Comparative Anlysis of DQN and Dueling DQN on Atari Pac-Man

Hsiao Ching Teng
Department of Electrical Engineering,
National Tapei University,
New Tapei, Taiwan
norateng1339@gmail.com

Zu Xiang Chen
Department of Electrical Engineering,
National Tapei University,
New Tapei, Taiwan
pgn63117@gmail.com

***Abstract*—This work explores and compares the performance of Deep Q-Network (DQN) and Dueling DQN in the context of the Atari game Ms. Pac-Man. Both methods are implemented using convolutional neural networks (CNNs) to process preprocessed grayscale images with stacked frames. The results demonstrate that the Dueling DQN, by separating state value and action advantage, achieves superior learning efficiency and higher cumulative rewards in this dynamic, high-dimensional environment.**

Keywords—CNN, DQN, Dueling DQN, Reinforcement Learning , Atari, Ms. Pac-Man

I. INTRODUCTION

Pac-Man 是 1982 年推出的經典街機遊戲，現由 Atari Learning Environment (ALE) 提供作為強化學習研究的平台。玩家需控制 Pac-Man 吃豆子（+10 分）、能量豆（+50 分）與水果（+100 至 +5000 分），同時避開四隻鬼怪（Blinky、Pinky、Inky、Clyde）。若吃下能量豆，可在短時間內反吃鬼怪，獲得 +200 至 +1600 分不等的獎勵。遊戲結束條件為生命耗盡或吃光所有豆子。

本研究比較兩種基於深度學習的強化學習模型：傳統的 DQN 與改良版本 Dueling DQN，透過 Convolutional neural network（CNN）從遊戲畫面學習動作策略，並探討兩者在處理高維觀測與複雜決策情境下的表現差異。

II. METHODOLOGY

A. DQN

DQN 是深度強化學習中的一種經典演算法，結合 Q-Learning 與深度神經網路，並成功解決高維狀態下的強化學習問題。此演算法最初由 DeepMind 提出，於 2015 年論文“Human-level control through deep reinforcement learning”發表。

DQN 是一種使用深度神經網路已近似 Q 值函數 Q(s, a)的強化學習方法，適用於解決高維狀態的強化學習問題。透過核心目標學習策略，並最大化累計報酬。

下式 (1)為目標函數 Q(s, a) 定義：

$$\mathrm{Q(s,a)} = \mathrm{E}\left[\sum_{\mathrm{t=0}}^{\infty} \gamma^{\mathrm{t}} \mathrm{r_{t+1}} \middle| \mathrm{s_0 = s}, a_0 = \mathrm{a}\right] \qquad (1)$$

其中定義 s: 當前狀態、a: 在狀態 s 採取的動作、$\mathrm{r_{t+1}}$: $\mathrm{t+1}$ 步的獎勵、 γ: 折扣因子、E: 期望值。以下為 DQN 演算法流程：

Algorithm 1 Deep Q-Network (DQN) Algorithm

Initialize Q-network $Q(s, a)$
Initialize target network $Q'(s, a)$
Initialize replay buffer D
for episode = 1 to M **do**
  Initialize environment and observe initial state s
  for $t = 1$ to T **do**
    With probability ϵ select a random action a
    Otherwise select $a = \arg\max_a Q(s, a)$
    Execute action a, observe reward r and next state s'
    Store transition (s, a, r, s') into D
    Sample random mini-batch of transitions (s_i, a_i, r_i, s'_i) from D
    Compute target:
    $y_i = r_i + \gamma \cdot \max_{a'} Q'(s'_i, a')$
    Perform a gradient descent step on loss:
    $L(\theta) = (y_i - Q(s_i, a_i, \theta))^2$
    if $t \mod C = 0$ **then**
      Update target network $\theta' \leftarrow \theta$
    end if
  end for
end for

B. Dueling DQN

Dueling DQN 是基於前述 DQN 所提出的新方法，用於解決 DQN 對於不同 action 、不同 state 下感知不敏感的問題, 以及在特定場域下訓練成效不彰等問題。

相較於 DQN 的目標函數 Q(s, a)，Dueling DQN 將 Q 值分成兩個部分: V(s) 和 A(s, a)。其中 V(s)：狀態 s 本身的值（Value）、A(s, a)：在狀態 s 下，選擇動作 a 的優勢（Advantage）。這樣的方法可以更好的判斷狀態的重要性，Q(s, a)的定義如下:

$$\mathrm{Q(s,a)} = \mathrm{V(s)} + [A(s,a) - \frac{1}{|A|}\sum_{a'} A(s,a')] \qquad (2)$$

