利用 RL 找出遊玩 21 點策略

108034555 蔡沛洹

Project 3

摘要

此 project 欲利用強化學習訓練 AI 遊玩 21 點(撲克牌遊戲)，並透過調整參數和採用遞減探索率的方式，以得到更好的訓練結果，並與隨機策略和基本策略對比結果表現。

一、背景介紹

新的一年即將到來，親朋好友們圍爐共度新春，是華人的傳統，而飯後不免俗的也會來點飯後娛樂，或是切磋牌技，小賭怡情。受此發想，此 project 希望使用強化學習，訓練 agent 學會玩 21 點(撲克牌遊戲)，人人發大財！

二、問題定義

採用常見的 5W1H 方法來定義問題：

What：要解決什麼問題？

解決打牌常常輸錢的問題

When：什麼時候進行？

準備大賺一筆前，比方說過年前就是訓練的好時機

Who：由誰來進行？

打牌的新手或容易輸的玩家

Where：在何處進行？

在編譯器中

Why：為什麼要做這件事？

若訓練的好，可找到一最佳玩法，提升勝率

How：如何進行？

利用強化學習方法中的 Q learning，搭配參數的調整

2.1.玩法規則(在此 project 中)

21 點是一個撲克牌遊戲，目標為讓牌的的總和在沒爆掉(bust)的情況下，離 21 點愈近愈好。牌面分成 13 種，牌面為 2 到 10 的牌，點數如牌面計算，而 J、Q、K 計為 10 點，A 比較特別，可以算作 1 點為 11 點。遊戲開始於玩家和莊家各取得兩張牌，其中莊家一張為明牌，即玩家也能看見，另一張則為暗牌，牌是蓋著的而玩家無法看見。

玩家在牌面爆掉前，可以要求再來一張牌，直到決定停止為止。若牌面爆掉，直

接判定莊家獲勝，而若玩家未爆掉且停止要牌，則輪到莊家的回合。莊家將暗牌揭開後，必須要牌直到牌面總和大於等於 17 點，若莊家爆牌，則玩家獲勝，若莊家沒爆牌，則點數較高者獲勝，點數相同則平手。勝利的報酬為+1，平手為 0，落敗為-1。

三、研究方法(介紹強化學習和程式)(參數調整)

強化學習簡單來說，便是訓練一個 AI 透過觀察環境，選擇最適合的決策，找出達到目標的方法，示意圖如下。圖中的 agent 即為欲訓練之 AI，而 environment 代表問題的環境，首先，agent 先取得 environment 目前的狀態(state)，再根據 state，選擇當下最好的決策(action)，此 action 將導致 environment 更新成新的狀態，同時 agent 也會得知此 action 所得到之 reward 值，經由判斷 state 決定 action，最終得到 reward 的流程，即是 agent 所學習的方法。

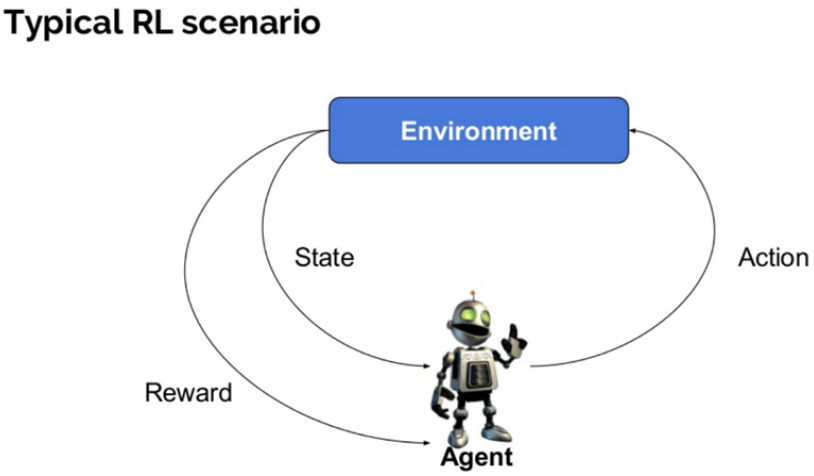


圖 1 強化學習概念圖

已知強化學習中有 state、action、reward 這三個重要部分，以下分別介紹

State

指的是環境的狀態，以 21 點來說則包括玩家的點數總和、莊家的明牌及有沒有 usable A(在此 project，A 為 1 點稱為 not usable，A 為 11 點稱為 usable)。舉例來說，State 可能為(14, 5, false)，代表玩家點數加總為 14，莊家明牌為 5 點，且玩家沒有作為 11 點的 A。

Action

指可執行的決策，以本 project 假設下較單純，玩家只有要牌和不要牌兩種決策，後面分別以 0 和 1 表示。

Reward

指得到的獎勵，以本 project 來說，玩家獲勝 reward 為 +1，平手為 0，而落敗 reward 為-1

本練習中的強化學習是透過 Q learning 來進行，Q table 中記錄各個 state 和 action 之 Q 值，一開始 Q table 為空的，每格 Q 值皆為零，此 Q 值會在每次做完決策後更新，最終便能得到一個最佳化後的 Q table，也就是 21 點的遊玩策略。

