第十九屆旺宏科學獎

創意說明書

參賽編號：

作品名稱：Algorithms to play UNO

姓名：吳邦寧

關鍵字：基因算法、強化學習、機器學習

摘要

前人使用Genetic algorithm來訓練神經網路玩UNO，在淺層的神經網路下十分有效，但是在相對深層的神經網路下成效不彰。

細菌使用基因來記憶所學所能，因此，只能完成相對簡單的任務；而人類使用大腦來記憶所學所能，因此，人類能完成相對複雜的任務。受此啟發，何不捨棄基因算法，使用Deep-Q-Learning來設計神經網路呢？

本研究使用五種不同的算法，Deep-Q-Learning algorithm、Genetic algorithm、Naïve offensive/defensive algorithm以及Dummy algorithm進行實驗。

Deep-Q-Learning algorithm對上Dummy algorithm約有的勝率，對上Naïve offensive/defensive algorithm約有/的勝率，對上另外一個Deep-Q-Network的時候，先手約有的勝率，後手約有的勝率。

Genetic algorithm對上Dummy algorithm約有的勝率，對上Naïve offensive/defensive algorithm約有/的勝率，對上另外一個Genetic Network的時候，先手約有的勝率，後手約有的勝率。

Deep-Q-Learning algorithm對上Genetic algorithm，在先手的情況下約有的勝率，在後手的情況下約有的勝率，就算關閉了Q-Learning的學習功能，Genetic algorithm也難以望其項背，可見Q-Learning是較為優秀的演算法。

根據實驗數據得知，使用Deep-Q-Learning能夠有效的增加勝率，印證了「人類比細菌更能處理複雜問題」的事實。

壹、研究動機

我在網路上看到一篇「Notes on machine learning - Playing UNO」，文章中使用Genetic algorithm、Dummy algorithm以及Naïve offensive/defensive algorithm進行實驗。與Dummy Algorithm、Naïve offensive/defensive algorithm相較下，Genetic perceptron的勝率到達左右，文章作者在Genetic perceptron內加入Hidden layers，不過成效不彰。

我認為，成效不彰是因為難以用單純的Genetic fluctuation來訓練神經網路，細菌使用基因來記憶所學所能，因此，只能完成相對簡單的任務；而人類使用大腦來記憶所學所能，因此，人類能完成相對複雜的任務。

Q-Learning好比人腦，Genetic algorithm好比細菌，若將Q-Learning應用在玩UNO上，好比細菌演化成人類，

貳、研究目的

一、探討先手與後手的利與弊

二、探討各演算法之間的勝率

三、探討Deep-Q-Network與Genetic-Network的利與弊

參、文獻回顧

# 一、Notes on machine learning - Playing UNO

<https://john-hearn.info/articles/notes-on-machine-learning-playing-uno>

As previous stated, the article shows several ways to conduct UNO games with neural networks.

# 二、RL - Cards

<https://arxiv.org/pdf/1910.04376.pdf>

It’s an open source library recently published in Feb 2020 for conducting researches of card games with Reinforcement learning. Despite its usefulness, the library can’t meet my customized needs, so I decide to use it as a blueprint to build my own.

# 三、Deep-Q-Learning

肆、研究器材

|  |  |  |
| --- | --- | --- |
|  | 器材 | 用途 |
| 研究器材 | Tensorflow | Lessen the coding complexity for developing. |
| Tkinter | Visualize the gaming process. |
| NVIDIA 1660 Ti | Hardware for increase training performance. |
| CUDA | Utilize GPU to increase training performance. |
| Tensorboard | Plot the win rate. |
| Python | Programming language for conducting the project. |
| Github | Online web service for storaging codes and files. |

伍、研究方法

# 一、遊戲規則

# 二、實驗步驟

# 三、Dummy algorithm

一言以蔽之，這個演算法等同「找一張能丟的卡或抽牌」。

# 四、Naïve Offensive/Defensive Algorithm

激進演算法等同「先丟最強的功能牌，再丟普通牌，沒牌再抽卡」，猶如忿狷之人，有好牌就出，不顧後果；保守演算法等同「先丟普通牌，再丟功能牌，沒牌再抽卡」，猶如杞人憂天，永遠把好牌留在手上。