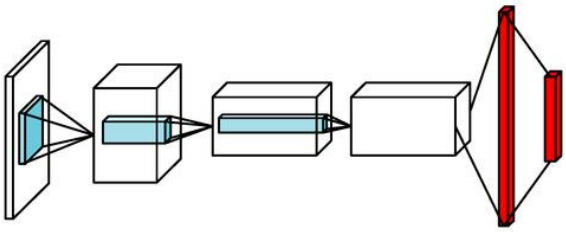
以下為 Dueling DQN 演算法：

Algorithm 2 Dueling Deep Q-Network (Dueling DQN) Algorithm

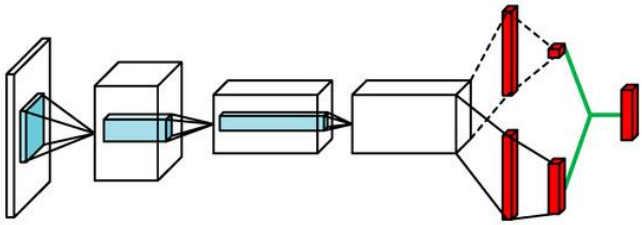
Initialize dueling Q-network $Q(s, a; \theta)$ with state value stream $V(s; \theta)$ and advantage stream $A(s, a; \theta)$
Initialize target dueling Q-network $Q'(s, a; \theta')$ with parameters $\theta' \leftarrow \theta$
Initialize replay buffer D
for episode = 1 to M **do**
  Initialize environment and observe initial state s
  for $t = 1$ to T **do**
    With probability ϵ select a random action a
    Otherwise select $a = \arg\max_a Q(s, a; \theta)$
    Execute action a, observe reward r and next state s'
    Store transition (s, a, r, s') into D
    Sample random mini-batch of transitions (s_i, a_i, r_i, s'_i) from D
    Compute $Q(s_i, a_i; \theta) = V(s_i; \theta) + \left(A(s_i, a_i; \theta) - \frac{1}{|A|}\sum_{a'} A(s_i, a'; \theta)\right)$
    Compute target:
    $y_i = r_i + \gamma \cdot \max_{a'} Q'(s'_i, a'; \theta')$
    Perform a gradient descent step on loss:
    $L(\theta) = (y_i - Q(s_i, a_i; \theta))^2$
    if $t \mod C = 0$ **then**
      Update target network $\theta' \leftarrow \theta$
    end if
  end for
end for

各符號定義如下: Q(s, a) : 在狀態 s 下執行動作的 Q 值、$\sum_{a'} A(s, a')$： 該狀態下所有動作的優勢平均。

下為 DQN 及 Dueling DQN 架構展示：



圖一：DQN 模型架構



圖二：Dueling DQN 模型架構

由上（圖一）、（圖二）對比，可見唯一的差別在紅色區塊，Dueling的輸出會經過兩段全連接層，分別對應 advantage 和 value function，最後兩者合併即為每個動作之 Q value。

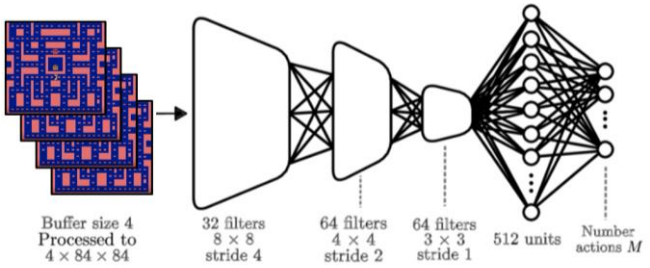
III. Implementation

A. *Environments*

本實驗使用 OpenAI Gym 提供的 ALE/MsPacman-v5 環境，搭配 AtariPreprocessing 畫面預處理，將原始畫面轉為 84×84 的灰階影像，再以 FrameStack 堆疊連續 4 幀，形成輸入張量 shape = (4, 84, 84) ，保留時間連續資訊。動作空間為 5 個離散行為（靜止、上下左右移動）。

