Class 4　強化學習與模仿學習

課程地圖

沒有標準答案，怎麼學？

有人示範：模仿學習
只知道輸贏：強化學習

實作：坦克獎勵設計、模仿 + 強化、冰湖、21 點



今天的主角是「獎勵」與「示範」：期中的坦克，就站在這兩者的交會點。

今天的路線

課程地圖

① 強化學習

0–30 分

果蠅大腦開場；只知道輸贏，怎麼學？

② 怎麼學

30–70 分

Policy Gradient、Critic、Q 學習

③ 獎勵示範

80–110 分

獎勵設計、模仿學習、先模仿再強化

④ 練習

110–155 分

坦克、冰湖

⑤ 期中

155–180 分

專題現況、常見問題、截止前檢查

70–80 休息 ▶ 110–155 Colab（必做 A、B；冰湖、21 點） ▶ 155–180 期中專題時間

一條主線：沒有標準答案時，有示範就模仿、只知道輸贏就強化——兩個一起用最強。

開場時事：果蠅大腦爆紅

時事

- 2024：成蟲果蠅的腦完整連線圖，139,255 個神經元、5,450 萬個連結
- 2025：虛擬果蠅模仿真果蠅的影片，用強化學習學會走路和飛
- 2026/9：雄果蠅完整神經系統圖，166,700 個神經元
- 之後網路上到處是「把果蠅大腦放進 Minecraft」的影片



資料來源：FlyWire：Dorkenwald et al.（美國，普林斯頓大學），Nature, 2024；Vaxenburg et al.（美國，HHMI Janelia / Google DeepMind），Nature, 2025；Google Research（美國）、HHMI Janelia，Cell, 2026

一顆只有十幾萬個神經元的腦，能學會飛、找食物、躲危險——它是怎麼學的？

果蠅怎麼學：多巴胺就是獎勵訊號

時事

- 果蠅聞到某個氣味時喝到糖水 → 之後會主動飛向那個氣味
- 某個氣味時被電一下 → 之後會避開
- 腦中的多巴胺神經元，負責傳遞「獎勵」和「懲罰」
- 猴子實驗：多巴胺反映「比預期好多少」（第二段的 TD 誤差就是這個）



資料來源：Aso et al.（美國，HHMI Janelia），eLife, 2014；Schultz, Dayan & Montague（瑞士 / 美國），Science, 1997

動物的大腦裡，早就有一套「強化學習」：獎勵訊號 + 比預期好多少。

影片：果蠅大腦接到虛擬身體

時事

①強化學習 ②怎麼學 ③獎勵示範 ④練習 ⑤期中

網路爆紅：放進 Minecraft

YouTuber Ro0oney · 2026/9/7
「I Put A Fly's Conscious Brain into Minecraft」

▶ 點這裡開啟

網路影片：「第一個上傳的大腦」

Eon Systems 的虛擬果蠅（2026/3）
網路上流傳的介紹影片

▶ 點這裡開啟

作者自己的技術說明

Eon Systems · 2026/3/10
How the Eon Team Produced a Virtual Embodied Fly

▶ 點這裡開啟

先看前兩支影片：果蠅自己走路、理毛、吃東西。再看第三個連結：作者自己說了哪些是人接的線。

網路說法 vs 作者說明 vs 真正的研究

時事

①強化學習 ②怎麼學 ③獎勵示範 ④練習 ⑤期中

連線圖做到的是代表真果蠅真的做得到嗎?

動作	網路上的說法	作者的技術說明 / 研究證據
走路	「沒教牠走路，牠自己就會走」	走路控制器是另外用模仿學習訓練好的
輸入輸出	「整個大腦被上傳了」	要刺激哪些神經元、讀哪 5 ~ 6 個神經元，人手挑的
學習	「牠會自己學」	模擬時沒有任何學習：權重不會更新
連線圖的價值 ✓	—	Shiu 2024（Nature）：只靠連線圖 + 簡單模型，就預測出「嘗到糖 → 伸出口器」的神經元，再用真果蠅實驗驗證成功

資料來源：Eon Systems 技術說明，2026/3/10；Shiu et al.（美國，UC Berkeley），A Drosophila computational brain model reveals sensorimotor processing, Nature 634, 2024

真正的科學結果：連線圖能預測某些「由誰傳給誰」的反應；但會走、會學，目前還要靠人接線和另外訓練。

從期中的坦克出發：誰在給回饋？

銜接

- Class 1～3：每筆資料都有標準答案
- 期中：規則策略示範，網路模仿它的動作
- 今天：連示範都沒有，只有一個「評分者」
- 今天：有示範就模仿、只有輸贏就強化

學習方式	誰給回饋	回饋長什麼樣
監督式（Class 1～3）	標準答案	答錯多少
模仿學習（期中）	規則策略的示範	動作和示範差多少
強化學習	環境	獎勵(分數) ：贏 +10、被打 −1
GAN（下週）	另一個網路 （分類器）	像不像真的

從上往下，回饋越來越少、越來越間接。

第一段　強化學習：從獎勵中學

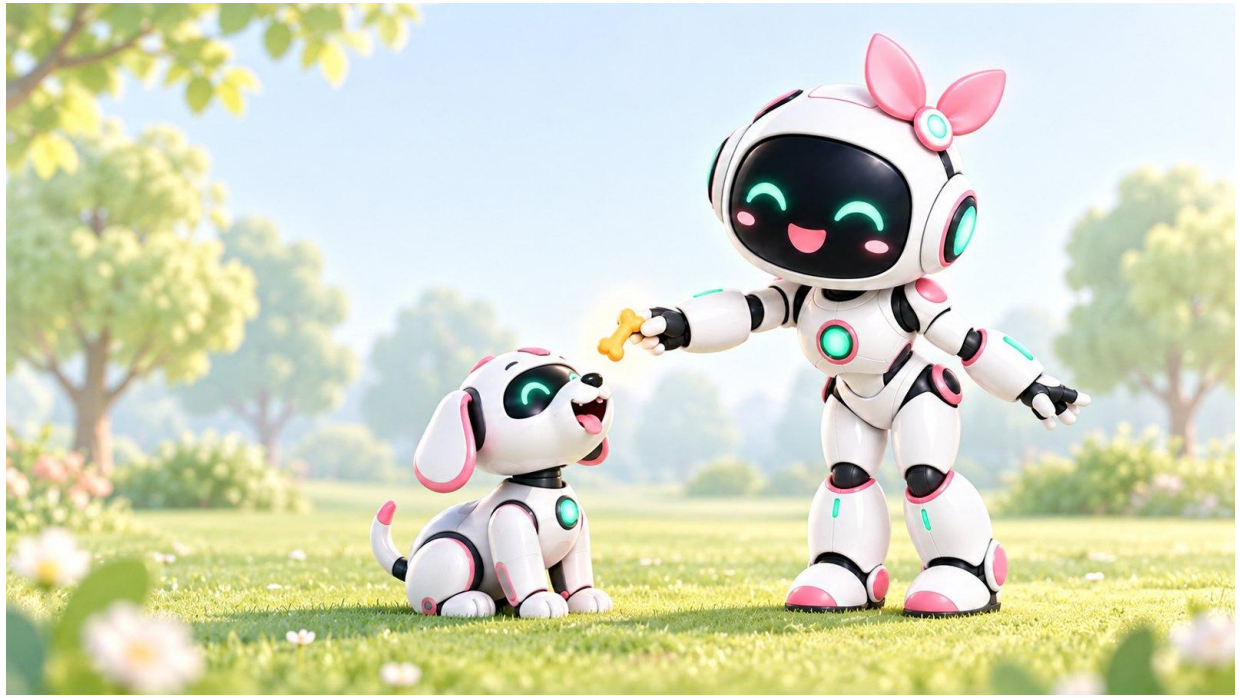
章節導覽

上一段

期中：照規則策略的動作學（模仿學習）。

這一段要回答

沒人示範、只知道最後贏或輸，AI 怎麼學？



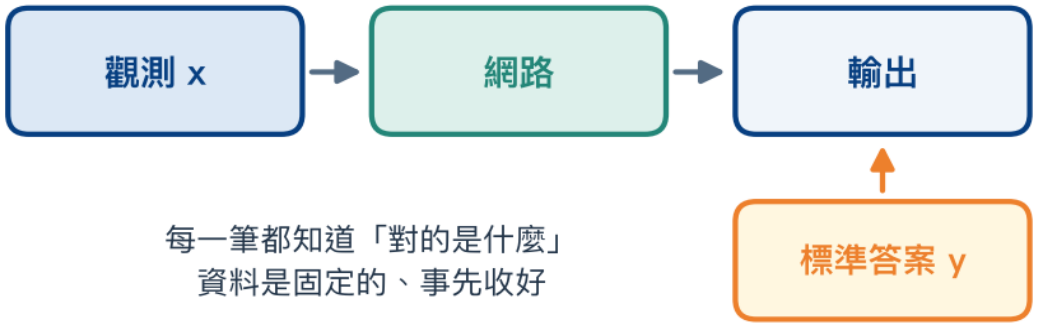
1. 互動迴圈：看、做、拿獎勵
2. 難題一：功勞算誰的
3. 難題二：探索還是利用

先用訓練狗狗來想：沒有人示範，只有零食。

為什麼不直接用監督式學習？

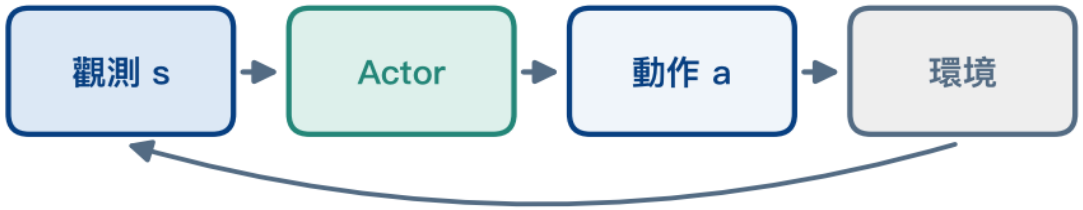
銜接

監督式學習：每筆資料都有標準答案



每一筆都知道「對的是什麼」
資料是固定的、事先收好

強化學習：只有分數，而且自己的動作決定下一筆資料



下一個觀測（由自己的動作決定）
＋獎勵：只是一個分數，常常很晚才給

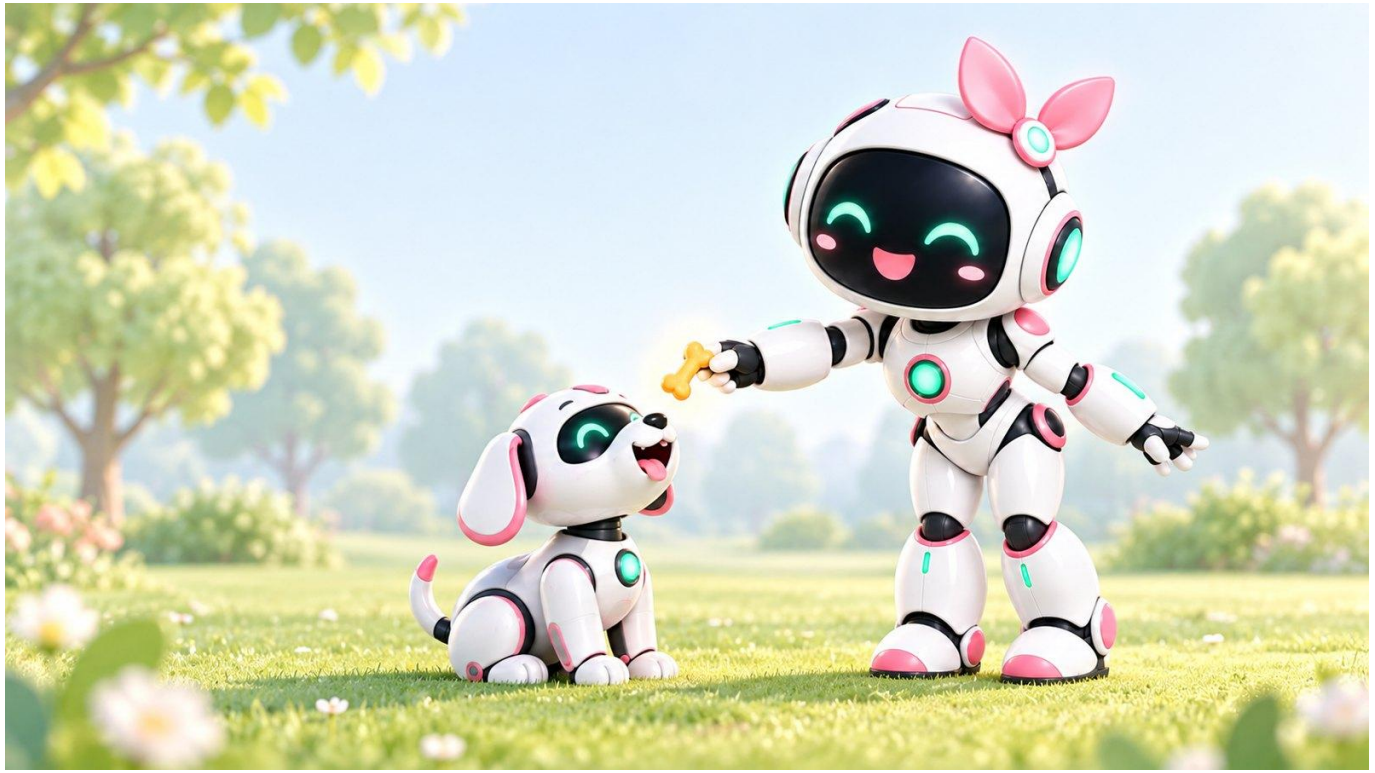
監督式要每一步都有標準答案；強化學習不用，透過打分數——而且自己的動作會決定之後看到什麼。

AI 版的訓練狗狗

類比

①強化學習 | ②怎麼學 | ③獎勵示範 | ④練習 | ⑤期中

- 沒有人示範「坐下」的動作
- 做對了給零食，做錯什麼都沒有
- 狗自己試出：哪個動作會有零食
- 這就是強化學習（Reinforcement Learning）



強化學習：不教動作，只給獎勵，讓它自己試出來。

用期中的坦克來對照

對照

- 上一頁的「看到、去做、拿獎勵」換成正式的名字
- 後面每一頁都會用到這些名詞
- Policy 就是我們要訓練的網路（也叫 Actor）

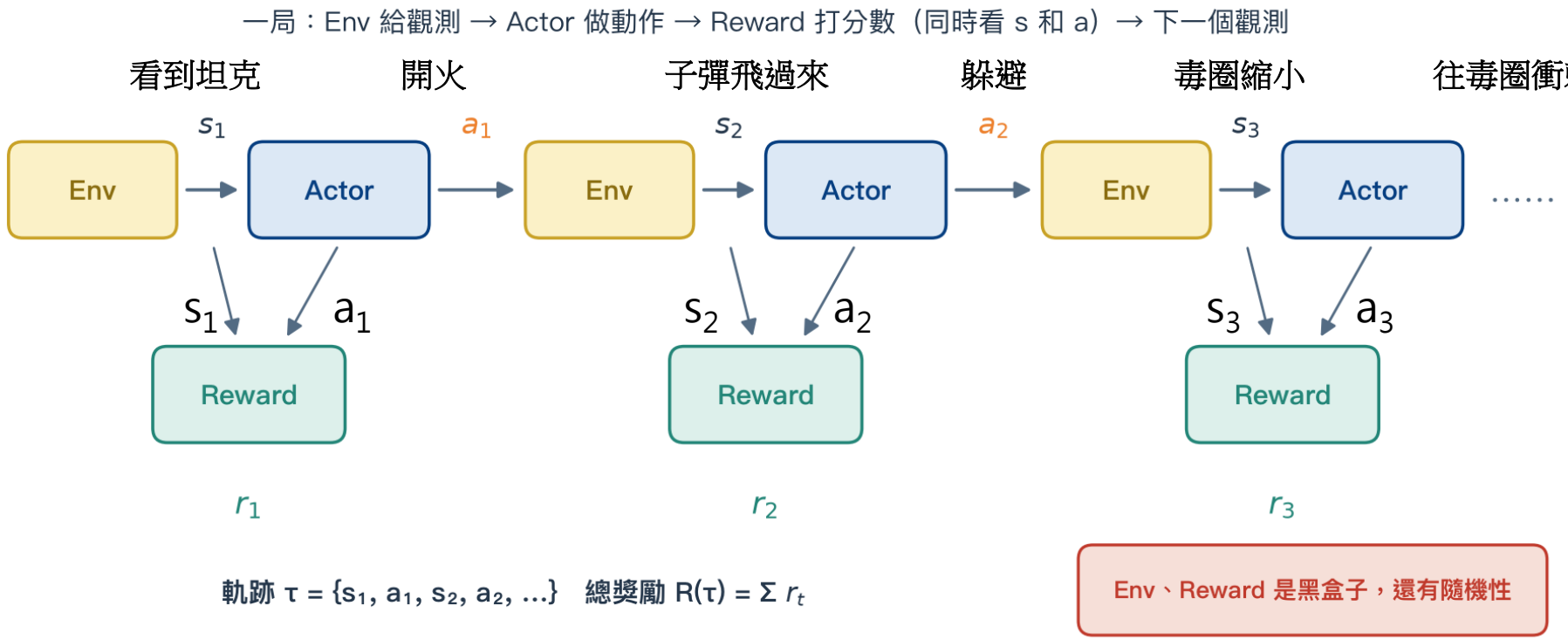
名詞	在坦克裡是什麼
Agent（代理人）	你的坦克
Environment（環境）	戰場、牆、其他坦克
State（狀態）-看到	91 個觀測數字
Action（動作）-去做	移動、車身、砲塔、開火、技能
Reward（獎勵） - 算獎勵	要你自己設計！
Policy（策略）	神經網路：看狀態選動作