# 五、Genetic algorithm

演算法如同在一個Cell Petri上模擬許多個Cells，細胞將與目標假人(Dummy)決鬥，根據決鬥結果計算細胞的最終積分，再進行基因擴散(Genetic diffusion)藉此交換基因(Exchange genetic)。

不適應環境的基因會在基因擴散的過程中漸漸消失，適應環境的基因會在基因擴散的過程中壯大，如同達爾文的「適者生存」說，演化過程會淘汰掉不適應的個體，留下「贏家」，也就是適應環境的個體。

# 六、Deep-Q-Learning Algorithm

Q-Learning是一個廣為人知的演算法，相關論文可以在這裡找到，Deep-Q-Learning的概念也相當簡單，只是將Q-Matrix換成一個Neuron-network而已，好處是能夠大幅降低儲存的空間複雜度(Space complexity)，也能夠藉由已知表格預測未知表格的值。

Behavior cloning是一個常見的處理手法，如同徒弟模仿師傅的行為，如同小孩模仿家長的舉止。

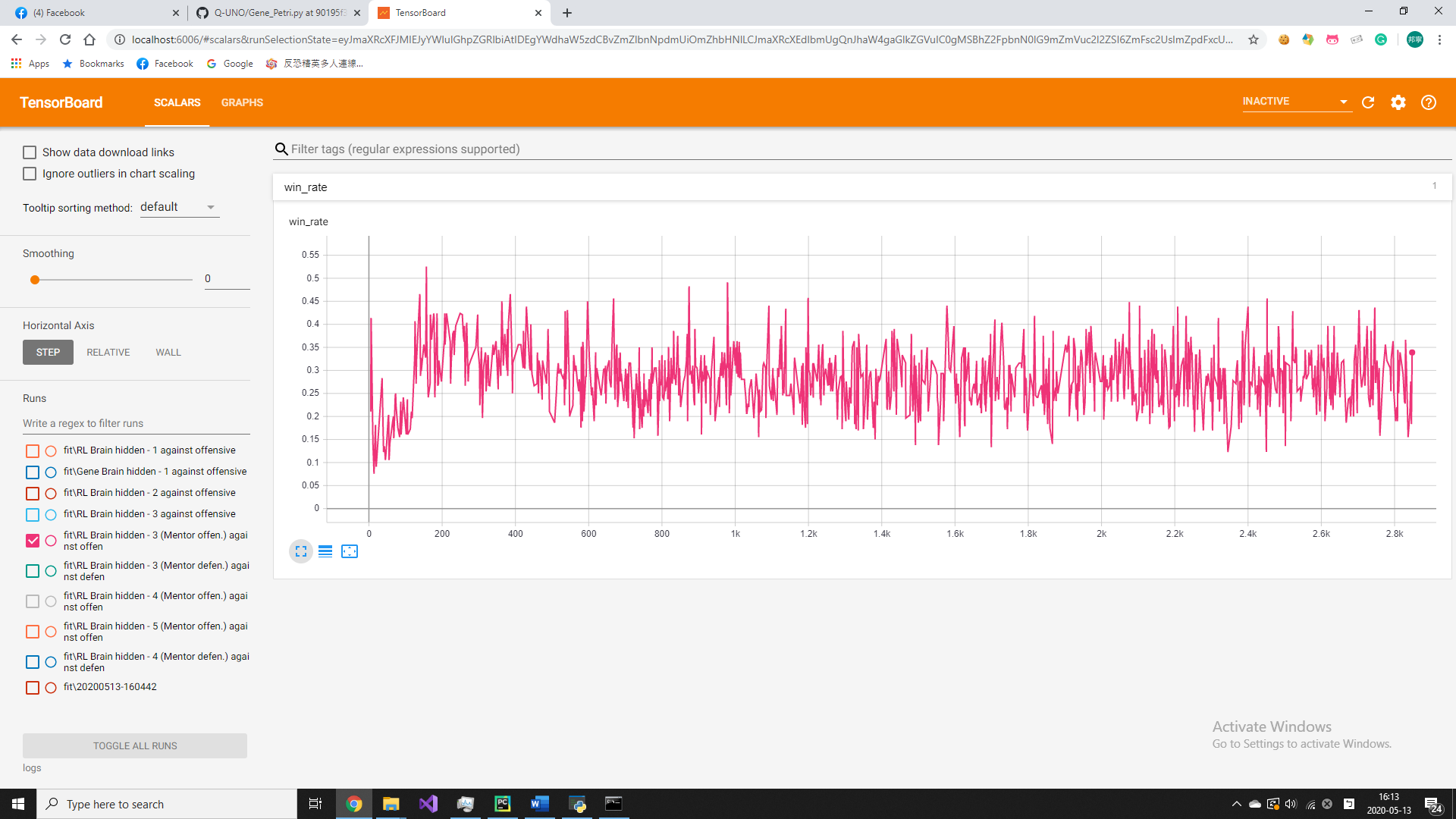
陸、現階段研究成果

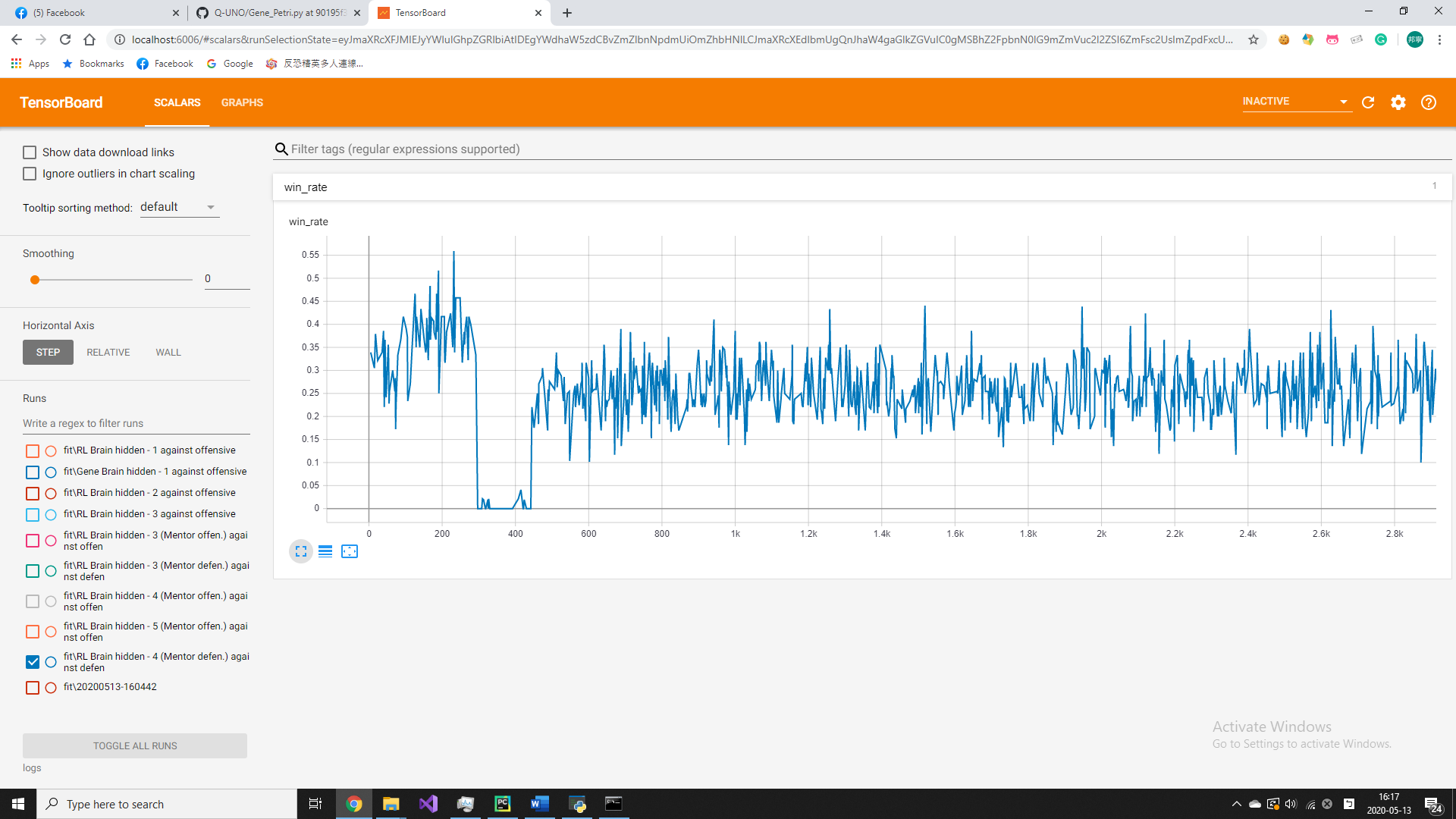
# 一、先手制霸

不論是使用哪一種演算法，先手的勝率都較高，如下表所示。

|  |  |
| --- | --- |
| 演算法 | 先手勝率 |
| Dummy algorithm |  |
| Naïve offensive algorithm |  |
| Naïve defensive algorithm |  |
| Genetic algorithm |  |
| Deep-Q-Learning algorithm |  |