Q 值更新公式：

$$Q(s_t,a_t) \leftarrow \underbrace{Q(s_t,a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1},a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t,a_t)}_{\text{old value}} \right)$$

圖 2 Q 值更新公式

公式中包括 α 和 γ 這兩個參數。α 是學習率，α 愈高代表比較相信當前這步的 reward，愈低則代表比較相信舊的 Q 值。γ 是衰退率，愈高則代表考慮到許多步後，愈低則代表只考慮下一步。

隨機探索：

為了確保 agent 對環境有足夠的了解，必須先充足探索環境，ε 便在強化學習中代表探索環境的機率，ε 愈高代表愈高機率探索環境，也就是挑選 action 時不依照 Q 值最高的 action 行動而是隨機選取。在此 project 中，採用遞減的 ε，讓 agent 一開始可以充足探索環境，而隨著決策的回合數增加，agent 探索環境愈來愈充足，探索率也逐漸下降，確保 agent 能夠開始增強學習，最後收斂為最佳的 Q table 值。

```
(19, 10, True): {0: 0.5, 1: -0.041499999999999995},
(15, 10, False): {0: -1.0550000000000002, 1: -0.6816559075975512},
(20, 8, False): {0: 0.8528, 1: -0.5625},
(22, 8, False): {0: 0.0, 1: 0.0},
(18, 4, False): {0: 0.488, 1: -0.75},
(22, 4, False): {0: 0.0, 1: 0.0},
(14, 3, False): {0: 0.62, 1: -0.75},
(23, 3, False): {0: 0.0, 1: 0.0},
(13, 7, True): {0: -0.020000000000000018, 1: 0.0},
(16, 6, False): {0: -0.875, 1: -0.019999999999999997},
(20, 6, False): {0: 1.088, 1: 0.0},
(13, 10, False): {0: -0.889779763125, 1: -0.34343983928253813},
(17, 10, False): {0: -0.28174643712, 1: -0.91625},
(27, 10, False): {0: 0.0, 1: 0.0},
```

圖 3 Q table 樣貌

參數調整：

參數調整主要是調整α和γ這兩個參數，各選 3 個水準來模擬，最終得到α為 0.5，γ為 0.2 表現最佳。

四、研究結果

為了檢驗此 project 訓練出的 Q table 表現如何，找兩組對照組一同比較。分別為隨機策略和基本策略。

隨機策略：

每次選擇策略時，皆隨機挑選 action，即 50%的機率選擇要牌，50%的機率選擇不要牌。

基本策略：

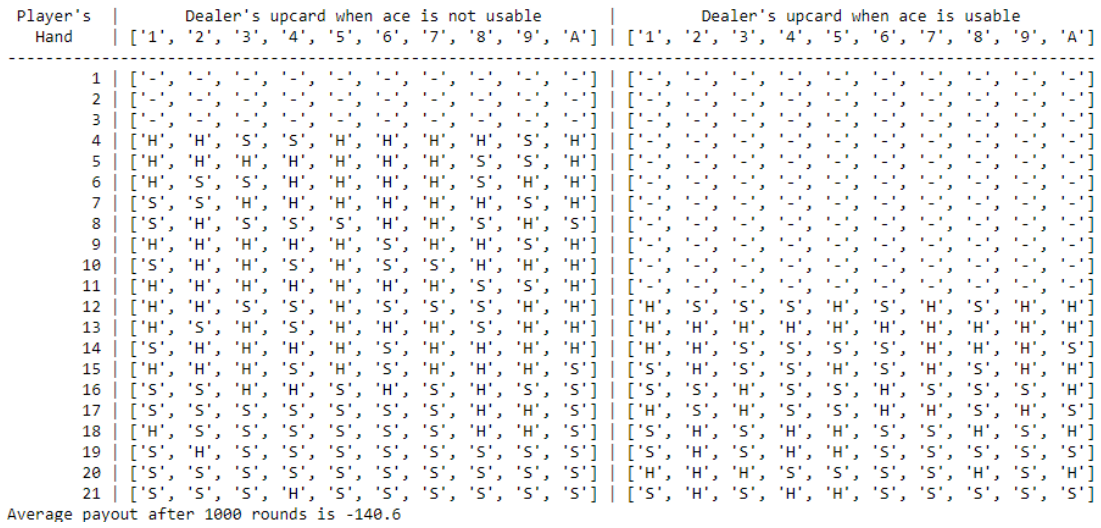
一般在玩 21 點時，其實是可以查表的，此表是透過數學和統計基礎得來，表上會列出手牌如何時，應該選擇要牌或是選擇不要牌，如圖。此表格為基本策略之一小部分，指示手牌總數多少且莊家明牌多少時，應作何決策，而 H(Hit)代表要牌，S(Stand)代表不要牌，舉例來說，手牌點數 12 而莊家明牌為 2 時，查表得到 H，即代表應該要牌的意思。