名詞對到坦克：只有獎勵要你自己設計，其他都和期中一樣。

寫成數學：一局的軌跡與總獎勵

圖解  做的每一件事都有代價 (分數)

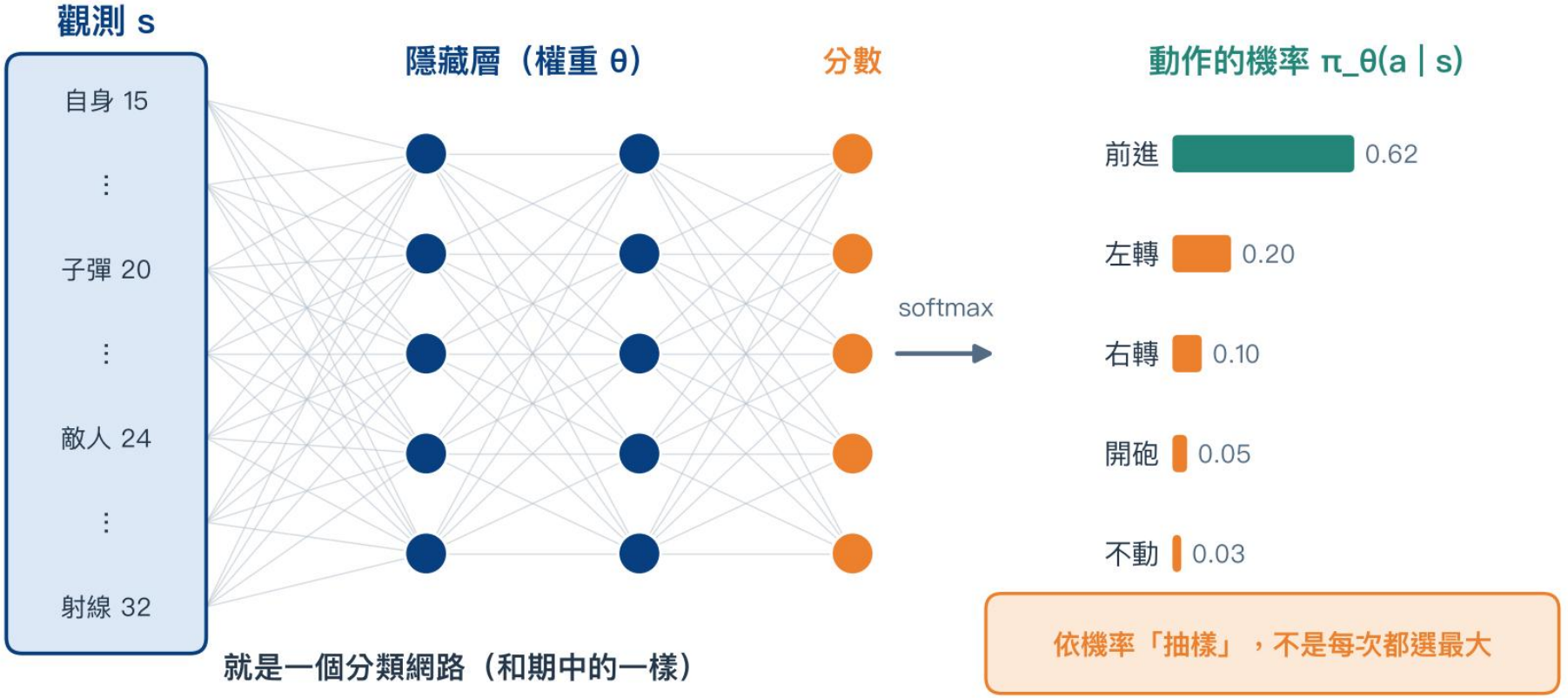
目標：讓一局的總獎勵 $R(\tau) = r_1 + r_2 + \cdots + r_T$ 平均起來越大越好



Actor 看狀態、做動作，環境回報獎勵和下一個狀態——一局就是一串 s、a、r。

圖解：Actor 就是一個分類網路

圖解  **簡單說就是訓練看到什麼應該怎麼做的分類器**



輸入 91 個觀測，經過隱藏層，最後用 softmax 變成 5 個動作的機率：和期中的網路完全相同。

強化學習也是機器學習的三個步驟

圖解 神經網路訓練~87%只要用一招

和 Class 1 一樣是三個步驟，只是第 3 步卡住了



前兩步和 Class 1 一樣；第 3 步卡住：環境和獎勵是黑盒子，梯度穿不過去。

黑盒子怎麼辦？用「試玩」繞過去

圖解  不管怎麼樣，先得到分數在說

想做的：一路微分（做不到）



梯度要穿過環境和獎勵 ✗
（不知道裡面的公式，還會隨機）

例：玩了兩局

第 1 局 贏 +10	做過的：前進、開砲	↑ 機率調高
第 2 局 輸 −5	做過的：左轉、不動	↓ 機率調低

「分數怎麼算、怎麼調」是第二段的 Policy Gradient

繞過去：只對 Actor 微分

① 讓 Actor 照機率去玩很多局
環境只要「會回分數」就好

② 記下每一步的 (觀測, 動作)
和這一局拿到的分數

③ 把做過的動作當「標籤」做分類
分數高 → 機率調高　分數低 → 調低

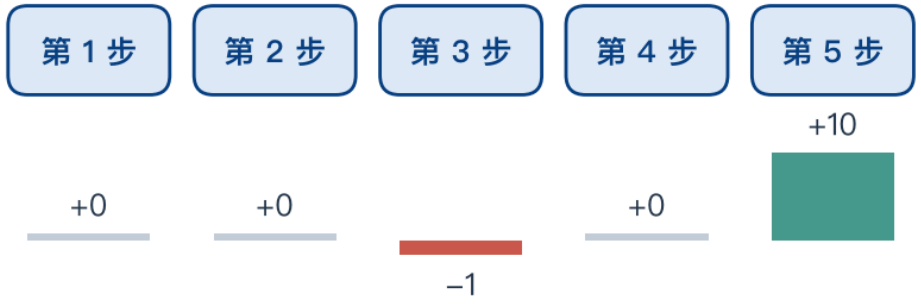
③ 就是 Class 1 的分類＋梯度下降：梯度只經過 Actor 網路

不對環境微分：讓 Actor 去玩，把分數高的局裡做過的動作當標籤，就變回分類問題。

如何幫每一步計算代價

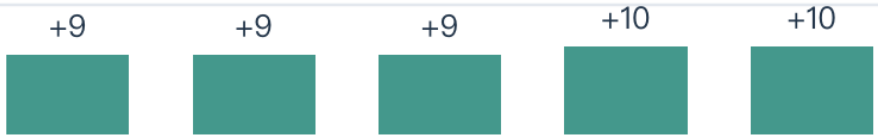
圖解

例：一局 5 步，第 3 步被扣 1 分（撞牆），第 5 步打贏 +10



① 只看當下的獎勵 r_t

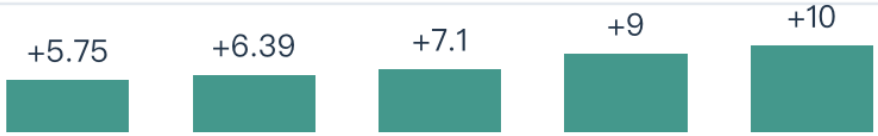
問題：前兩步在往勝利走，卻 0 分
第 3 步通往勝利，卻被扣分



② 之後全部加起來 G_t

功勞傳到前面了。問題：第 1 步和
第 4 步一樣多；900 步的遊戲，
第 1 步也算進 900 步後的事

全部加起來的問題：遊戲若沒終局，就爆炸了→ 都是無限大



③ 越遠的打越多折 G'_t（γ=0.9）

離結果越近，功勞越大
越遠的事，和這一步的關係越弱

0.71*0.9　9*0.9-1　分數打折傳遞可以推進往正確終局前進

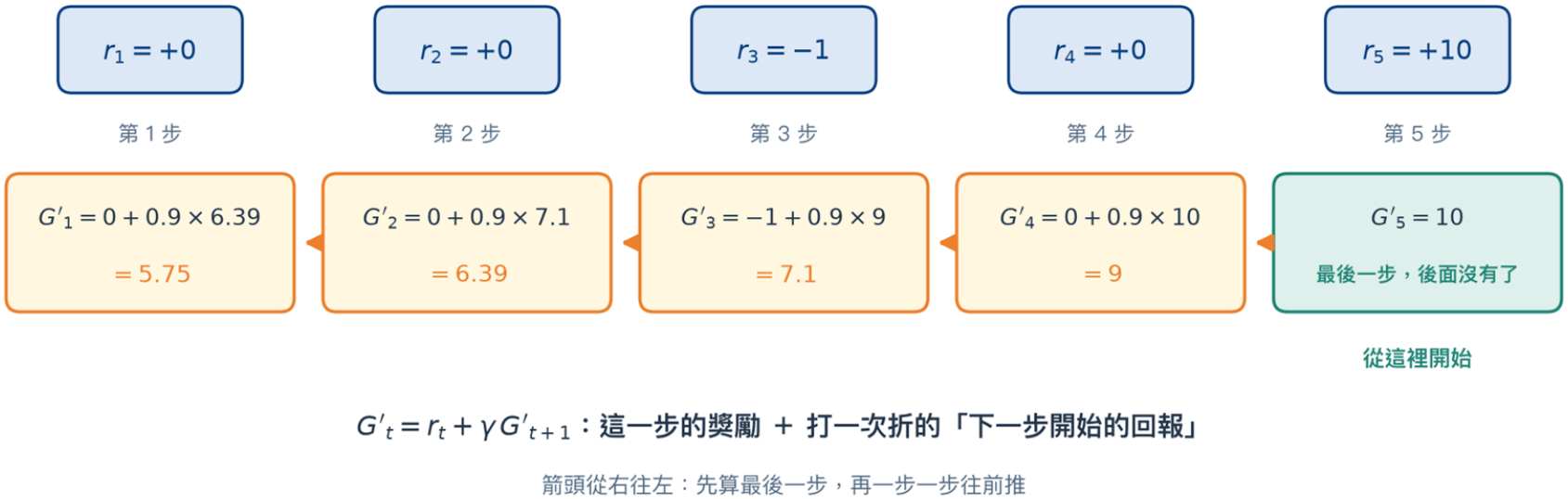
只看當下 → 功勞傳不到前面；全部加起來 → 遠近一樣多；所以越遠的打越多折。

推導：回報可以一步一步往回算

圖解

$$G_t = r_t + \gamma G_{t+1}$$

G'_t：從第 t 步開始，之後全部（越晚的打越多折，γ = 0.9）



$G'_t = r_t + \gamma G'_{t+1}$：這一步的獎勵 ＋ 打一次折的「下一步開始的回報」

箭頭從右往左：先算最後一步，再一步一步往前推

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$$

$$G_t = r_t + \gamma \underbrace{\left(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots \right)}_{\text{這就是 } G_{t+1}}$$

③ 的數字不用一項一項加：從最後一步往回算，每一步 = 自己的獎勵 + γ×下一步的回報。

想一想②

手算

前三步都沒有獎勵，第 4 步贏了 +10，γ = 0.9。第 1 步的回報 G 是多少？

A　10

B　7.29

C　2.5



用紙筆算，再到 Colab A1 對答案。

想一想②：解答

手算

前三步都沒有獎勵，第 4 步贏了 +10，γ = 0.9。第 1 步的回報 G 是多少？

A　10

B　7.29 ✓

C　2.5

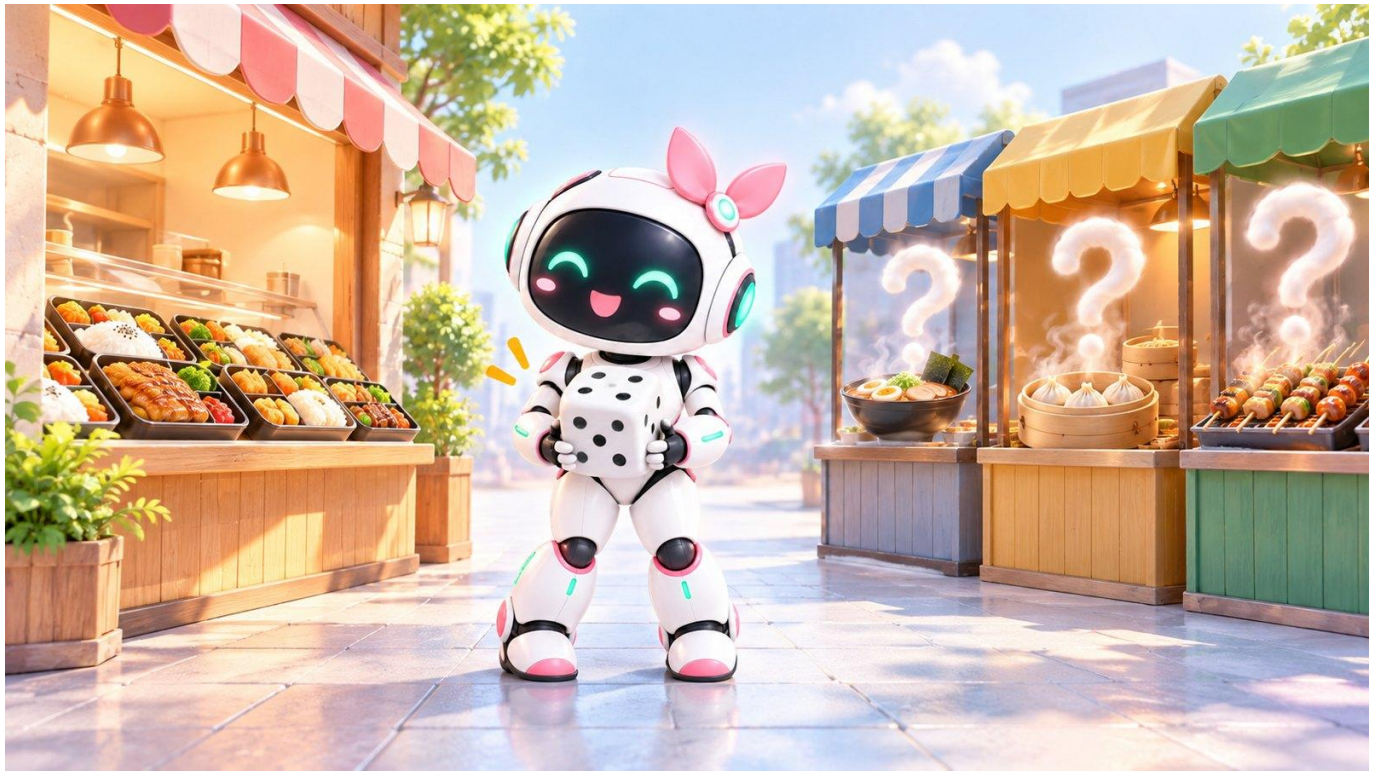
三步之後才拿到，所以打了三次折。



難題二：探索還是利用？

核心觀念

- 利用：去吃目前知道最好吃的那家
- 探索：試試新店，可能更好、也可能踩雷
- 只利用：永遠不知道有更好的
- 只探索：天天踩雷
- ε-greedy：擲骰子，ε 的機率亂試