B. *CNN*

CNN 在 DQN 中負責處理來自遊戲畫面的原始視覺資訊，將高維圖像轉換為低維的特徵表示，做為後續 Q value 運算的基礎。



圖三：DQN 中 CNN 模型架構

由於 Pac-Man 的輸入為一組經過預處理的遊戲畫面（經灰階處理並堆疊為 4 幀），CNN 能從中提取出如鬼怪位置、豆子分布、通道牆壁等關鍵空間資訊，作用如下：

(1) 特徵擷取（Feature Extraction）：透過多層卷積運算與 ReLU 非線性激活，CNN 能夠捕捉圖像中從低階邊緣到高階物件的抽象特徵，例如辨識哪些區塊是可移動空間、鬼怪是否接近等。

(2) 空間資訊壓縮（Spatial Reduction）：使用卷積核（filters）和池化層（Max Pooling）逐層縮小圖像空間尺寸，同時保留最重要的特徵，降低運算成本。

(3) 時間特徵整合：由於輸入為連續四幀畫面，CNN 亦學習如何從時間上的變化中提取速度、方向等動態資訊，幫助判斷鬼怪是否逼近或水果是否即將消失。

C. *DQN*

使用 AtariNetDQN，包含三層卷積層（32、64、64 通道）和兩層全連接層（512 至 5 個行動的 Q 值），權重初始化使用 He/Kaiming 方法以配合 ReLU 激活函數。最終輸出為對應每個動作的 Q 值。

D. *Replay buffer*

DQN 與 Dueling DQN 均使用「經驗回放機制」（Replay Buffer），將智能體與環境互動產生的轉換記錄（state, action, reward, next_state, done）儲存至記憶體中，再以隨機抽樣的方式取樣進行訓練，減少樣本間的自相關性並模擬監督學習。此設計可大幅提升訓練穩定性。

同時採用固定目標網路（Target Network），每隔固定步數將行為網路的權重複製到目標網路，用以估算下一狀態的最大 Q 值，避免因即時更新導致的目標不穩定。

E. *Dueling_DQN*

使用 AtariNetDuelingDQN，共享卷積層後分為價值函數（Value, 單一輸出）和優勢函數（Advantage, 5 個行動輸出），和並輸出最終 Q value。

這樣的結構能更清楚地學習狀態本身的價值函數，並避免因動作分數差異微小而影響學習效率。

F. *Training*

最佳超參數如下：

Steps: 10,000,000

batch size: 32

gamma rate: 0.99

epsilon: 從 0.99 逐步衰減至 0.1

Replay buffer: 100,000

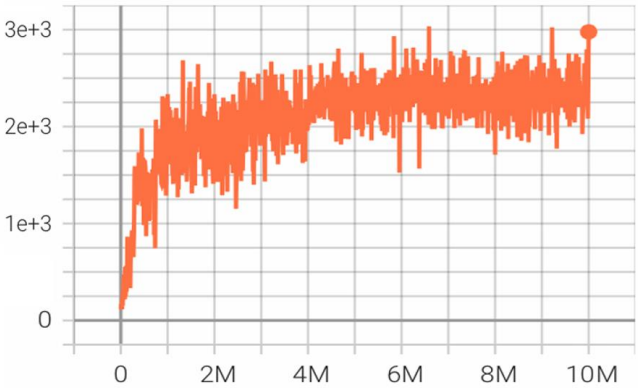
Optimizer：Adam (lr = 1e-4, ε =1.5e-4)

Loss function：MSE Loss

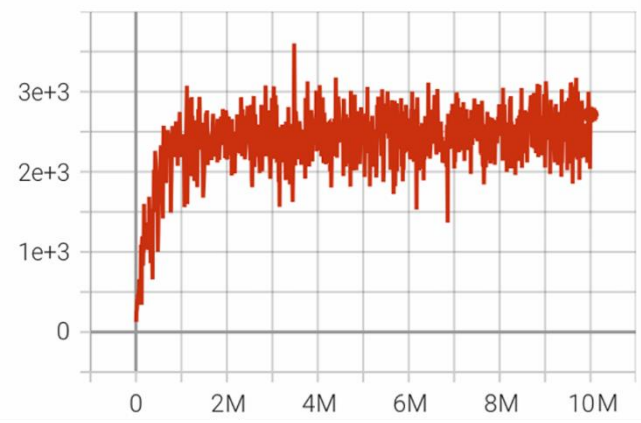
Update behavior network freq：每隔 10000 steps 將行為網路（behavior network）的權重複製到 target network，以穩定 Q 值的更新。