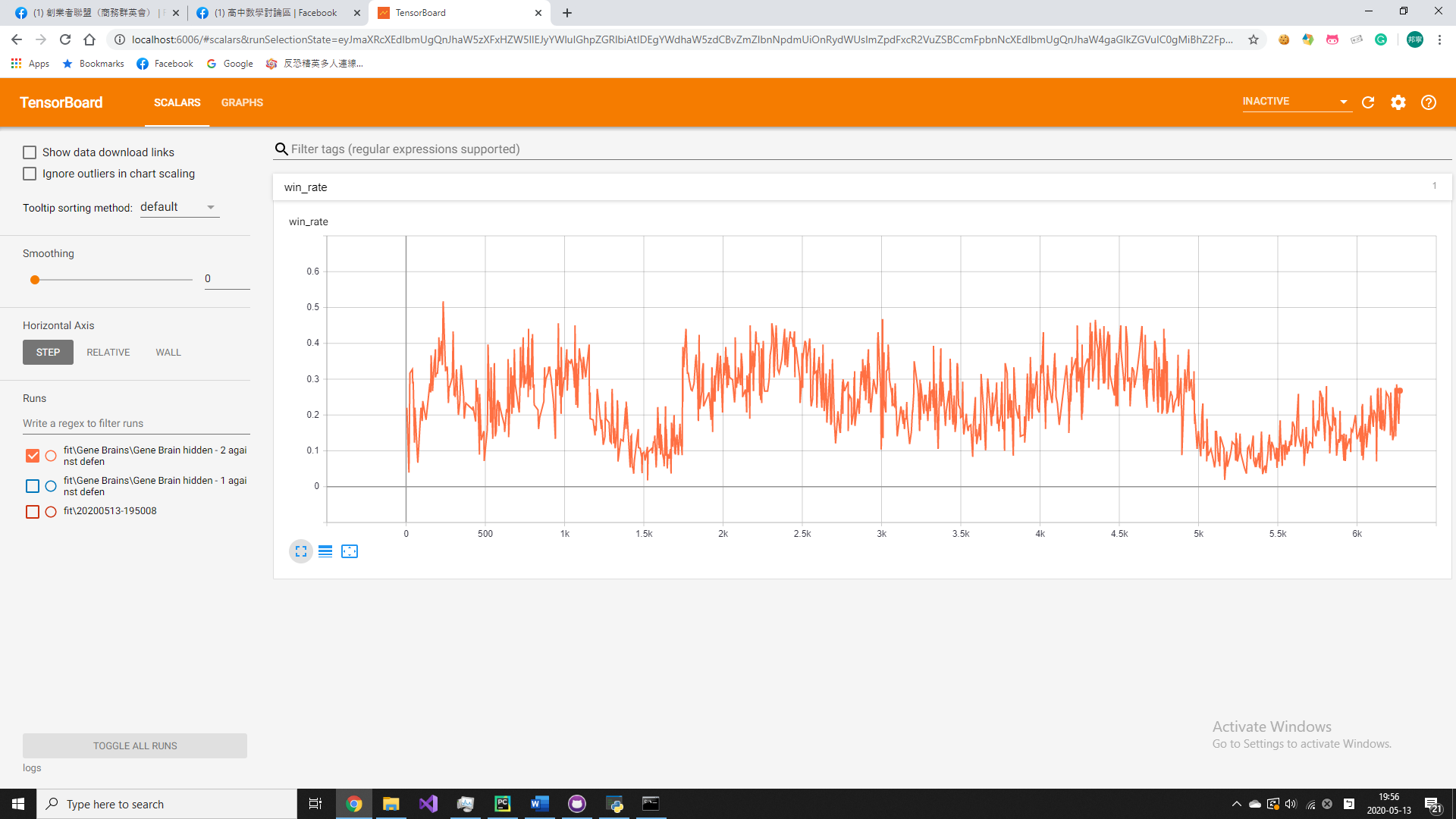
# 二、Deep-Q-Learning的特色 – 恍然大悟或瞬間失憶

 可以見到，Deep-Q-Learning具有「恍然大悟」的潛力，神經網路彷彿突然觀察到某種重要線索，勝率瞬間飆升，如下圖所示。

 相對的，Deep-Q-Learning也具有「瞬間失憶」的潛力，神經網路彷彿突然失憶，把畢生所學給全忘了，勝率瞬間跌落谷底，如下圖所示。

由此可知，Deep-Q-Learning有可能「恍然大悟」，也有可能「瞬間失憶」，不過跟Genetic-Algorithm相較，Deep-Q-Learning是較為穩定的演算法。