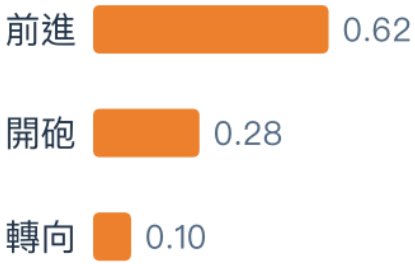


學習初期多探索，越學越多利用。

探索：收集資料一定要有隨機性

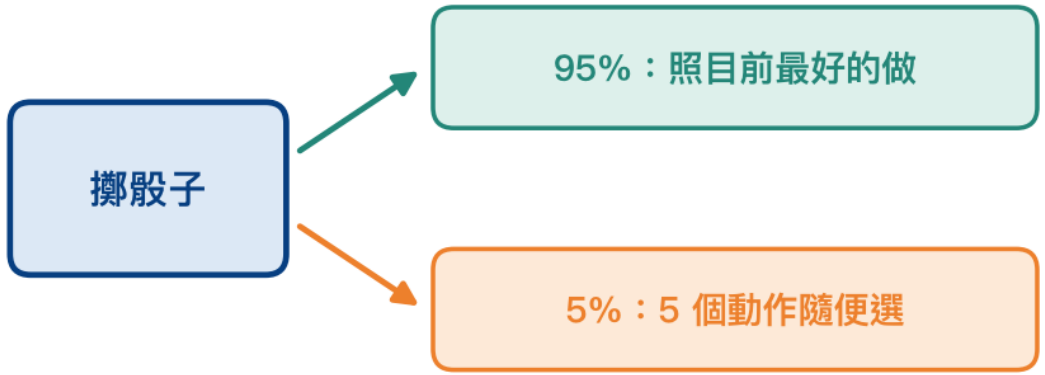
圖解  也就是ACTOR即便決定做這個動作，我還是亂數選動作

照機率抽動作：本來就會探索



0.10 的轉向偶爾也會被抽到
(不用另外加什麼)

每次都選「目前最好的」：要自己加隨機性



這叫 ε−greedy（ε = 亂選的機率，這裡 0.05）
第二段的 Q 學習就是這樣探索

收集資料時一定要有隨機性：從沒試過「開砲」，就永遠不知道開砲會怎樣

照機率抽動作本來就會探索；每次都選最好的，就要自己擲骰子（ε-greedy）。

第二段　Policy Gradient、Critic、Q 學習

章節導覽

上一段

互動迴圈、回報與打折、探索與利用。

這一段要回答

Actor 要怎麼訓練？A 該怎麼給？



1. Policy Gradient：A 的四個版本
2. Critic：MC、TD、Actor-Critic
3. Q 學習：只有 Critic 也能選動作

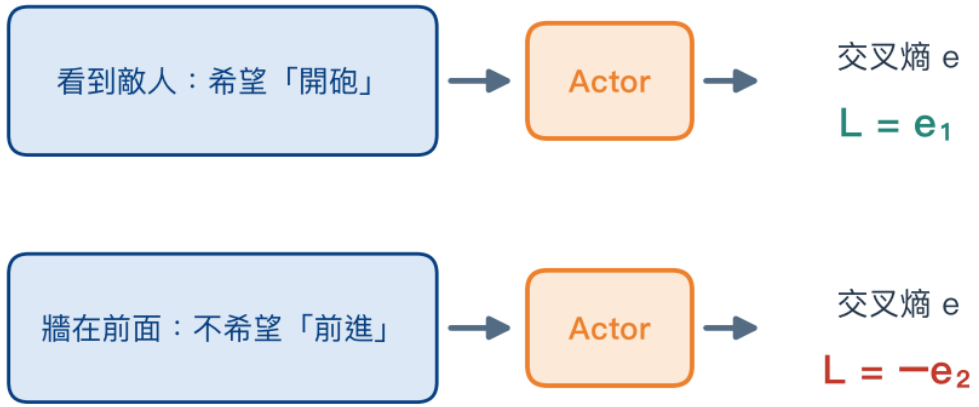
先把第一段的試玩寫成式子（Policy Gradient），再加一個會打分的 Critic，最後只用 Critic 選動作（Q 學習）。

Policy Gradient：用來控制你的ACTOR

圖解

想做 $\hat{a}$：$L = e$　　**不想做：$L = -e$**　　**有程度之分：$L = \Sigma_n A_n e_n$**

① 想做 / 不想做



合起來：L = e₁ − e₂，讓 L 越小越好

② 訓練資料：每筆 ±1

$\{s_1, \hat{a}_1\}$	+1
$\{s_2, \hat{a}_2\}$	−1
$\{s_3, \hat{a}_3\}$	+1
⋮	
$\{s_N, \hat{a}_N\}$	−1

L = +e₁ − e₂ + e₃ … − e_N

③ 換成程度：A_n

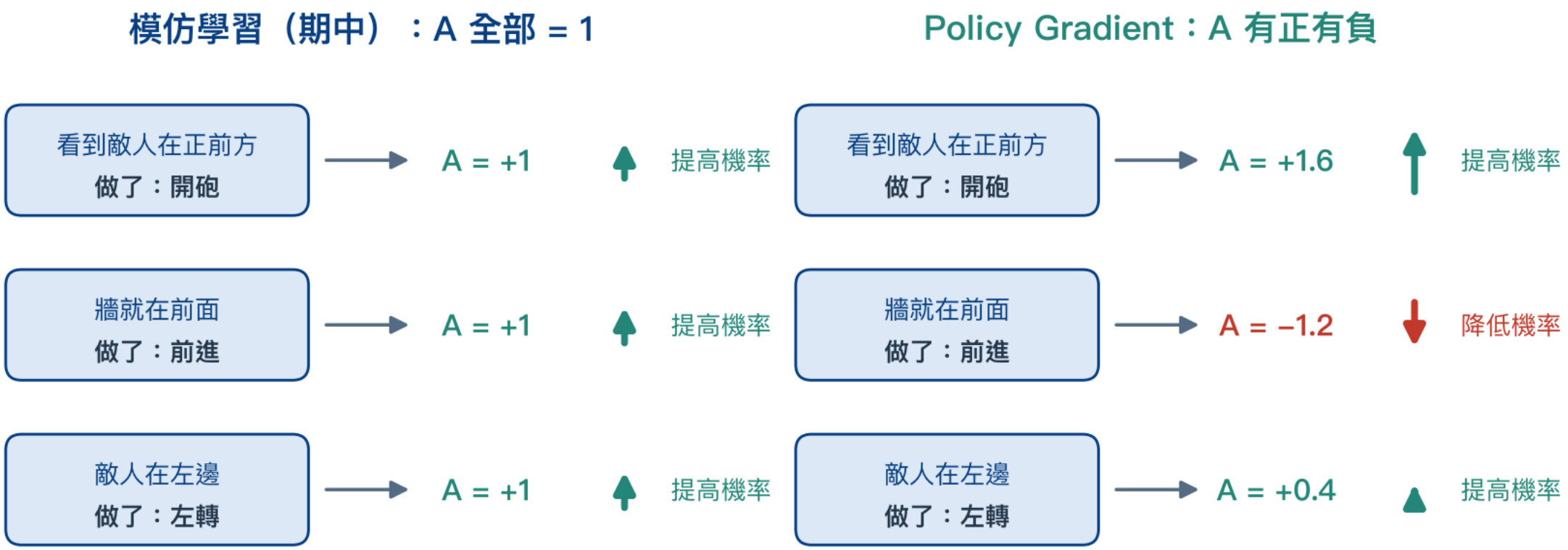
$\{s_1, \hat{a}_1\}$	+1.5
$\{s_2, \hat{a}_2\}$	−0.5
$\{s_3, \hat{a}_3\}$	+0.5
⋮	
$\{s_N, \hat{a}_N\}$	−10

$L = \Sigma\, A_n \cdot e_n$

Policy Gradient = 有權重的分類；期中的模仿學習就是 A 全部等於 1。

Policy Gradient → 如何讓 Actor 在s1執行a1

圖解 每一步照做→變成這步做得好不好



第一段的「把做過的動作當標籤」：每筆資料乘一個分數 A。模仿學習的 A 全部是 1。

A 該怎麼給？把第一段的分數接上來

對照

版本	A =	第一段的哪一種	還有什麼問題
Version 0	r_t	① 只看當下的獎勵	功勞傳不到前面
Version 1	G_t	② 之後全部加起來	遠近一樣多
Version 2	G'_t	③ 越遠打越多折	獎勵全是正的時候，做過的動作都被鼓勵
Version 3	$G'_t - b$	再減掉平均 b（下一頁）	要知道「平均」是多少
Version 3.5、4	用 Critic 估 b	學完 Critic 再講	—

Version 0 ~ 2 第一段算過了，這裡只多一件事：減掉平均（baseline）。

推導：為什麼 A 要減掉 baseline？

圖解  今天拉肚子沒參加到隨堂測驗→ 避免沒有翻身的機會

$$A = G'_t - b \quad (b：\text{平均回報})$$

① 這一輪發生了什麼

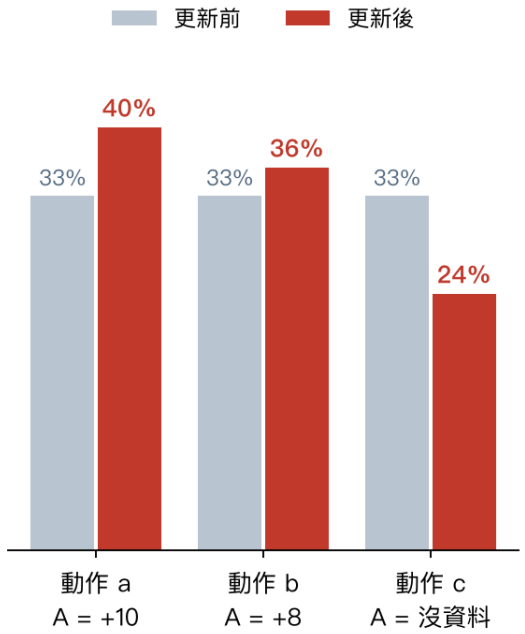
動作 a　抽到　得 +10 分

動作 b　抽到　得 +8 分

動作 c　沒抽到　（其實 +12）

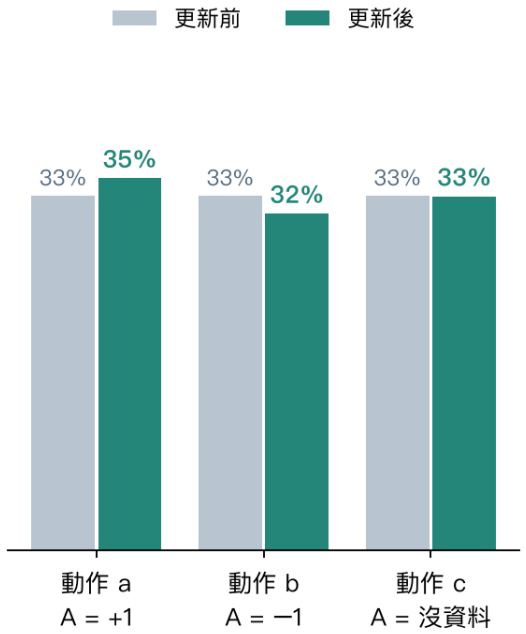
一開始三個動作的機率都是 1/3
獎勵全部是正的

② 沒有 baseline：A = 獎勵



a、b 都變高 → c 被擠下去
最好的 c 反而越來越難被試到

③ 有 baseline：A = 獎勵 − 平均 9

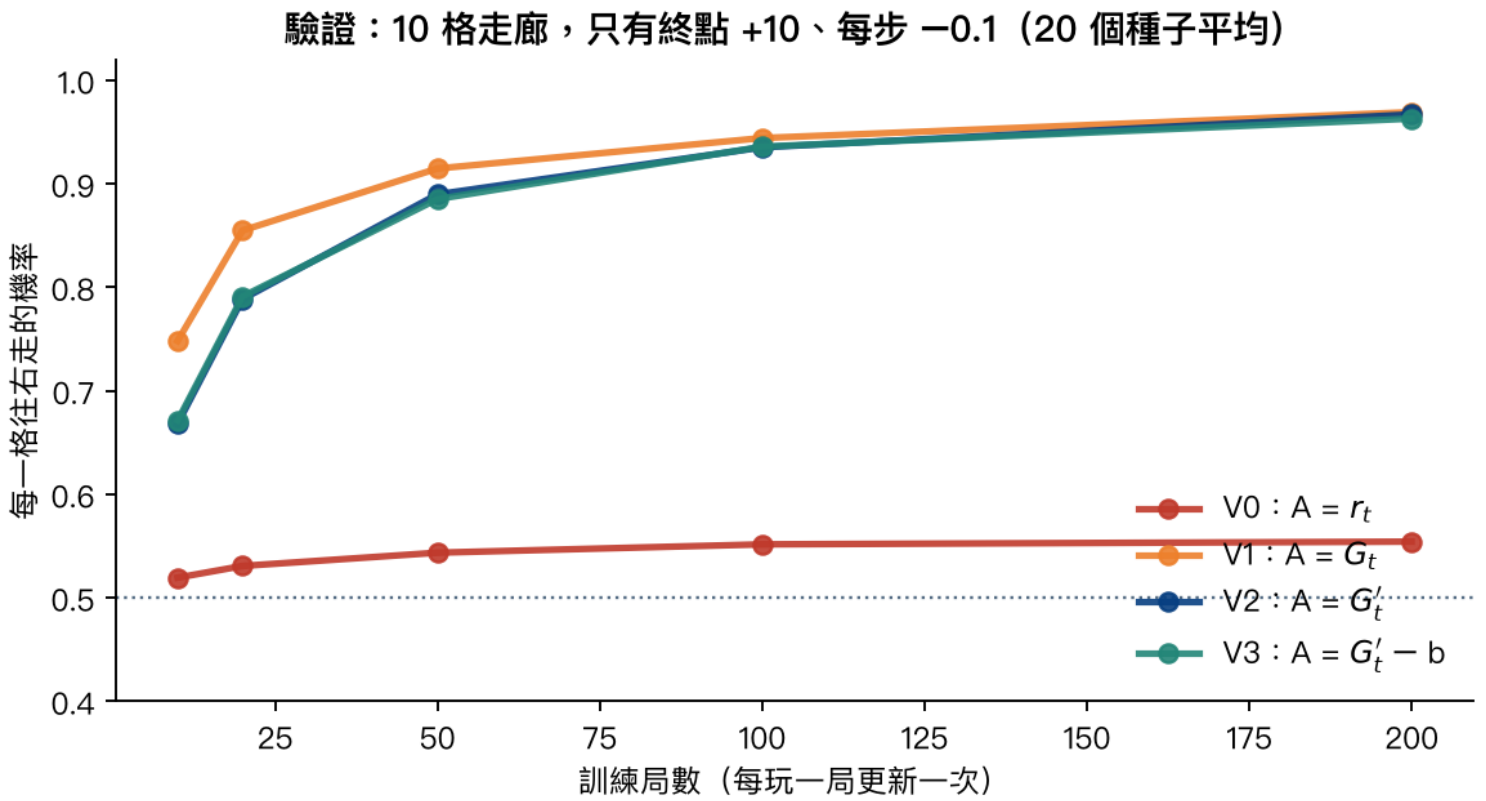


比平均好的 a 才上升、b 下降
c 維持原樣，還有機會被試到

同一筆資料：a 得 10、b 得 8。不減平均，兩個都被鼓勵、c 被擠下去；減掉平均 9，只有 a 被鼓勵。

驗證：不同版本學得多快？

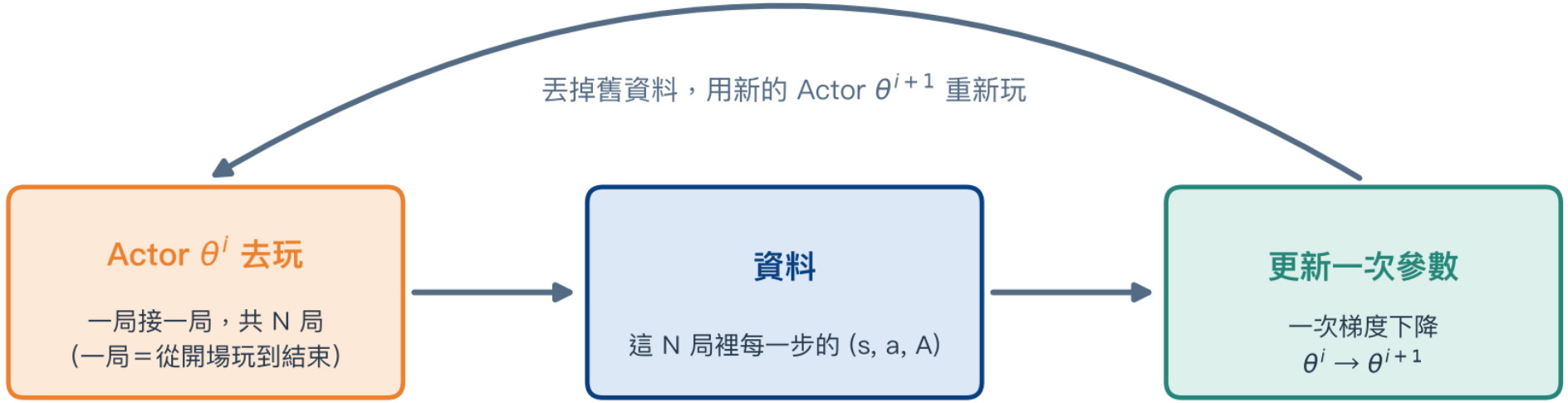
- 10 格走廊：只有終點 +10，每步 −0.1
- 每玩一局更新一次，20 個種子平均
- Version 0 學不會：前面每一步都只看到 −0.1
- Version 1 ~ 3 很快學會往右走
- baseline 的好處，要看獎勵全正的情況（上一頁）



