DSAI - HW4 report

P76061425 林聖軒

Algorithm

Double Deep Q Network With Prioritized Experience Replay

Experience Replay 可以達到打破數據關聯性來幫助神經網路學習，naive 的 Experience Replay 是使用均勻採樣的方式，但各 transition 的重要程度其實不盡相同，因此此作業使用 Prioritized Experience Replay 來改變 DQN 的 Replay Memory 中不同 transition 被 sample 到的機率，提高較重要之 transition(TD-error 高的 transition)被 sample 到的機率，使得學習能夠更有效率，模型使用 Double DQN，並使用 Sum Tree 來實作機率採樣。

參考原文:https://arxiv.org/pdf/1511.05952.pdf

Algorithm 1 Double DQN with proportional prioritization

Input: minibatch k, step-size η, replay period K and size N, exponents α and β, budget T.
Initialize replay memory $\mathcal{H} = \emptyset$, $\Delta = 0$, $p_1 = 1$
Observe S_0 and choose $A_0 \sim \pi_\theta(S_0)$
for $t = 1$ **to** T **do**
  Observe S_t, R_t, γ_t
  Store transition $(S_{t-1}, A_{t-1}, R_t, \gamma_t, S_t)$ in $\mathcal{H}$ with maximal priority $p_t = \max_{i<t} p_i$
  if $t \equiv 0 \mod K$ **then**
    for $j = 1$ **to** k **do**
      Sample transition $j \sim P(j) = p_j^\alpha / \sum_i p_i^\alpha$
      Compute importance-sampling weight $w_j = (N \cdot P(j))^{-\beta} / \max_i w_i$
      Compute TD-error $\delta_j = R_j + \gamma_j Q_{\text{target}}(S_j, \arg\max_a Q(S_j, a)) - Q(S_{j-1}, A_{j-1})$
      Update transition priority $p_j \leftarrow |\delta_j|$
      Accumulate weight-change $\Delta \leftarrow \Delta + w_j \cdot \delta_j \cdot \nabla_\theta Q(S_{j-1}, A_{j-1})$
    end for
    Update weights $\theta \leftarrow \theta + \eta \cdot \Delta$, reset $\Delta = 0$
    From time to time copy weights into target network $\theta_{\text{target}} \leftarrow \theta$
  end if
  Choose action $A_t \sim \pi_\theta(S_t)$
end for

Game

Mountain Car (OpenAI gym)

Observation:

Num	Observation	Min	Max
0	position	-1.2	0.6
1	velocity	-0.07	0.07

Reword Setting:

pos = (new_state[0] + 1.2) / 0.9 - 1
vel = abs(new_state[1]) / 0.035 -1
(Clipped Car Position and abs(Car Velocity) within [−1, 1] .)
$\boxed{\text{reward = pos + vel}}$

Parameter Settings

Episode : 100
Timestep per Epoch : 500 (limited timestep in case the car can't climb to the target.)
learning_rate : 0.005
Optimizer: Adam

Replay Memory (Sum Tree):

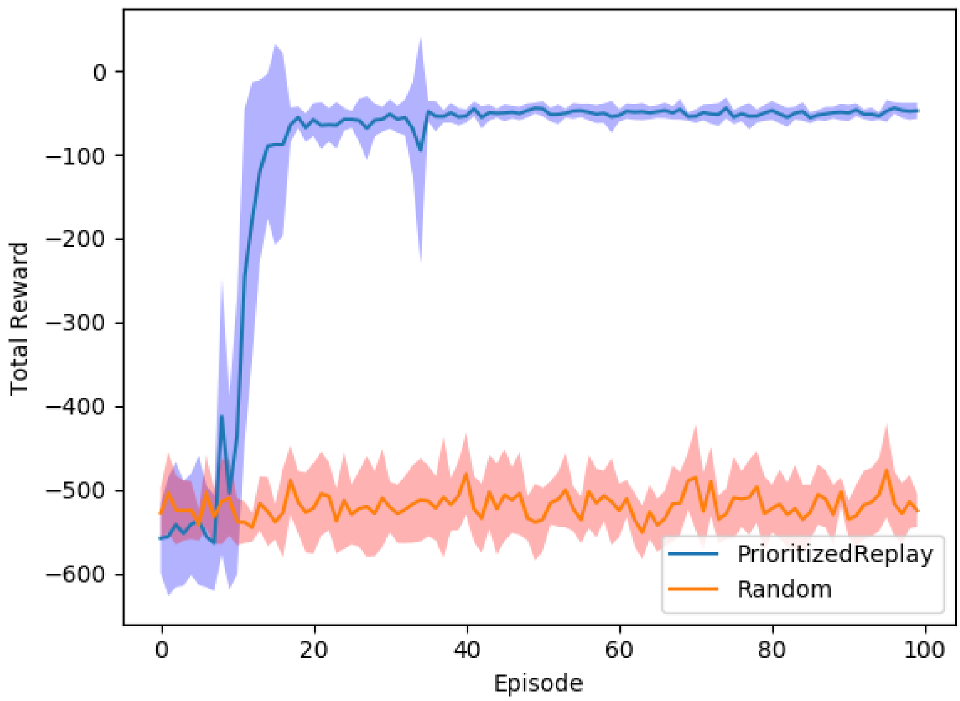
Memory Size : 4000
Batch_size : 32 / 64
epsilon : 1.0
(epsilon-greedy)
epsilon_min : 0.01
epsilon_decay : 0.995

gamma : 0.85
alpha : 0.6
beta : 0.4
epsilon : 0.01
(TD-error + epsilon prevent error is zero.)