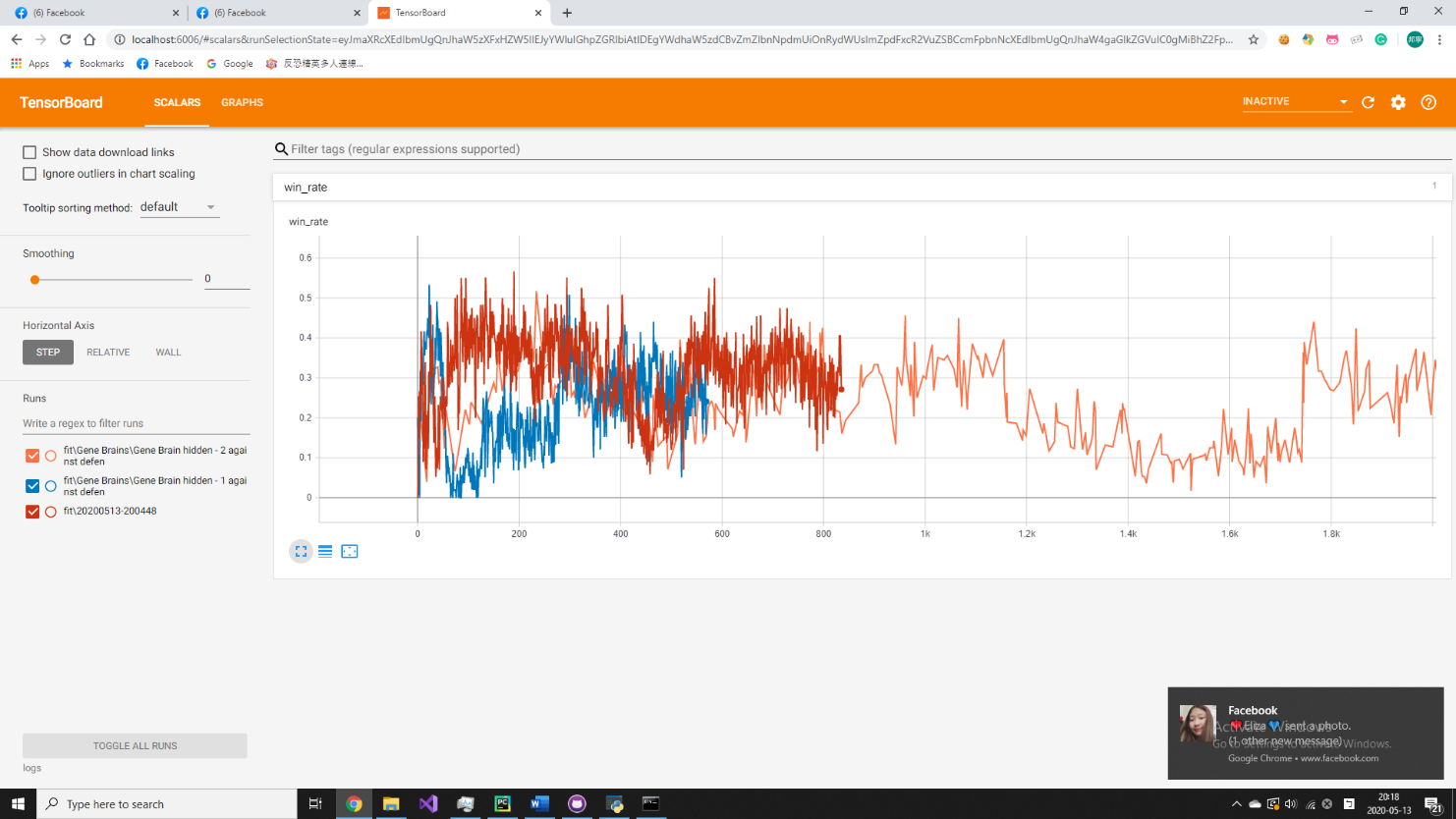
# 三、Genetic Algorithm的特色 – 勝率震盪

 可以見到勝率像是在做週期性震盪，時高時低，如下圖所示。