只看當下的獎勵（Version 0）會完全學不會延遲的獎勵——功勞一定要往前傳。

圖解：Policy Gradient 的資料要邊玩邊收

圖解



這一圈叫「一輪」（iteration）＝ 玩 N 局 ＋ 更新一次；一直重複：$\theta^1 \rightarrow \theta^2 \rightarrow \theta^3 \cdots$

上一頁的走廊實驗：N = 1，每玩完一局就更新一次；200 局 ＝ 更新 200 次

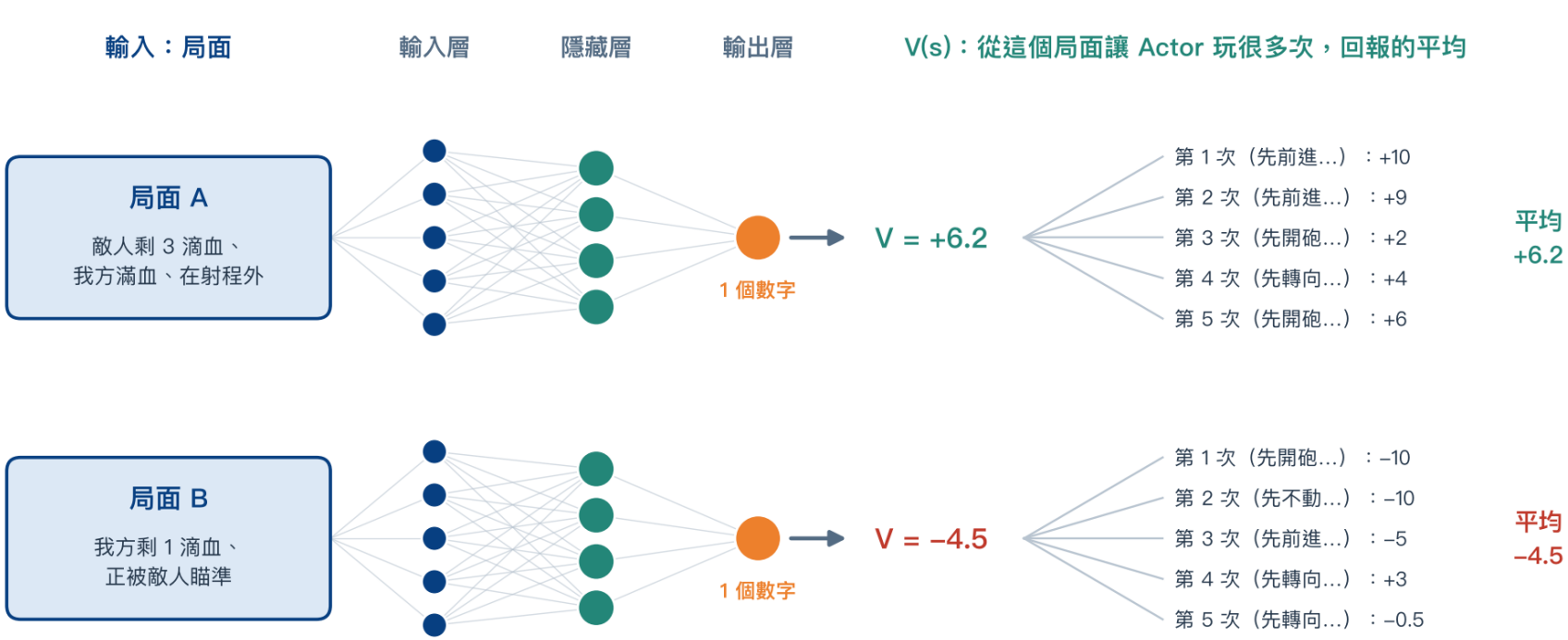
舊資料是舊 Actor 玩的，不代表新 Actor 會怎麼做：更新後就不能再用（on-policy），所以要玩很多局

一輪 = 玩 N 局（episode）+ 更新一次參數；更新後舊資料就丟掉，用新的 Actor 重玩。

Critic：這個局面值多少？

圖解  有點像玩21點發牌完你的牌面好壞

$V^{\theta}(s)$：看到 s 之後，照目前的 Actor θ 玩下去，平均還能拿多少（打折後的）回報



輸入層：91 個觀測（圖只畫 5 個）；隱藏層：和 Actor 一樣；輸出層只有 1 個神經元＝這個局面值幾分（Actor 是 5 個動作的機率）

每次 Actor 照機率抽動作，所以過程不同、分數不同；如果固定第一個動作再平均，就是後面的 Q(s, a)

Critic 也是網路，但只輸出 1 個數字：從這個局面玩下去，回報的平均。

Critic 怎麼學：MC 與 TD

圖解

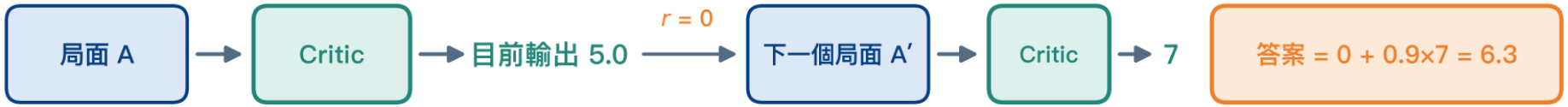
MC： $V(s_t) \approx G'_t$　　**TD：** $V(s_t) \approx r_t + \gamma V(s_{t+1})$

MC（Monte Carlo）：玩到結束，用真的回報當答案　　**要等一局結束**



5.0 比答案 9 小 → 往上調。　缺點：每次玩下去的結果差很多（像前一頁的 +10、+2…），答案很晃

TD（Temporal Difference）：只走一步，用「獎勵＋下一個局面的估計」當答案　　**走一步就能學**



5.0 比答案 6.3 小 → 往上調。　好處：不用等結束、答案比較穩；缺點：Critic 自己估的 7 不一定準

訓練 Critic 要有答案：MC 玩到結束用真的回報，TD 只走一步用「獎勵 + 下一個局面的估計」。

推導：TD 的式子從哪裡來？

推導

① 第一段推過：回報可以往回算

$$G_t = \underbrace{r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots}_{= \gamma G_{t+1}}$$

G_t ＝ 這一步 ＋ γ ×（從下一步開始的回報 G_{t+1}）

② 兩邊取平均：G 會隨機，V 是它的平均

$G_t = r_t + \gamma G_{t+1}$ → 取平均 → $V(s_t) \approx r_t + \gamma V(s_{t+1})$

右邊只要「走一步」就看得到：
r_t 是剛拿到的獎勵，$V(s_{t+1})$ 是 Critic 自己估的
→ 不用等整局結束

$V(s_t)$ 是 G_t 的平均、$V(s_{t+1})$ 是 G_{t+1} 的平均

③ 拿來訓練（γ = 0.9）

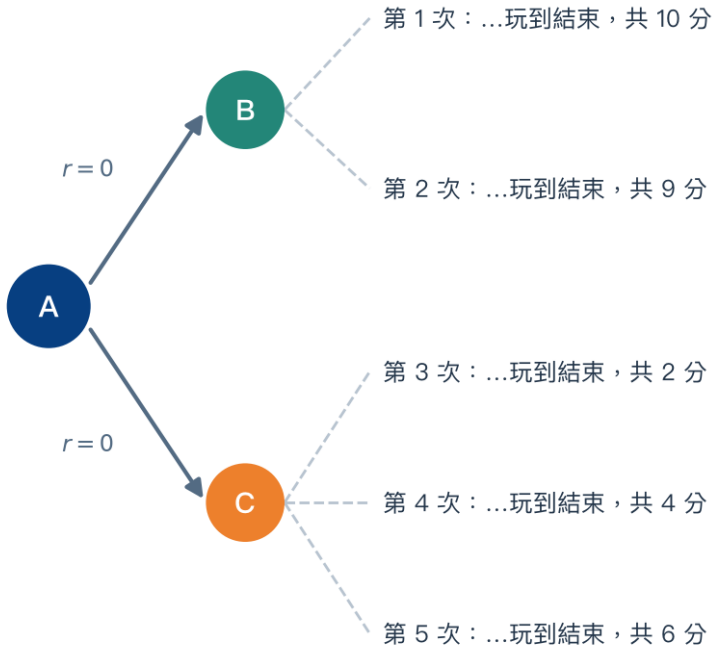
目前估 $V(s_t) = 6$ → 走一步：$r_t = 1$，估 $V(s_{t+1}) = 8$ → 目標 = 1 + 0.9×8 = 8.2 → δ = 8.2 − 6 = +2.2 → $V(s_t)$ 往上調

損失 = $(V(s_t) - [r_t + \gamma V(s_{t+1})])^2$：就是 Class 2 的 MSE，δ 叫 TD 誤差

回報可以拆成「這一步 + 打折的下一步回報」，兩邊取平均就是 TD；每走一步就能算一次誤差來訓練 Critic。

圖解：同一個例子，MC 和 TD 各算出多少？

這 5 次發生了什麼？



同樣走到 B，之後的運氣不同，所以一次 10、一次 9
玩到結束的分數是給 MC 用的；TD 只需要 A → B 這一步

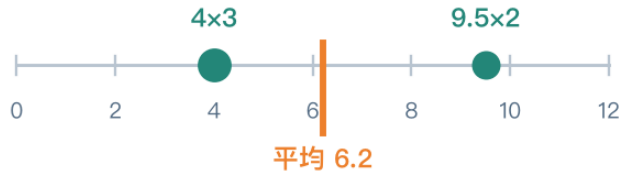
MC：答案＝玩到結束的回報 G'



答案：$G' = 10,\ 9,\ 2,\ 4,\ 6$

$V(A) \approx (10 + 9 + 2 + 4 + 6)/5 = 6.2$

TD：答案＝$r + \gamma V(s')$（Critic 已學好，$\gamma = 1$）



學好時 $V(B) = 9.5$、$V(C) = 4$（＝從 B、C 玩下去的平均）

答案：$0 + 1 \times 9.5$（第 1、2 次）、$0 + 1 \times 4$（第 3~5 次）

$V(A) \approx (9.5 \times 2 + 4 \times 3)/5 = 6.2$

MC、TD 都求出 V(A) ≈ 6.2；TD 的答案比較集中，但 Critic 還沒學好時，就偏到 5.2。

補充：用開車回家理解 MC 與 TD

類比  **修正Map 在第0秒行車預估時間**

開車回家：這裡的分數是「花掉的時間」（$\gamma = 1$，學習率 $\alpha = 0.5$），算法和獎勵一樣



① 出發

估計：共 30 分鐘

$V(s_t) = 30$

② 開了 5 分鐘

導航：還要 32 分鐘

$r_t = 5$，$V(s_{t+1}) = 32$

③ 到家

實際：共 40 分鐘

$G = 40$

MC：等到家才修正

什麼時候修：到家之後（③）

答案：實際總共花了 40 分鐘

新估計 = 30 + 0.5 × (40 − 30) = 35

真實結果，但要等整趟開完

TD：開 5 分鐘就修正

什麼時候修：開了 5 分鐘（②）

答案：$r_t + V(s_{t+1}) = 5 + 32 = 37$

差 δ = 37 − 30 = +7 → 新估計 = 30 + 0.5 × 7 = 33.5

不用等到家；但「還要 32 分鐘」也只是估計

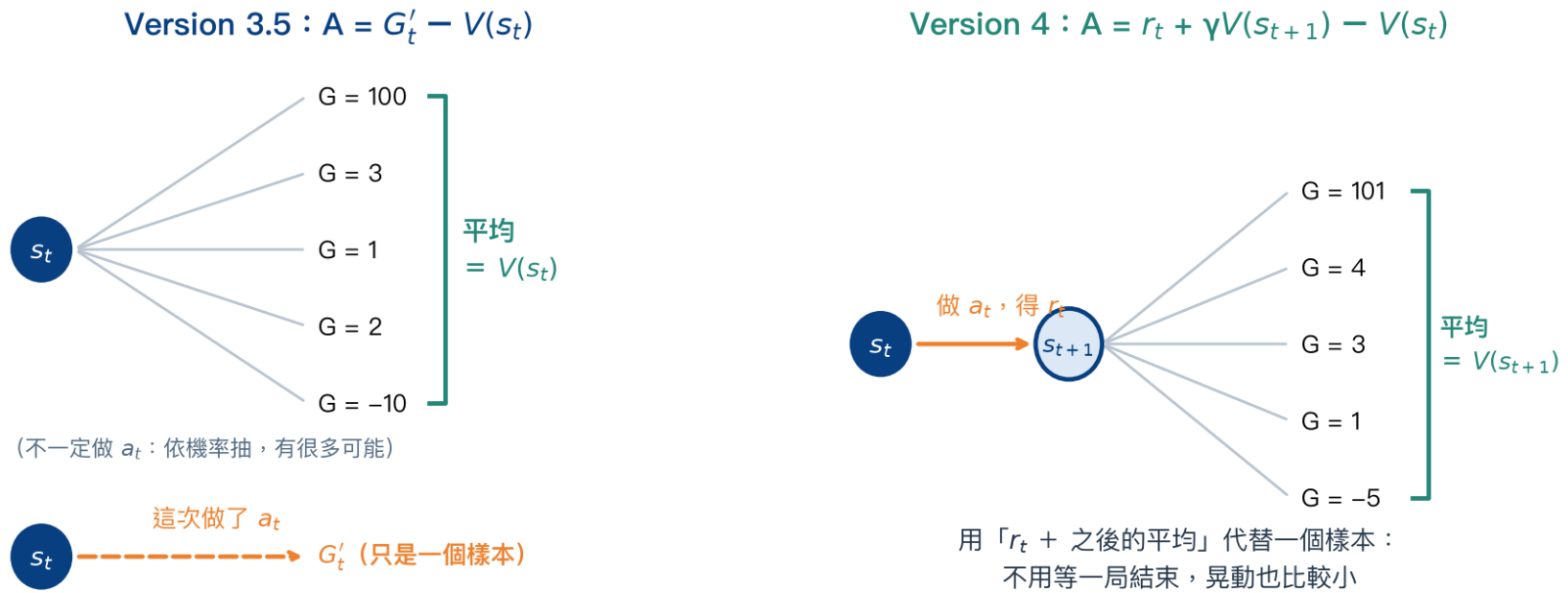
MC 要等到家才知道總共花多久；TD 開了 5 分鐘、看到塞車，就能先修正「從公司回家要多久」的估計。