IV. Result and Discussion

本次實驗對比 DQN 和 Dueling DQN 兩模型。實驗結果之 evaluating curve 中對應下（圖四），（圖五），DQN 算法的最高分數為 3030 分，平均約在 2300 分、而 Dueling 最高分數為 4100 分、平均約在 2800 分。 可見 Dueling 成效較佳。



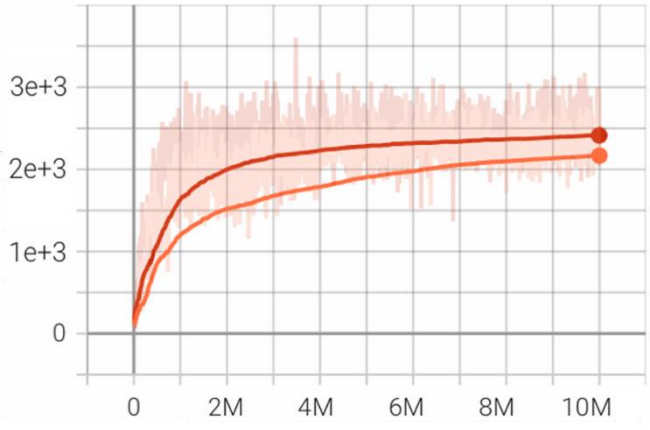
圖四：DQN training curve



圖五：Dueling DQN training curve

Dueling DQN 與傳統 DQN 僅在神經網路設計有所不同，其將 Q value 分開為 Value 與 Advantage function，並透過減去 A 平均值，防止模型可能只改動 A 而使 V 為 0，讓模型更有效率學到哪一 action 在當前 state 下更好。

由下（圖六）對比兩 training curve，可見初期 1M steps 前，Dueling 上升較迅速、較 DQN 更快收斂至穩定。全程 Dueling 分數皆高於 DQN，代表使用 Dueling DQN 嘗試將學習的目標分為 V 與 A 真的加強了模型的學習，因此在 training 上 Dueling 的效果才會更好。



圖六：兩模型之 trainingcurve 對照圖

DQN 與 Dueling 之 testing 比較如下（圖七）、（圖八），在五次測試中之平均分數，Dueling 較 DQN 高 400 分左右，可驗證，Dueling DQN 不管是在 training 與 testing 的結果相比於 DQN 皆好上不少。

```
Episode 1: Reward = 2660.0
Episode 2: Reward = 2240.0
Episode 3: Reward = 2050.0
Episode 4: Reward = 2400.0
Episode 5: Reward = 2360.0
Average Reward over 5 episodes: 2342.00
```

圖七：DQN testing result

```
Episode 1: Reward = 2490.0
Episode 2: Reward = 2300.0
Episode 3: Reward = 2650.0
Episode 4: Reward = 3830.0
Episode 5: Reward = 2630.0
Average Reward over 5 episodes: 2780.00
```

圖八：Dueling testing result

V. Conclusion

本研究針對 Ms. Pac-Man 遊戲場景，比較了傳統 DQN 與 Dueling DQN 在動態強化學習環境中的效能。實驗結果顯示，Dueling DQN 架構透過將 value function 與 advantage function 分離的設計，不僅加快了模型在早期訓練階段的收斂速度，亦在整體學習效率與最終獲得分數上顯著優於傳統 DQN。 這突顯了該架構在複雜決策環境中的潛力，特別是 Q value 學習可能受限的情境下，能夠有效強化策略學習。

未來研究可延伸此方法應用於其他 Atari 遊戲，或引入進階技術如 Prioritized Experience Replay、Noisy Networks 或 Multi-step Learning，進一步提升學習效率與策略穩定性，強化在廣泛強化學習任務中的適用性。

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," arXiv preprint arXiv:1312.5602, Dec. 2013.

[2] A. M. Ahmed, T. T. Nguyen, M. Abdelrazek, and S. Aryal, "Reinforcement learning-based autonomous attacker to uncover computer network vulnerabilities," Neural Comput. Appl., vol. 36, pp. 14341–14360, Aug. 2024

[3] T. A. Spears, B. G. Jacques, M. W. Howard, and P. B. Sederberg, "Scale-invariant temporal history (SITH): Optimal slicing of the past in an uncertain world," arXiv preprint arXiv:1712.07165, Dec. 2017.