For
End For

Experience replay (1%)
因為在玩遊戲的過程中，對序列連續的資料做學習沒有甚麼用處，他們彼此之間並非獨立很有可能前後有很大的關連性。因此為了破除這一點我們用了 Experience Replay 這個機制，在實作上我們用了將過去一百萬筆 transition(St, At, Rt+1, St+1)儲存起來。在每一輪作 Q learning update 時，我們隨機從 memory 裡面選用 mini-batch 數目的 transitions 來做 gradient，這麼一來每次參數更新時，我們就可以使用彼此之間具有獨立特性的資料，使用一定以上數量的資料也確保我們可以做更為 smooth 的 learning 避免少數極端樣本影響模型訓練走向。

Target network (1%)

跟一般監督學習不同的是，在 Q learning update 時

$$L(w) = \mathbb{E}_{s,a,r,s' \sim D}\left[\left(r + \gamma \max_{a'} Q(s', a', w^{-}) - Q(s, a, w)\right)^2\right]$$

我們需要的 target value(r+gamma*Q)是會依據模型參數而變化的，為了要盡量固定 target value，我們另外使用一組 target network 地參數來計算訓練所需的 target value，這個 target network 會不頻繁地更新到正在訓練的 Q network。

epsilon greedy (1%)

在一個只有 partially observed 的環境中，agent 面臨兩種選擇，exploration 以及 exploitation。Epsilon greedy 代表 agent 有 epsilon 機率會隨機的從可採取的行動中選一個，(1-epsilon)的機率會採取目前 policy 評估最好的動作。這麼做的原因在於保持模型適當的探索，去找到更加良好的遊玩方式。

clip reward (1%)

在這個模型訓練過程中，穩定度扮演很關鍵的角色，所以我們去為 reward 的最大最小值做出硬性規定，這樣可以確保不會出現極端的 Q value，也可以讓 gradient 被良好的限制住，不要震盪。

If a game can perform two actions at the same time, how will you solve this problem using DQN algorithm? (2%)

我們建構一個新的 action space，新動作是由舊的 action space(單一動作的 action space)中兩兩一組構成。