圖解：Version 3.5 與 Version 4

簡單來說，得到獎勵目標要大於所有好壞結果的平均預期

Version 3.5： $A = G'_t - V(s_t)$

Version 4： $A = r_t + \gamma V(s_{t+1}) - V(s_t)$

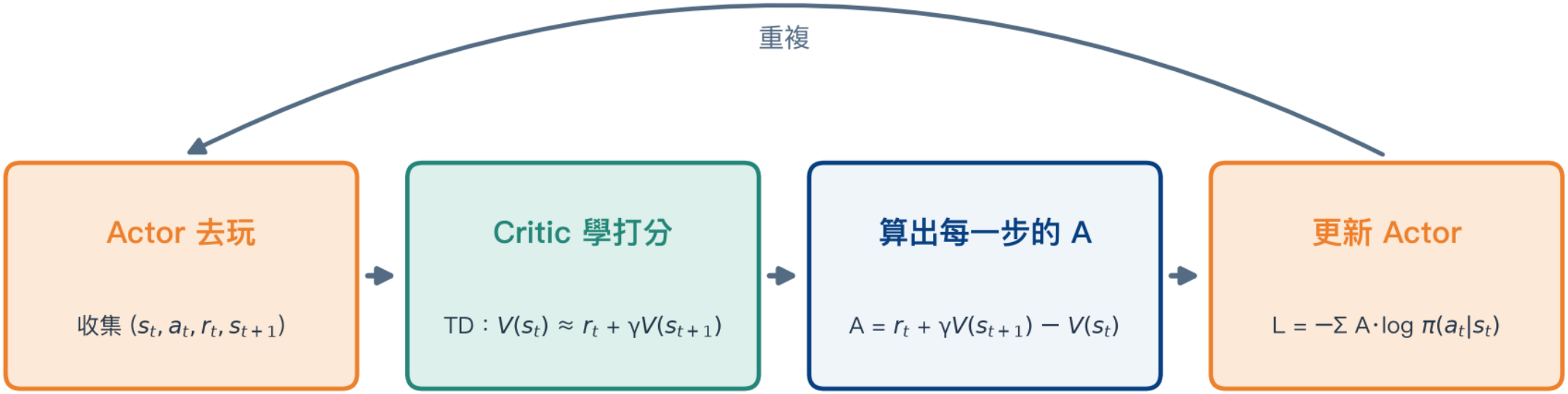


A > 0：a_t 比平均好（多做）；A < 0：a_t 比平均差（少做）

有了 Critic，回到 Version 3 的 baseline：用 V(s) 當「這個局面的平均」，A = 實際表現 − V(s)。

Actor-Critic：讓評論家估計 A

圖解

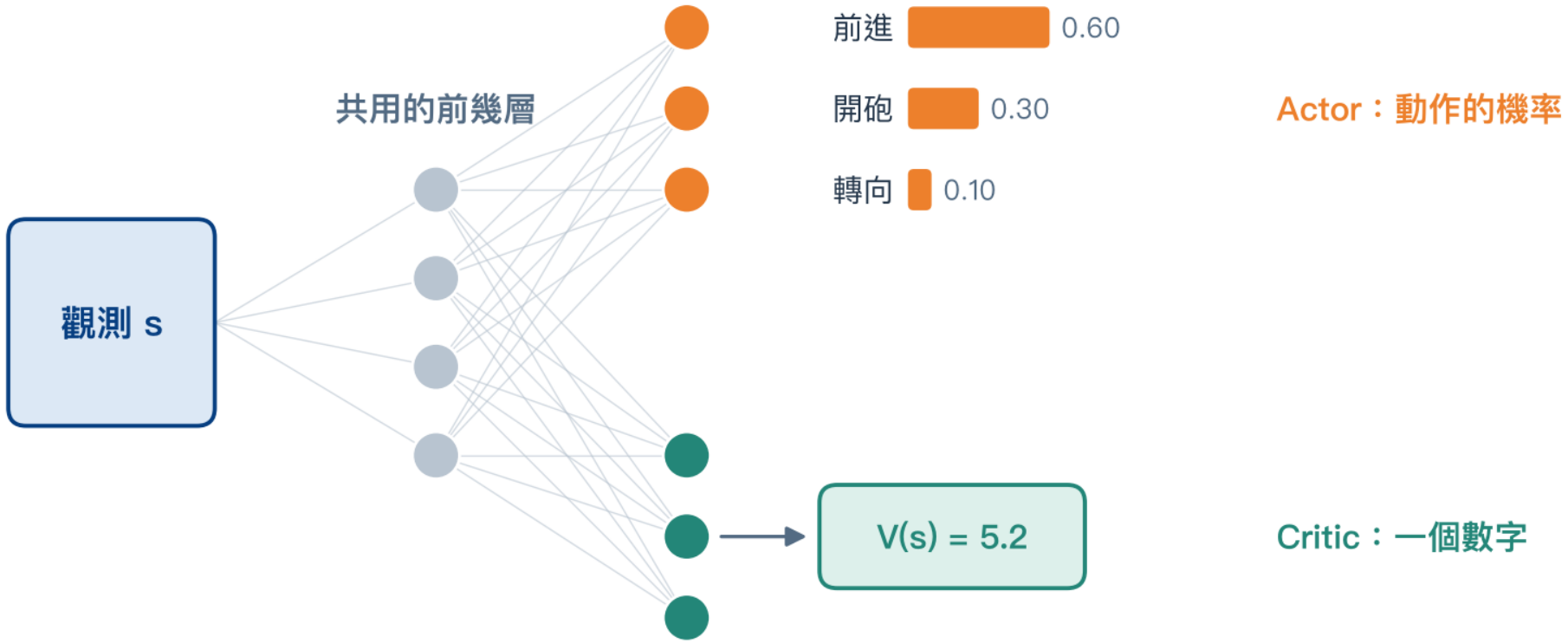


小技巧：Actor 和 Critic 吃一樣的觀測，前面幾層可以共用，只有最後一層分開

Actor 去玩、Critic 學打分、算出每一步的 A、再更新 Actor——一直重複。

圖解：Actor 與 Critic 兩個網路

圖解｜課後閱讀（放映略過）



Actor 和 Critic 吃一樣的觀測：前面幾層可以共用，只有最後分開

Actor 輸出動作機率、Critic 輸出這個局面的價值 V(s)；兩者合起來算出 A，再拿 A 更新 Actor。

另一種 Critic：Q(s, a) 與 Q 學習

銜接

- V(s)：這個局面值多少
- Q(s, a)：在這個局面「做 a」值多少
- 有了 Q，就算沒有 Actor 也能選：選 Q 最大的動作
- 這就是 Q 學習、DQN（打 Atari 的那個）

	V(s)	Q(s, a)
評估什麼	局面	局面 + 動作
怎麼用	當 baseline 給 Actor	直接選最大的動作
代表方法	Actor-Critic	Q 學習、DQN

Q 學習就是「只有 Critic、沒有 Actor」：接下來用它來推貝爾曼方程，並在坦克上驗證。

Q 表：每一格記「往這走，之後能拿幾分」

核心觀念

- 一張「狀態 × 動作」的大表格
- Q(s, a)：在狀態 s 做動作 a，之後總共能拿幾分
- 做決定：查表，選分數最高的動作（argmax）
- 最好的那張表叫 Q*；一開始全是 0，邊玩邊修正

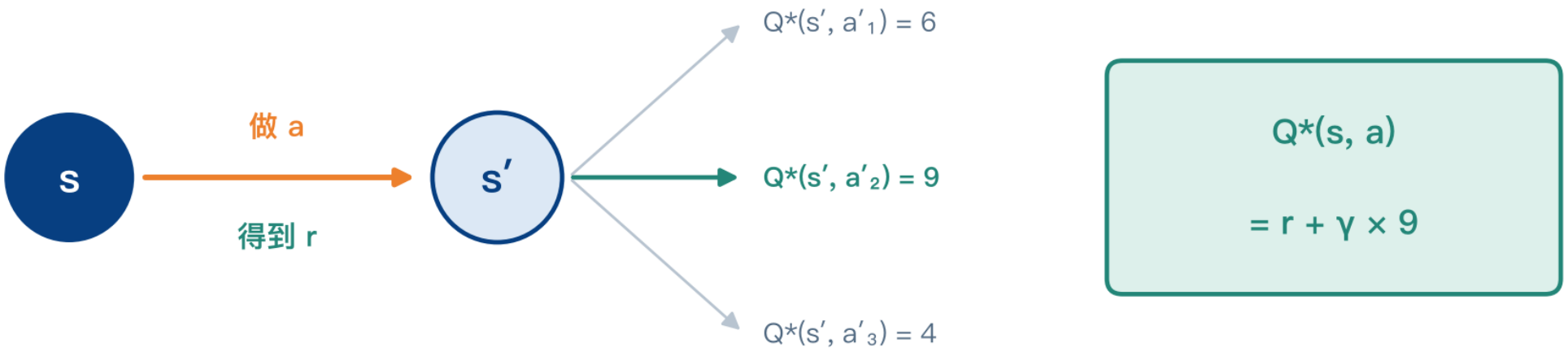


Q 值 =「做這個動作，未來的回報」的估計。

推導：貝爾曼方程

圖解

$$Q^*(s,a) = \mathbb{E}\left[r + \gamma \max_{a'} Q^*(s',a')\right]$$

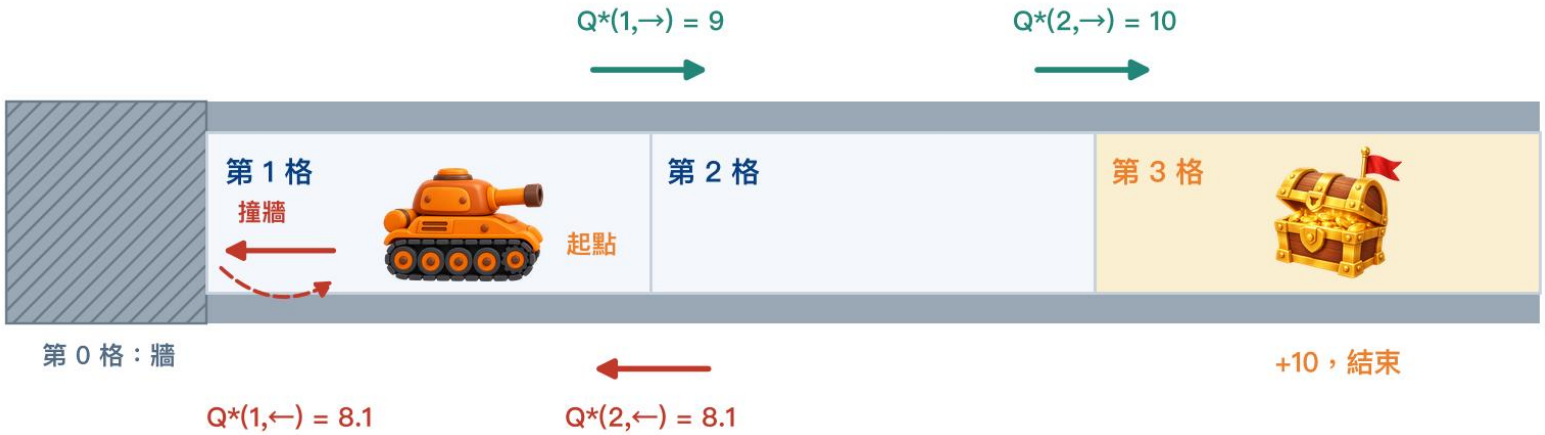


貝爾曼方程：現在的 Q ＝ 這一步的獎勵 ＋ γ × 下一個局面「最好動作」的 Q

貝爾曼方程：做這個動作的價值，由「下一格最好動作的價值」決定。

圖解：三格走廊的 Q*（拿到價值+接下來最好動作價值）

圖解



分數（Q）＝ 這一步的獎勵 ＋ 0.9 ×「走到的那一格，最好動作的分數」，從終點往回算：

Q* 的 * 是「最好的」：假設之後每一步都選最好的動作，能拿到的分數

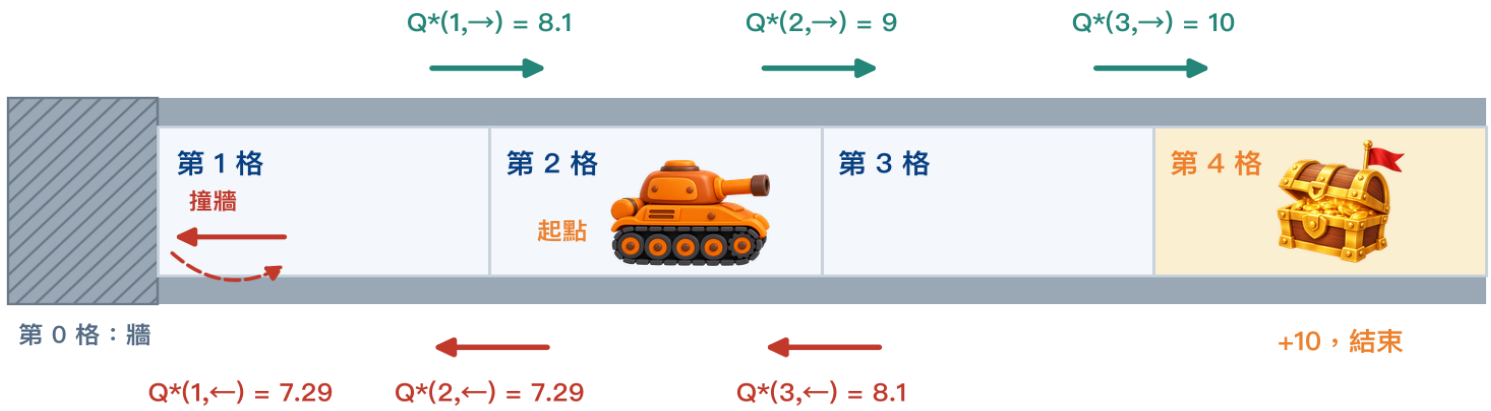
在哪一格、做什麼	這一步的獎勵	走到哪一格	那一格最好的動作（分數）	這個動作的分數
第 2 格，往右 →	+10	第 3 格（結束）	沒有了（0）	10 + 0.9 × 0 = 10
第 1 格，往右 →	0	第 2 格	往右（10）	0 + 0.9 × 10 = 9
第 2 格，往左 ←	0	第 1 格	往右（9）	0 + 0.9 × 9 = 8.1
第 1 格，往左 ←	0	第 1 格（撞牆沒動）	往右（9）	0 + 0.9 × 9 = 8.1

同一格出發，往右都比往左大 → 每一格都選往右；最後兩列都走到第 1 格，所以分數一樣

第 2 格往右直接拿 10；第 1 格往右多走一步，打一次折是 9；兩個往左都回到第 1 格，所以都是 8.1。

圖解：四格走廊的 Q*（多一格、多打一次折）

圖解



分數（Q）＝ 這一步的獎勵 ＋ 0.9 ×「走到的那一格，最好動作的分數」，從終點往回算：

Q* 的 * 是「最好的」：假設之後每一步都選最好的動作，能拿到的分數

在哪一格、做什麼	這一步的獎勵	走到哪一格	那一格最好的動作（分數）	這個動作的分數
第 3 格，往右 →	+10	第 4 格（結束）	沒有了（0）	10 + 0.9 × 0 = 10
第 2 格，往右 →	0	第 3 格	往右（10）	0 + 0.9 × 10 = 9
第 1 格，往右 →	0	第 2 格	往右（9）	0 + 0.9 × 9 = 8.1
第 3 格，往左 ←	0	第 2 格	往右（9）	0 + 0.9 × 9 = 8.1
第 2 格，往左 ←	0	第 1 格	往右（8.1）	0 + 0.9 × 8.1 = 7.29
第 1 格，往左 ←	0	第 1 格（撞牆沒動）	往右（8.1）	0 + 0.9 × 8.1 = 7.29