Learning curve

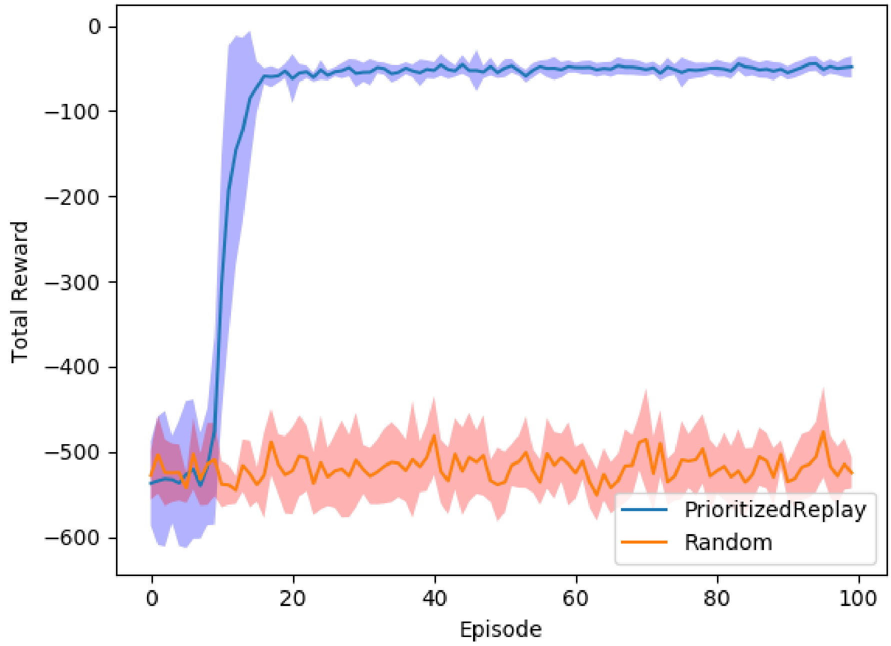
使用 Double Deep Q Network With Prioritized Experience Replay
和僅作 Random action 依據 Episode 所獲得的 reward 來作比較
淺色面積部份為 10 次取樣的標準差

1.Total Reward (batch size = 32)



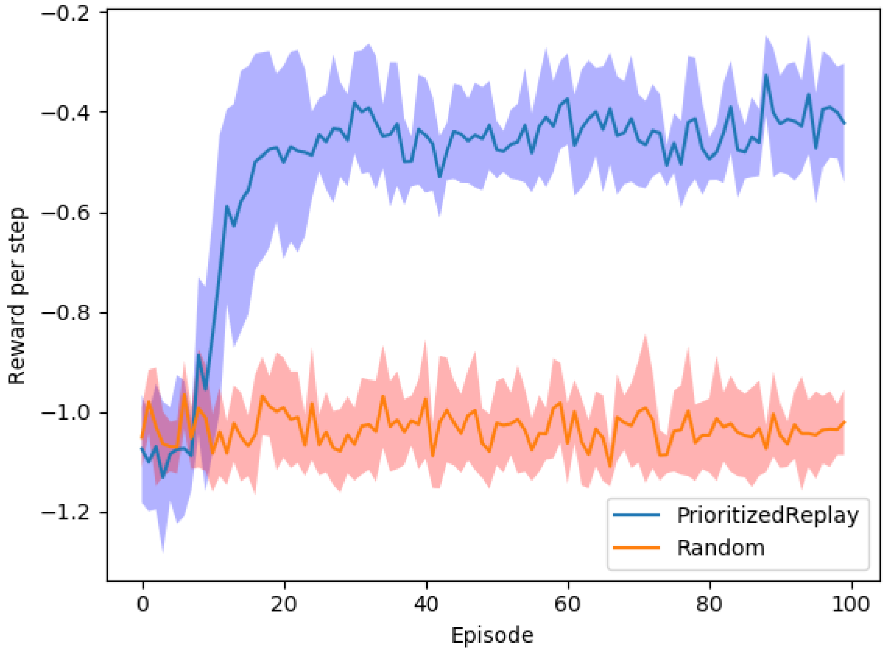
Prioritized Experience Replay 理所當然的比僅採取 Random action 的 total reward 要高得許多。(Prioritized Experience Replay 方法 Memory size 在到達 4000 個 transition 前仍在收集經驗)

2.Total Reward (batch size = 64)



此圖可以觀察到 Prioritized Experience Replay 使用 batch size 為 64 時較上圖 batch size 為 32 的表現更為穩定。

3.Reward per step



1. What kind of RL algorithms did you use? value-based, policy-based, model-based? why?

使用 Double Deep Q Network With Prioritized Experience Replay，此為 value-based 的方法。Experience Replay 可以達到打破數據關聯性來幫助神經網路學習，再加上使用 Prioritized Experience Replay，能使學習更加有效率。

2. This algorithms is off-policy or on-policy? why?

此方法為 off-policy，因為是使用 Replay Memory 中的經驗來去學習並更新 policy，Replay Memory 中的 transition 含有各種之前的 policy，而不是使用當前的 policy。

3. How does your algorithm solve the correlation problem in the same MDP?

Experience Replay 使用 Replay Memory 中的 transition 作機率採樣，採樣樣本中會包含新的及舊的 transition，而不是依照時間順序，此舉可以達到打破數據關聯性來幫助神經網路學習