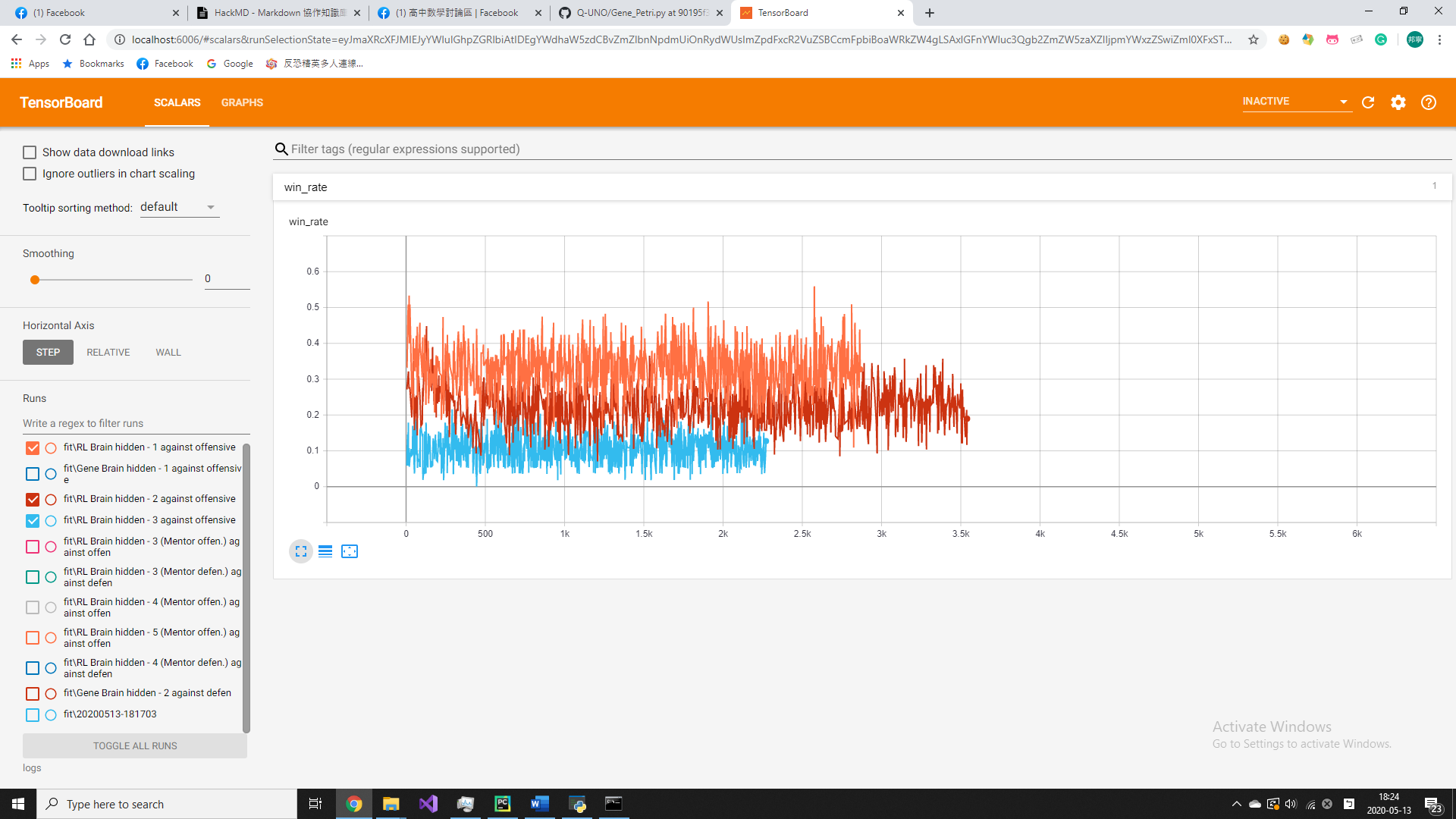
好的基因，適應力強的基因，也就是勝率高的基因，會隨著基因擴散(Genetic Diffusion)快速的散播到培養皿的每個細胞內，使得勝率增加，也就是震盪的上升期。