同一格出發，往右都比往左大 → 每一格都選往右；最後兩列都走到第 1 格，所以分數一樣

多一格，前面每一格就多打一次折：往右 10、9、8.1；往左也一樣往回推：8.1、7.29、7.29。

推導：Q 學習怎麼更新？往「這次的體驗」修一點

圖解

$$Q \leftarrow Q + \alpha\,(y - Q)\,,\quad y = r + \gamma \max_{a'} Q(s', a')$$

① 發生什麼事

現在的格子 s



原本以為：Q(s, 前進) = 2

前進

拿到 r = +1

走到下一格 s′

前進 5（最好）

左轉 3

開砲 1.5

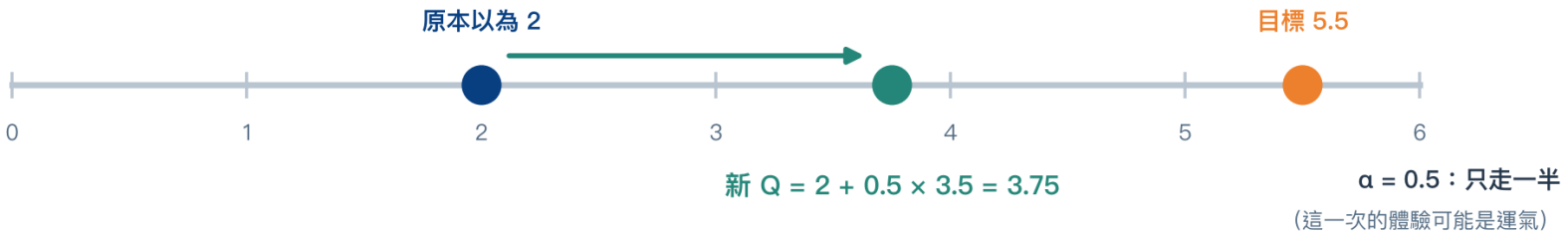
② 算目標（這次的體驗）

$y = r + \gamma \times$「下一格最好動作的 Q」

$= 1 + 0.9 \times 5 = 5.5$

比原本以為的 2 多了 3.5

③ 往目標修一點：新 Q = 舊 Q + α ×（目標 − 舊 Q）



為什麼叫梯度下降：把目標 y 當答案，損失 ½(y − Q)² 對 Q 微分是 −(y − Q)，沿著反方向走 α，就是這條式子（Class 2 的回歸）

算出這次的目標 y，比原本以為的 Q 多多少，就往那邊修一點（α 決定修多少）。

想一想⑤

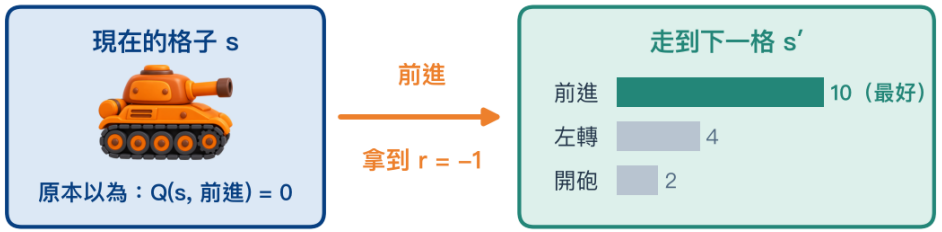
手算

照右圖的數字（α = 0.1、γ = 0.9），新的 Q(s, 前進) 是多少？

A　0.8

B　8

C　−0.1



α = 0.1，γ = 0.9

新的 Q(s, 前進) 是多少？

Colab A2 挑戰 2。

想一想⑤：解答

手算

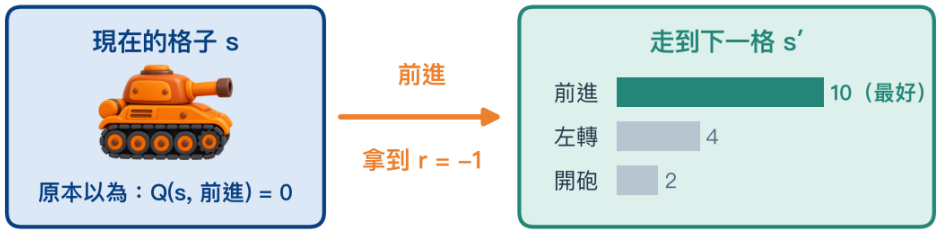
照右圖的數字（α = 0.1、γ = 0.9），新的 Q(s, 前進) 是多少？

A　0.8　✓

B　8

C　−0.1

目標 = −1 + 0.9×10 = 8；新 Q = 0 + 0.1×(8 − 0) = 0.8。被扣分，但通往好地方，Q 還是往上修。



α = 0.1，γ = 0.9

新的 Q(s, 前進) 是多少？

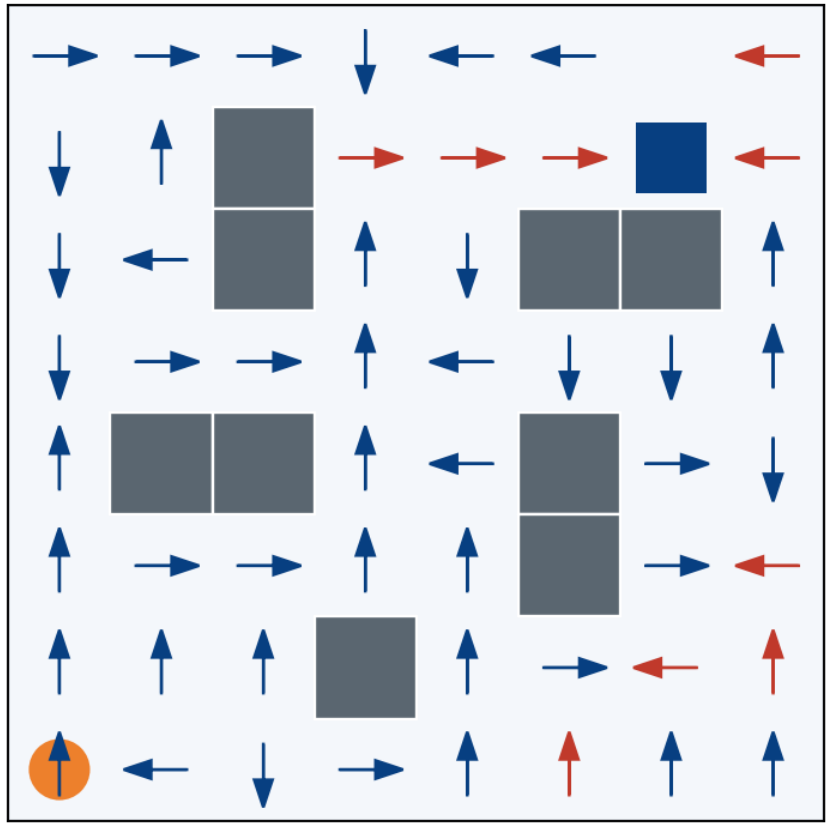
目標 = −1 + 0.9×10 = 8；新 Q = 0 + 0.1×(8 − 0) = 0.8。被扣分，但通往好地方，Q 還是往上修。

學好的 Q 表長什麼樣？

結果

- 坦克自己玩 8000 局學出來的
- 每一格都有一個「最想朝的方向」
- 紅色：對準敵人，開砲
- 從起點（橘）跟著箭頭走，就會打到敵人

學好的 Q 表：每一格最想朝哪裡（紅＝開砲）



沒有人畫這張地圖：它是從獎勵裡摸索出來的。

為什麼需要 Q？只有 V 不夠嗎？

圖解

$$V(s) = \max_a Q(s, a)$$ （這個局面的價值＝最好那個動作的分數）

只有 $V(s)$

「這個局面值 7 分」

前進 → 會走到哪個局面？

開砲 → 敵人會怎樣？

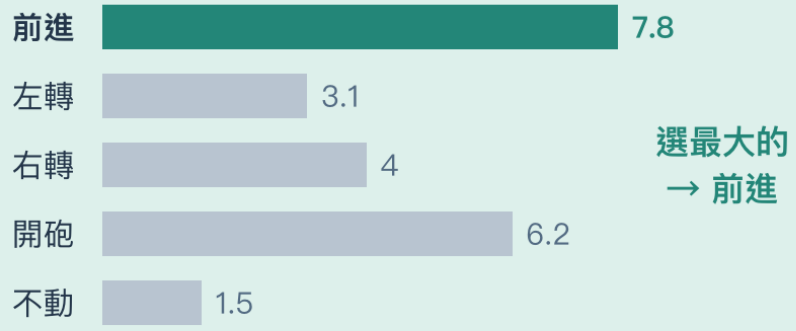
轉向 → 會看到什麼？

要先懂環境的規則，才能選動作

坦克通常不知道規則 → 選不出來

有 $Q(s, a)$

「每個動作各值幾分」



不用懂環境，比 5 個數字就能決定

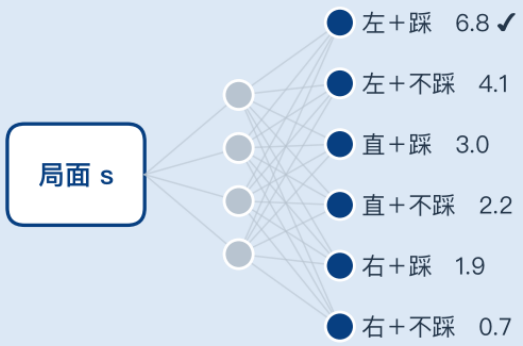
只有 V，要懂環境才能選動作；有 Q，直接比每個動作的分數就好。

總結：同時要做好幾個動作怎麼辦？（轉彎 + 煞車）

整理

例：開車時同時決定「方向盤（左、直、右）」和「煞車（踩、不踩）」

① 組合成一個動作

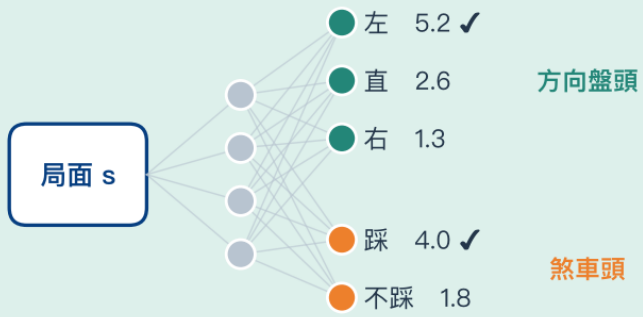


輸出 3 × 2 = 6 個 Q，選最大的

計算方式完全不變

缺點：組合數相乘，很快爆炸

② 每個動作一個頭（分支）



輸出 3 + 2 = 5 個，每個頭各選最大

期中網路也是 5 個頭（那是分類）

假設：各動作的好壞可以分開算

③ 連續的力道、角度

煞車 0～100%、方向盤 −30°～+30°

有無限多種 → 沒辦法列出來取 max

Actor 直接輸出動作 → Critic 替這個動作打分

「轉 15°、煞車 40%」→ Q = 6.3

改用 Actor–Critic（DDPG、SAC）

或切成幾段：0%、30%、60%、100%

自駕車、機器人多用這類方法

組合成一個動作最直接但會爆炸；每個動作一個頭比較省；連續的力道、角度就改用 Actor-Critic。

Q 學習 vs Actor-Critic：各自的優勢

整理

兩種方法都用到 Critic；差在動作怎麼選：Q 學習查分數選最大，Actor-Critic 由 Actor 輸出動作

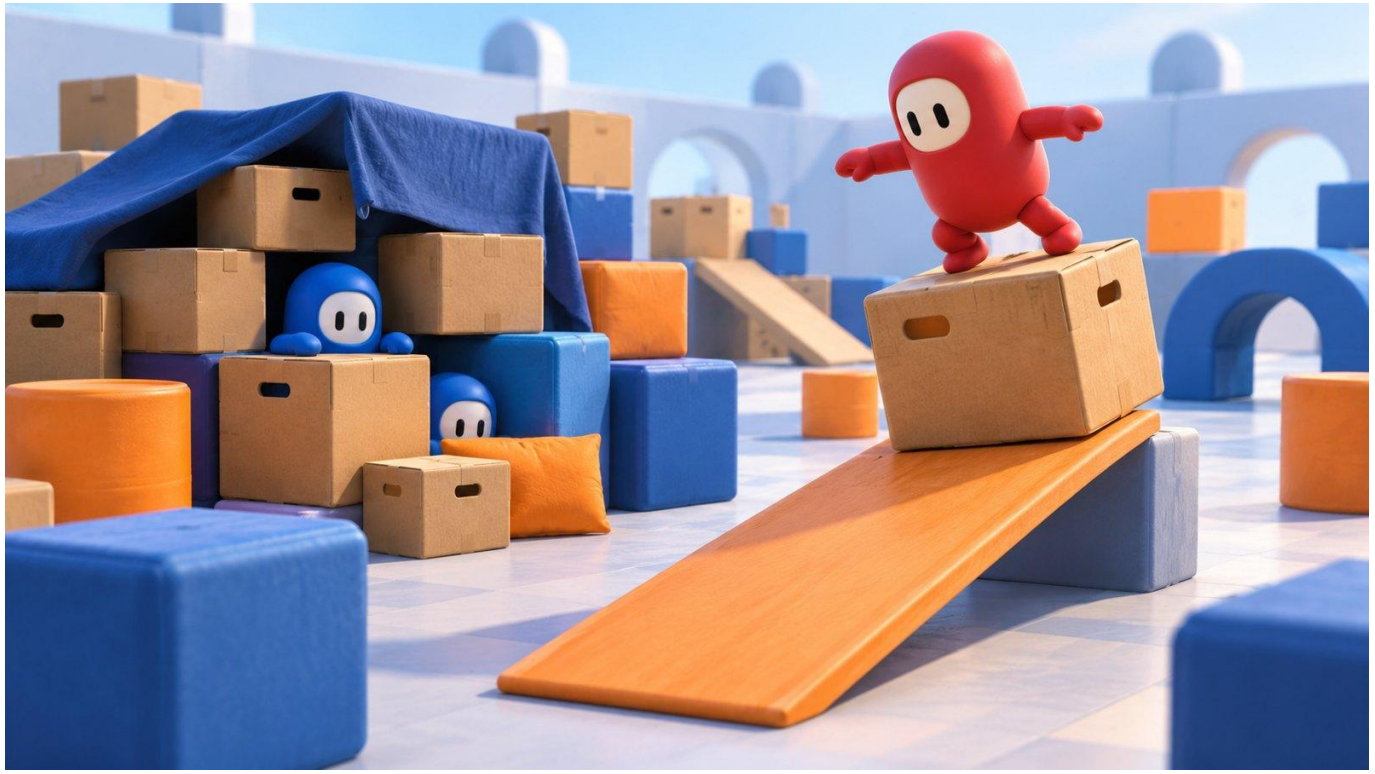
	Q 學習（DQN）	Actor-Critic（含 PPO）
要訓練幾個網路	1 個（Q）	2 個（Actor + Critic）
資料能不能重複用	**可以，所以要玩的局數較少（off-policy）✓**	只能用目前 Actor 玩的（on-policy）；PPO 可多用幾次
動作種類	要能一個一個列出來（5 個鍵）	**連續的也可以（角度、力道）✓**
動作很多時	組合會爆炸（坦克 108 種）	**每個動作一個頭，比較好處理 ✓**
探索	要自己加 ε-greedy	**照機率抽，本身就會探索 ✓**
隨機的策略	做不到（永遠選最大的）	**做得到（例如猜拳）✓**
訓練穩定度	換成網路後容易不穩、Q 值常被高估	**一般比較穩，但超參數要調 ✓**
代表應用	Atari（DQN）、我們的坦克練習	ChatGPT、DeepSeek-R1、機器人、自駕車

資料來源：DQN：Mnih et al., Nature 2015；PPO：Schulman et al., 2017；ChatGPT：Ouyang et al., 2022；DeepSeek-R1：DeepSeek-AI, Nature 2025