Player's Hand	Dealer's Upcard									
	2	3	4	5	6	7	8	9	10	A
12	H	H	S	S	S	H	H	H	H	H
13	S	S	S	S	S	H	H	H	H	H
14	S	S	S	S	S	H	H	H	H	H
15	S	S	S	S	S	H	H	H	H	H
16	S	S	S	S	S	H	H	H	H	H
A2	H	H	H	D	D	H	H	H	H	H
A3	H	H	H	D	D	H	H	H	H	H
A4	H	H	D	D	D	H	H	H	H	H

圖 4 基本策略表

強化學習訓練策略：

透過訓練，最終可得到一最佳化 Q table，再將其轉化為決策表，如圖。



Player's Hand	Dealer's upcard when ace is not usable										Dealer's upcard when ace is usable									
	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'	'A'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'	'A'
1	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
2	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
3	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
4	'H'	'H'	'S'	'S'	'H'	'H'	'H'	'H'	'S'	'H'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
5	'H'	'H'	'H'	'H'	'H'	'H'	'H'	'S'	'S'	'H'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
6	'H'	'S'	'S'	'H'	'H'	'H'	'H'	'S'	'H'	'H'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
7	'S'	'S'	'H'	'H'	'H'	'H'	'H'	'H'	'S'	'H'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
8	'S'	'H'	'S'	'S'	'S'	'H'	'H'	'S'	'H'	'S'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
9	'H'	'H'	'H'	'H'	'H'	'S'	'H'	'H'	'S'	'H'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
10	'S'	'H'	'H'	'S'	'H'	'S'	'S'	'H'	'H'	'H'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
11	'H'	'H'	'H'	'H'	'H'	'H'	'H'	'S'	'S'	'H'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'	'-'
12	'H'	'H'	'S'	'S'	'H'	'S'	'S'	'S'	'H'	'H'	'H'	'S'	'S'	'S'	'H'	'S'	'H'	'S'	'H'	'H'
13	'H'	'S'	'H'	'S'	'H'	'H'	'H'	'S'	'H'	'H'	'H'	'H'	'H'	'H'	'H'	'H'	'H'	'H'	'H'	'H'
14	'S'	'H'	'H'	'H'	'H'	'S'	'H'	'H'	'H'	'H'	'H'	'H'	'S'	'S'	'S'	'S'	'H'	'H'	'H'	'S'
15	'H'	'H'	'H'	'S'	'H'	'S'	'H'	'H'	'H'	'S'	'S'	'H'	'S'	'S'	'H'	'S'	'H'	'S'	'H'	'H'
16	'S'	'S'	'H'	'H'	'S'	'H'	'S'	'H'	'S'	'S'	'S'	'S'	'H'	'S'	'S'	'H'	'S'	'S'	'S'	'H'
17	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'H'	'H'	'S'	'H'	'S'	'H'	'S'	'S'	'H'	'H'	'S'	'H'	'S'
18	'H'	'S'	'S'	'S'	'S'	'S'	'S'	'H'	'H'	'S'	'S'	'H'	'S'	'H'	'H'	'S'	'S'	'H'	'S'	'H'
19	'S'	'H'	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'H'	'S'	'H'	'H'	'S'	'S'	'S'	'S'	'S'
20	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'H'	'H'	'H'	'S'	'S'	'S'	'S'	'H'	'S'	'H'
21	'S'	'S'	'S'	'H'	'S'	'S'	'S'	'S'	'S'	'S'	'S'	'H'	'S'	'H'	'H'	'S'	'S'	'S'	'S'	'S'

Average payout after 1000 rounds is -140.6

圖 5 強化學習策略表

分別對這三組策略進行 1000 次的牌局，結果如下表所示。觀察發現強化學習策略雖然已較隨機策略優異，但仍和基本策略有段落差，有改善的空間。

表 1 不同策略表現比較

	隨機策略	基本策略	強化學習策略
報酬	-400	-100	-140

五、結論及後續研究

雖然使用強化學習所得出的結果確實較完全隨機來得更好，卻仍比一般玩 21 點常使用的表格遜色，可能代表模型還有改善的空間。另外，此次問題假設較單純，玩家只有要牌和不要牌兩種決策，真實世界中其實還有 double 和 split 的玩法，若要後續研究，則應把這兩種玩法也加進去。若要嘗試更複雜的問題，則可以加入算牌功能，目前問題不考慮算牌，而現實玩法中通常使用 4 到 6 副牌來進行遊玩，哪些牌已經出了也要納入考量，才能選擇最佳的決策，同時也要注意，目前只有約 300 個 state，因此可以使用 Q learning 來進行，若 state 隨著問題複雜化，則應考慮使用深度強化學習，結合神經網路的概念來解決更大的問題。

六、參考資料

https://curiouscoder.space/blog/machine%20learning/teaching-a-computer-blackjack-using-reinforcement-learning/

https://www.semanticscholar.org/paper/The-evolution-of-blackjack-strategies-Kendall-Smith/9f910a5b3f03ff21a09f2df319cbc3705ee005ad
https://pathmind.com/wiki/deep-reinforcement-learning