受到突變(Mutation)的影響，勝率高的基因也會變成勝率低的基因，若是突變影響到關鍵基因，則神經網路需要

# 四、隱藏層深度

基因演算法對隱藏層深度不太敏感，如下圖所示。

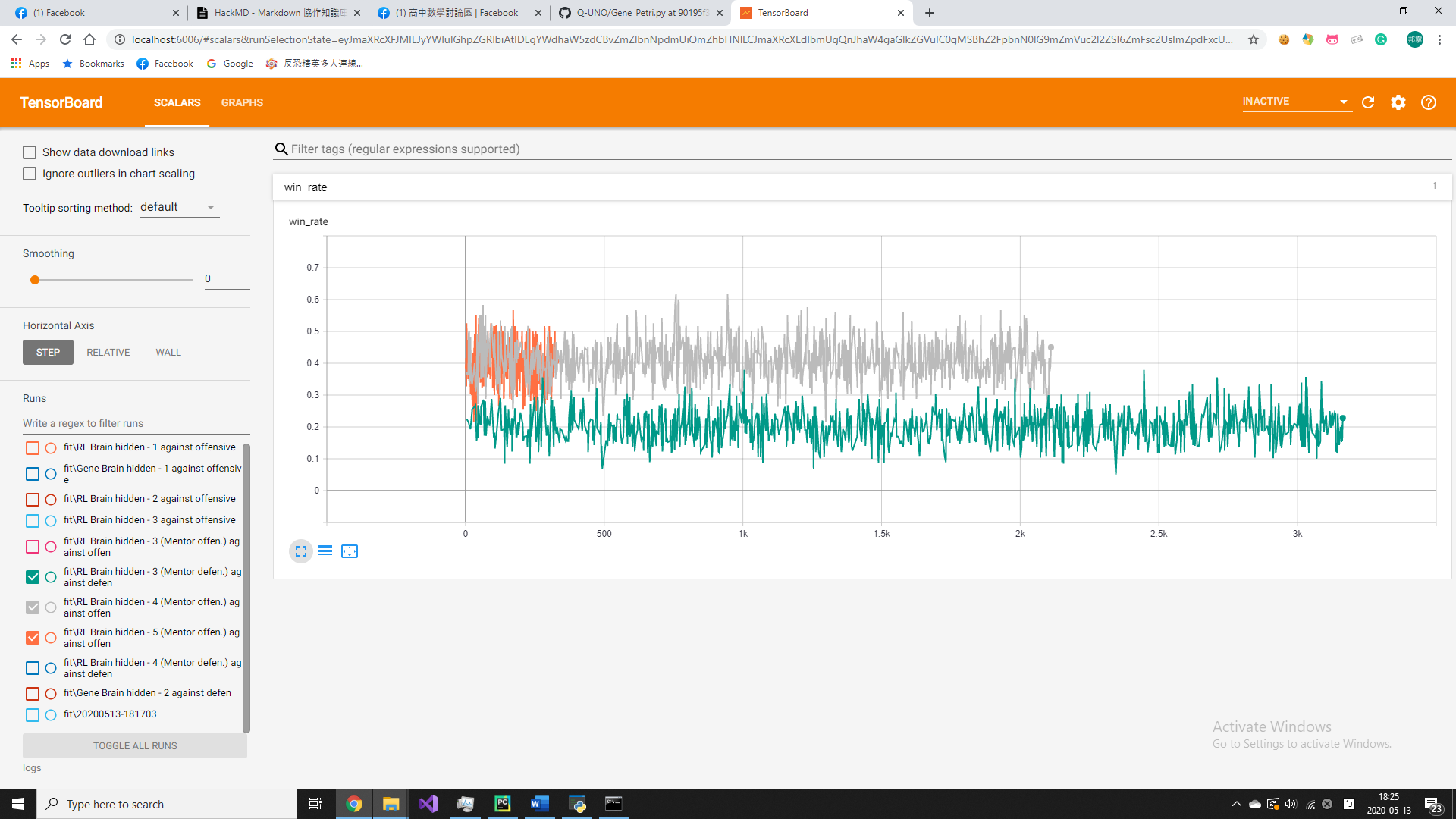
|  |  |
| --- | --- |
| 顏色 | 意義 |
| 藍 | Genetic algorithm with |
| 紅 | Deep-Q-Learning with 4-hidden layer and offensive algo as mentor |
| 橘 | Deep-Q-Learning with 3-hidden layer and offensive algo as mentor |