Q 學習：省資料、簡單、決定明確；Actor-Critic：連續動作、動作多、需要隨機時比較適合。

真實世界：它自己發現的招式

- 捉迷藏：躲的人學會用箱子築牆
- 抓的人學會搬坡道翻牆，還發明「站在箱子上衝浪」
- 躲的人再學會先把所有箱子鎖住
- 打磚塊：AI 自己發現先打穿一邊，讓球在上面反彈
- 沒有人教這些招式，只有輸贏



資料來源：捉迷藏：OpenAI（美國）· Baker et al., 2019；打磚塊：DeepMind（英國）· Mnih et al., Nature, 2015

只給獎勵、讓它自己玩，有時候會發明人類沒想到的方法。

真實世界：從遊戲到真實的身體

- 賽車：GT Sophy 在 Gran Turismo 贏過世界冠軍
- 練了約 4.5 萬小時的模擬
- 足球機器人：小型人形機器人學會跑、踢、跌倒後爬起來
- 先在模擬裡練，再搬到真的機器人上（sim-to-real）



資料來源：GT Sophy：Sony AI（日本）· Wurman et al., Nature, 2022；足球：Google DeepMind（英國）· Haarnoja et al., Science Robotics, 2024

強化學習最大的成本是「試很多次」：所以都先在模擬器裡練。

強化學習的里程碑（2015–2025）

從遊戲 → 棋類 → 真實物理世界 → 大型語言模型：每一個都靠「大量試錯 + 獎勵」，越往後越常「先模仿、再強化」

年	成果（機構）	為什麼重要
2015	DQN 玩 Atari（DeepMind，英國）	只看畫面，學會 49 款遊戲
2016	AlphaGo（DeepMind）	先模仿棋譜、再自我對弈，贏李世乭
2017	AlphaZero（DeepMind）	完全不看人類棋譜
2019	OpenAI Five、AlphaStar	多人合作、即時戰略
2022	核融合電漿控制（DeepMind + EPFL，瑞士）	控制真實的物理設備
2023	Swift 無人機（蘇黎世大學，瑞士）	真實運動中贏世界冠軍
2025	DeepSeek-R1（DeepSeek，中國）	用強化學習教出推理能力

資料來源：Mnih et al., Nature 2015；Silver et al., Nature 2016、Science 2018；Degrave et al., Nature 2022；Kaufmann et al., Nature 2023；DeepSeek-AI, Nature 2025

十年間，強化學習從「玩遊戲」走到「控制真實世界」和「教 AI 思考」。

商業上已經在用的地方

共同點：每一步的決定會影響之後的結果，而且能在模擬器或大量真實互動裡試錯

領域	例子	獎勵是什麼
推薦系統	YouTube 用 REINFORCE 推薦影片（Google，美國，2019）	使用者後續的觀看
叫車派單	滴滴的訂單派送（中國，2018）	長期的總成交量
資料中心	Google 資料中心冷卻（DeepMind，2016）	冷卻耗電降低（約 40%）
大型語言模型	ChatGPT 的 RLHF（OpenAI，美國，2022）	人類偏好的評分模型
推理模型	OpenAI o1（2024）、DeepSeek-R1（2025）	數學、程式答對與否

資料來源：Chen et al.（Google），WSDM 2019；Xu et al.（滴滴），KDD 2018；DeepMind 部落格，2016；Ouyang et al.（OpenAI），2022；OpenAI，2024；DeepSeek-AI, Nature 2025

只要決策會影響未來、而且能大量試錯，強化學習就有用武之地。

休息 10 分鐘

10 分鐘後回來

1. 打開 Class 4 Colab
2. 檔案 → 在雲端硬碟中儲存副本
3. 只執行第一格「準備工具」
   其他格上課時逐格做



回來前，先打開 Colab、另存副本、全部執行。

第三段　獎勵與示範：它只在乎你給的分數

章節導覽

上一段

第二段：Policy Gradient、Critic、Q 學習，都是照獎勵學。

這一段要回答

獎勵太少，或根本寫不出獎勵時，怎麼辦？



1. 難題一：獎勵太稀疏 → 中途給提示 (reward shaping)
2. 提示會被鑽：怎麼寫才安全
3. 難題二：連獎勵都寫不出來 → 模仿學習、逆強化學習

強化學習最難的部分，常常不是演算法，而是獎勵。

獎勵的兩個難題：太稀疏、寫不出來

章節導覽

難題一：有獎勵，但太稀疏

例：只有「打贏 +10」，900 步才拿到一次

Reward Shaping：中途給提示

存活、被打中、開砲、靠近、命中……

但提示會被鑽 → 位能型提示

走回去要扣回去

其他：好奇心、課程學習

先練簡單的，再練難的

難題二：連獎勵都寫不出來

例：開車、對話——「好」很難寫成分數

模仿學習（Behavior Cloning）

直接抄專家的動作＝你的期中

DAgger：補上偏離後的資料

讓模型自己開，專家標答案

逆強化學習（IRL）

從示範反推獎勵，再用 RL 學

強化學習要有獎勵。獎勵太少 → 中途給提示；連獎勵都寫不出來 → 模仿專家，或從示範反推獎勵。

稀疏獎勵怎麼辦？Reward Shaping：中途給提示

圖解



坦克可以給哪些提示？每一個都是「想鼓勵某件事」，但也可能被鑽

提示（滑桿名稱）	想鼓勵什麼	可能被怎麼鑽（後面驗證）
每步存活 +	活久一點，不要送死	躲在角落不動，活滿全場
被打中 −（生命損失）	別亂衝、會躲子彈	為了不被打，乾脆不靠近敵人
開砲 +	多攻擊	原地一直開砲，根本沒瞄準
靠近敵人 +	主動去找敵人	走遠再走近，來回刷分
命中敵人 +	打得準	要真的打到才有，比較難鑽

只有輸贏時幾乎都是 0；加上中途提示，每一步都有訊號——但每個提示都可能被鑽。

獎勵怎麼寫成程式？以「靠近敵人」為例

圖解

① 動作前、後各看一次



before = dist(pos) → 走一步 → after = dist(pos)

② 判斷發生了什麼事

靠近敵人	after < before	✓ 1
遠離敵人	after > before	✗ 0
命中敵人	開砲且對準	✗ 0
被打中	敵人打到我	✗ 0
每步存活	還活著	✓ 1

③ 每件事的分數 × 有沒有發生，加起來

靠近 +1 × 1

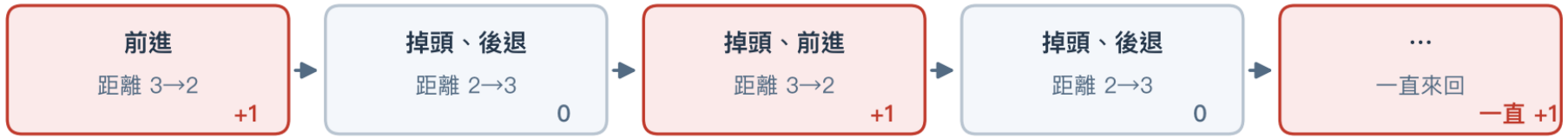
遠離 0 × 0　命中 +1 × 0

每步存活 −0.1 × 1

r = +0.9

滑桿調的就是這些「分數」

只有「靠近 +1」時的漏洞：掉頭走回去沒有扣分，再走過來又 +1



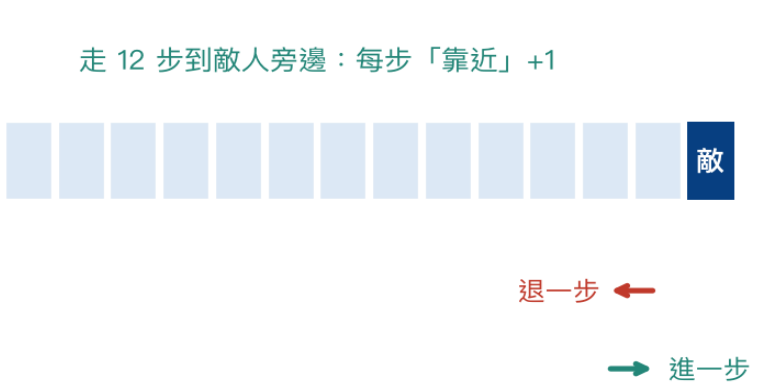
修法：遠離也給 −1（走回去扣回去），這叫位能型提示，後面推導

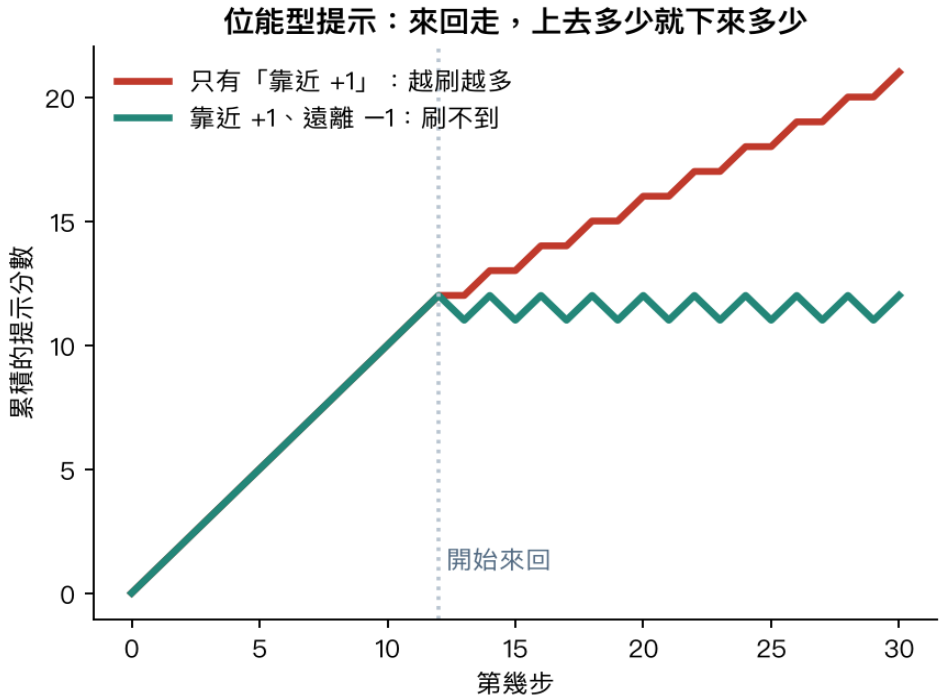
提示寫成程式：比較前後、判斷事件、分數加總；只獎勵靠近，來回走就一直拿分。

推導：怎樣的中途提示不會被刷？

圖解

位能型提示：$F = \gamma\Phi(s') - \Phi(s)$（簡化版：靠近 $+1$、遠離 -1）





提示＝位能（離敵人多近）的變化：總和只和起點、終點有關，繞圈不會多拿

資料來源：Ng, Harada & Russell（美國・UC Berkeley）・ICML 1999

實驗：只靠近 +1 → 勝率 0%；靠近 +1、遠離 −1 → 勝率 83%。

你獎勵什麼，它就學什麼

核心觀念

- 賽艇遊戲：撿到分數道具就加分
- AI 發現：原地繞圈撿道具，比跑完全程分數高
- 它沒有錯：完美做到了「你說的」
- 只是不是「你想的」



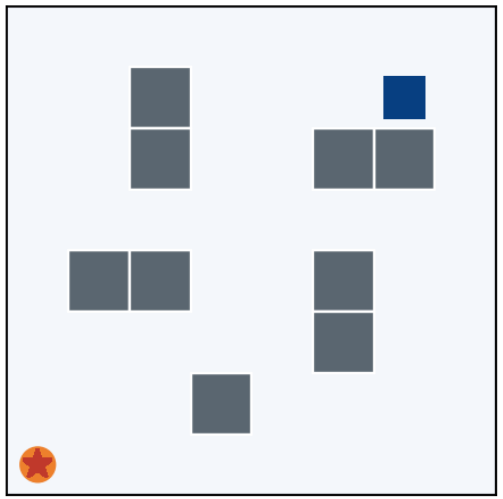
資料來源：OpenAI（美國）· Faulty Reward Functions in the Wild, 2016