相反的， Deep-Q-Learning對隱藏層較為敏感，隱藏層深度越深，反而會造成反效果，如下圖所示。

|  |  |
| --- | --- |
| 顏色 | 意義 |
| 橘 | Deep-Q-Learning with 1-hidden layer |
| 紅 | Deep-Q-Learning with 2-hidden layer |
| 藍 | Deep-Q-Learning with 3-hidden layer |

深層的神經網路如塊璞玉，未經雕琢不得其瑰，環境提供的樣本品質不夠好，使得神經網路無法好好的被「雕琢」，若是有一個師傅能夠提供高品質的樣本，去「雕琢」這塊璞玉，便能得到內部的瑰麗，便能得到良好的勝率。

Behavior cloning猶如師徒相授，師傅做什麼，徒弟便跟著做；過於淺層的神經網路如朽木，再怎麼教也教不會，神經網路擬和能力過差；過於深層的神經網路如大材小用，耗費大量運算資源，卻無優異成效。

筆者認為四層的神經網路搭配Behavior cloning，不僅能有效的增加勝率，也能妥善使用運算資源，最為恰當，因此，所有相關實驗皆以此法為準。

|  |  |
| --- | --- |
| 顏色 | 意義 |
| 橘 | Deep-Q-Learning with 5-hidden layer and offensive algo as mentor |
| 灰 | Deep-Q-Learning with 4-hidden layer and offensive algo as mentor |
| 綠 | Deep-Q-Learning with 3-hidden layer and offensive algo as mentor |

# 五、各演算法之間的勝率

下表展示了各演算法之間的勝率，由於Genetic-Network以及Deep-Q-Learning不是傳統演算法，數據以最高勝率以及平均勝率顯示。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 主角  對手 | Dummy | Naïve  Offensive | Naïve  Defensive | Gene-Network | Deep-Q-Learning |
| Dummy |  |  |  |  |  |
| Naïve  Offensive |  |  |  |  |  |
|  |  |  |  |  |
| Naïve  Defensive |  |  |  |  |  |
|  |  |  |  |  |
| Gene |  |  |  |  |  |
|  |  |  |  |  |

可以觀察到幾個重點

一、Defensive algorithm勝率比Offensive algorithm高

二、Deep-Q-Learning多數時候比Genetic algorithm好

三、Dummy algorithm幾乎穩輸

四、不論是DQN或是Gene algorithm都能與Naïve algorithm相抗衡

五、

柒、結論

# 一、探討先手與後手的利與弊

在特定條件下，先手能夠「完全制霸」對手，也就是存在一種出牌順序，使得先手必勝，如下圖所示。

先手只需要出「停止」、「迴轉」再出「紅5」，先手就贏了，若存在先手必勝策略，後手將無機會逆轉，造成先手勝率較高。

# 二、探討各演算法之間的勝率

Naïve offensive algorithm對上Naïve defensive algorithm時，defensive有較高的機率勝出，這是因為offensive algorithm常常把功能牌先丟完了，最後沒有功能牌可以丟。根據筆者的自身經歷，多數人喜歡把功能牌留到最後一刻再丟，通常大家會傾向於「存功能牌」，並在遊戲結束前盡速丟出，就像Defensive algorithm一樣。

由於Deep-Q-Learning與Genetic-Network是「活」的，因此他們的勝率都會浮動(fluctuation)，最佳狀態下，他們能夠佔Naïve algorithm的小便宜，因為Naïve algorithm是「死」的。不對稱賽局 傳統演算法不會被掌握行蹤 導致這兩個演算法無法完全輾壓對手。

Dummy algorithm遇到所有演算法幾乎都被「輾壓」

# 三、探討Deep-Q-Network與Genetic-Network的利與弊

捌、參考資料