獎勵是你和 AI 之間唯一的溝通管道。

我們的坦克也一樣

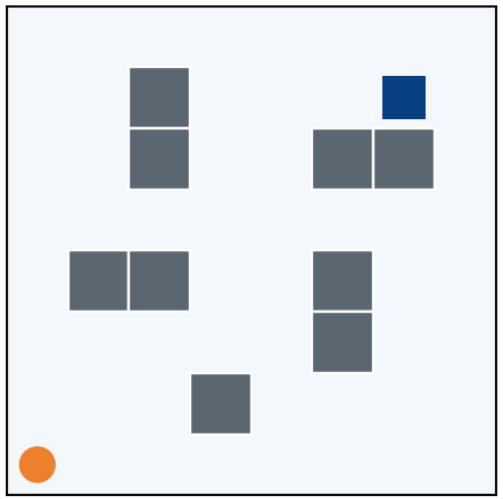
實驗

只獎勵開砲



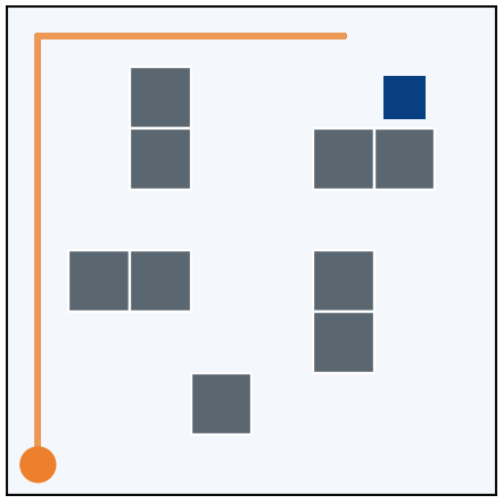
勝率 0% | 最常做：開砲 60 次

只獎勵存活



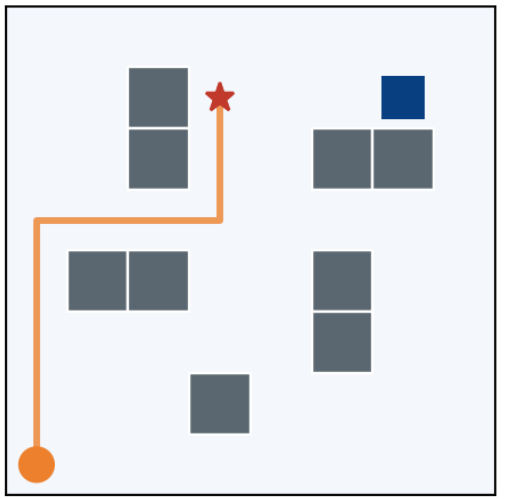
勝率 0% | 最常做：不動 60 次

只獎勵靠近敵人



勝率 0% | 最常做：前進 45 次

設計得不錯



勝率 80% | 最常做：前進 9 次

橘線＝坦克走過的路、紅星＝開砲；深藍方塊＝敵人、橘圓＝出發點、淡紅＝敵人射程

只獎勵開砲：原地狂開砲；只獎勵存活：躲著不動；只獎勵靠近：繞路刷分。設計好的獎勵：勝率 80%。

想一想⑥

思考題

只獎勵「每活一步 +1」，坦克會學到什麼？

A　躲在角落，永遠不出來

B　衝去把敵人打掉

C　一直開砲



它不是懶，是聰明：在你的規則下，躲起來就是最佳解。

想一想⑥：解答

思考題

只獎勵「每活一步 +1」，坦克會學到什麼？

A　躲在角落，永遠不出來　✓

B　衝去把敵人打掉

C　一直開砲

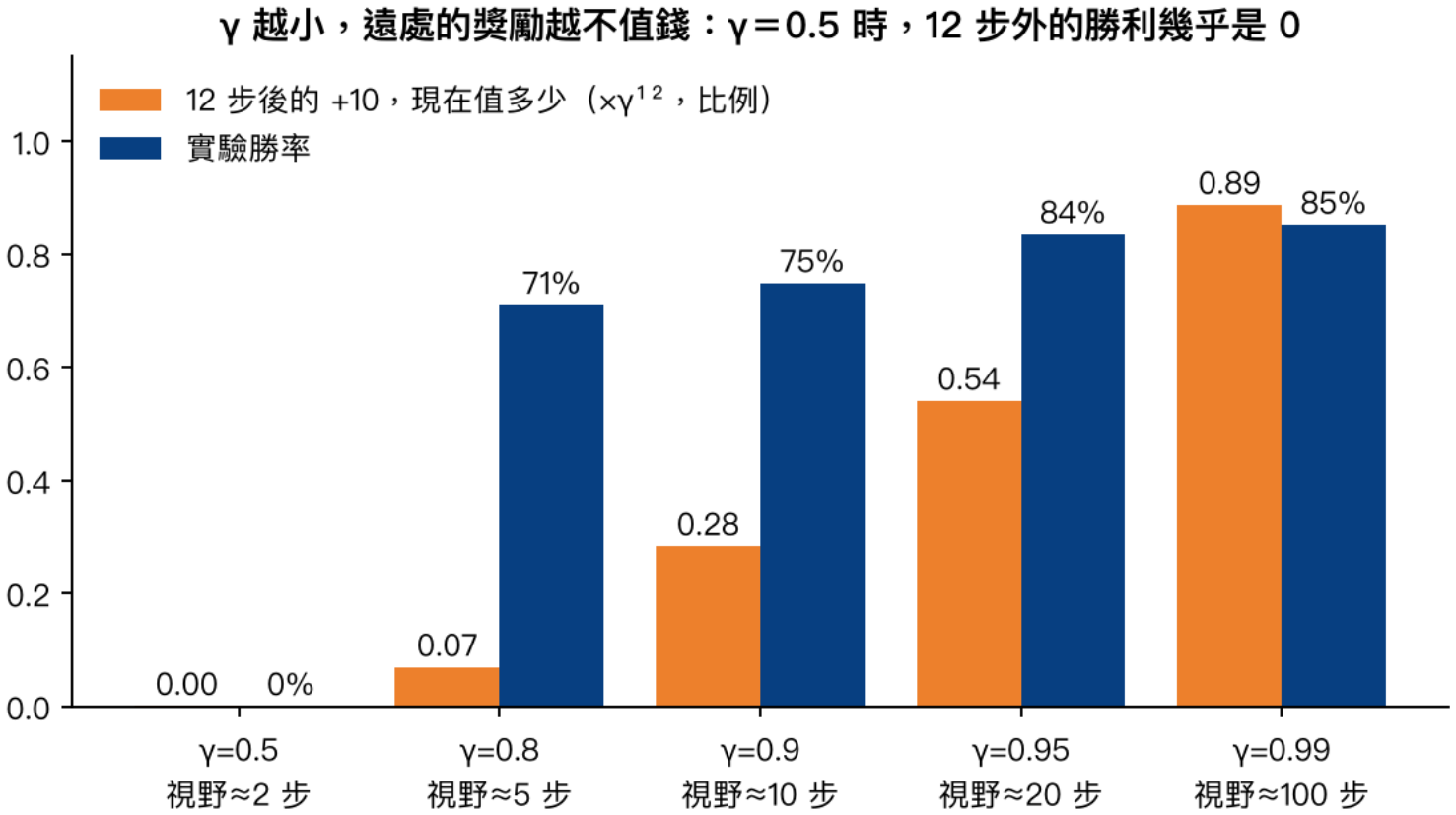
去打敵人會被打，被打死就沒分；躲著不動最安全，分數最高。
期中模擬也發現過：完全不動的坦克能拿不少分。



去打敵人會被打，被打死就沒分；躲著不動最安全，分數最高。期中模擬也發現過

驗證：γ 也會改變學到的行為

- 起點離敵人 12 步
- γ 0.5：視野約 2 步，12 步外的勝利剩 $0.5^{12} \approx 0.0002$ → 勝率 0%
- 走過去「不值得」，乾脆不動
- γ 0.95：視野約 20 步 → 勝率 84%
- γ 0.8 仍然學得會：一路上有「命中 +1」的提示



同一組獎勵，只改 γ：用第一段的「有效視野」就能解釋。

好的獎勵怎麼設計

整理

- 先想清楚：你真正要的是什麼
- 每加一項獎勵，就算一次：它的最佳解是什麼？
- 中途提示用位能型：走回來會扣回去
- 一定要看它實際在做什麼

原則	坦克的例子
獎勵「結果」，不是「動作」	擊殺、命中 ✓　開砲 ✗
中途提示用位能型	靠近 +1、遠離 −1 ✓　只有靠近 +1 ✗
給時間成本	每步 −0.1：不要拖
懲罰真正不要的	被打中 −1
γ 要看得到目標	有效視野要比路程長（γ = 0.9 約 10 步）

設計獎勵，就是寫一份不會被鑽漏洞的規則——而且可以先用數學檢查。

想一想⑦

思考題

期中每場得分 = 擊殺 + 0.8×剩餘血量。直接拿來當獎勵，最可能被鑽什麼漏洞？

A　躲在角落保血，不去打

B　一直衝撞敵人

C　把技能用光



你設計的評分方式，會決定大家（和 AI）怎麼玩。

想一想⑦：解答

思考題

期中每場得分 = 擊殺 + 0.8×剩餘血量。直接拿來當獎勵，最可能被鑽什麼漏洞？

A　躲在角落保血，不去打　✓

B　一直衝撞敵人

C　把技能用光



血量占了很大的比重：不打也能拿 0.8 分。這也是期中計分要用「擊殺」和稱號鼓勵交戰的原因。

血量占了很大的比重

沒有獎勵，只有示範：模仿學習

核心觀念

- 難題二：很多事情寫不出獎勵（開車、對話）
- 但有專家可以示範 → 照著學（Behavior Cloning）
- 這就是你的期中：照規則策略學
- 問題 1：專家沒示範過的情況，不知道怎麼辦（下一頁）
- 問題 2：連不重要的習慣也一起學

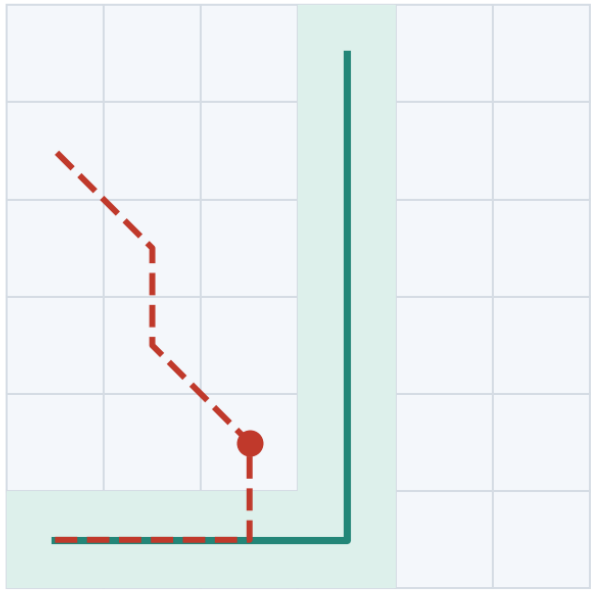


難題二：寫不出獎勵，就改看示範。Behavior Cloning = 監督式學習；問題出在「示範沒涵蓋到的地方」。

只抄示範的問題：偏一步就沒看過（DAgger 怎麼補）

圖解

只抄示範：偏一步，就到了沒看過的地方

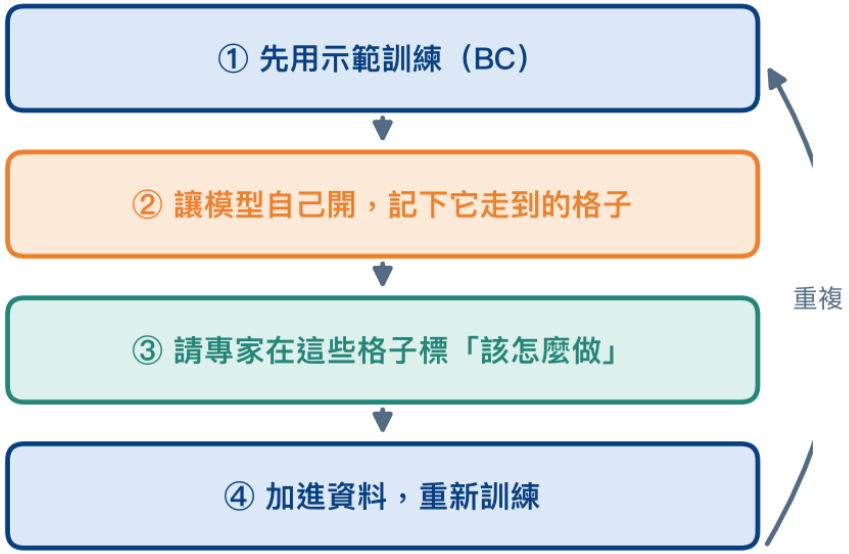


綠：專家示範走過的格子
（資料裡只有這些）

紅點：偶爾做錯一步
（手滑、被撞、地圖不同）
＝實驗裡的「被推一下」

紅虛線：這裡沒看過，
只能亂猜 → 越走越偏

DAgger：讓學生自己開，專家在旁邊標答案



資料慢慢補上「偏離之後怎麼回來」

期中：拿模型自己打出來的局面，請規則策略標動作

偶爾做錯一步就到了沒看過的地方；DAgger：讓模型自己開，專家在旁邊標答案。

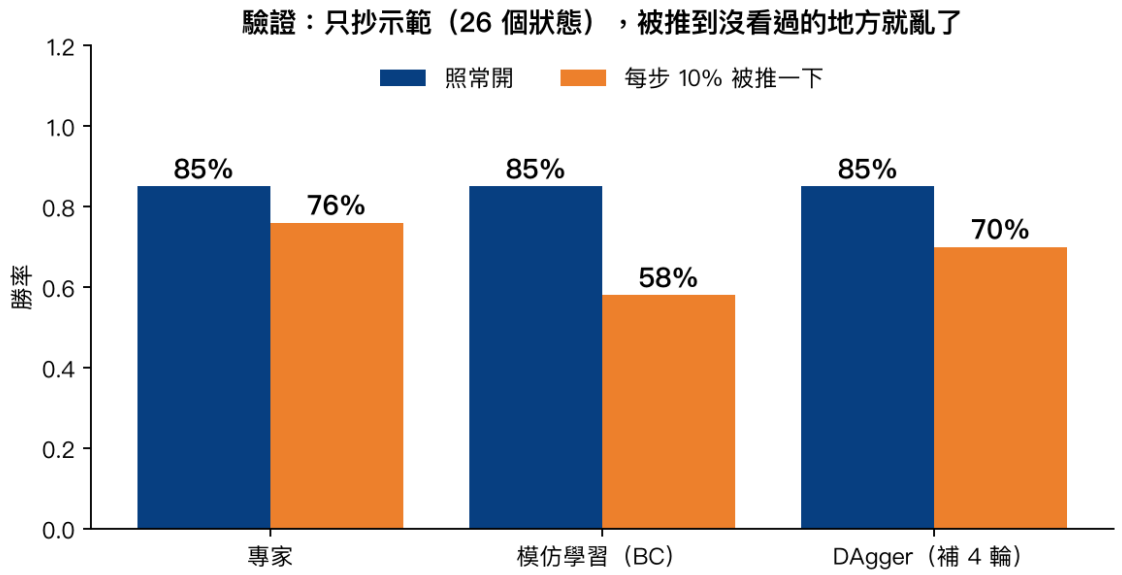
驗證：偶爾做錯一步（被推一下），誰撐得住？

驗證

①強化學習 ②怎麼學 ③獎勵示範 ④練習 ⑤期中

勝率	照常開	每步 10% 隨便做（被推一下）
專家（Q 學習）	85%	76%
模仿學習 BC	85%	58%
DAgger（補 4 輪）	85%	70% ✓

- 專家示範 50 局，只走過 26 個狀態；DAgger 補到 153 個
- ✓：被推一下時，學習者裡最好的



資料來源：DAgger：Ross, Gordon & Bagnell（美國，卡內基美隆大學），AISTATS 2011

照常開時 BC 和專家差不多；一旦偶爾做錯一步，BC 掉最多，DAgger 補過資料最穩。

逆強化學習：把強化學習反過來

圖解

強化學習：知道獎勵 → 學出行為



坦克：人寫「擊殺 +10、命中 +1…」
→ 它自己學會怎麼打

逆強化學習：看行為 → 反推獎勵 → 再用強化學習



和模仿學習差在哪？

模仿學習：抄「這個畫面按什麼鍵」——沒看過的畫面就不知道

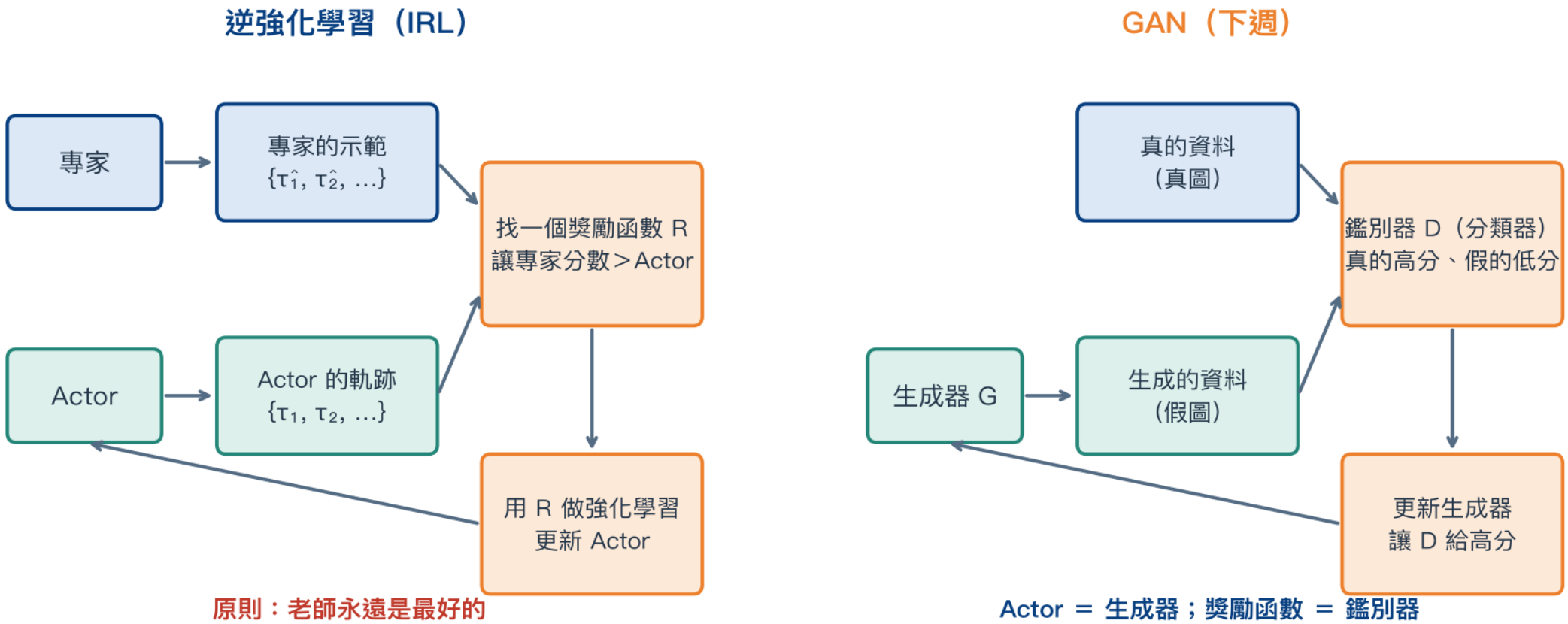
逆強化學習：猜「專家在乎什麼」（打得快？不被打？）——換地圖、遇到沒看過的情況，也能自己想辦法

資料來源：Ng & Russell（美國，UC Berkeley），ICML 2000

強化學習：知道獎勵 → 學行為。逆強化學習：看專家的行為 → 反推它在乎什麼（獎勵）→ 再用強化學習。

圖解：反推獎勵是一個迴圈（和 GAN 同一個架構）

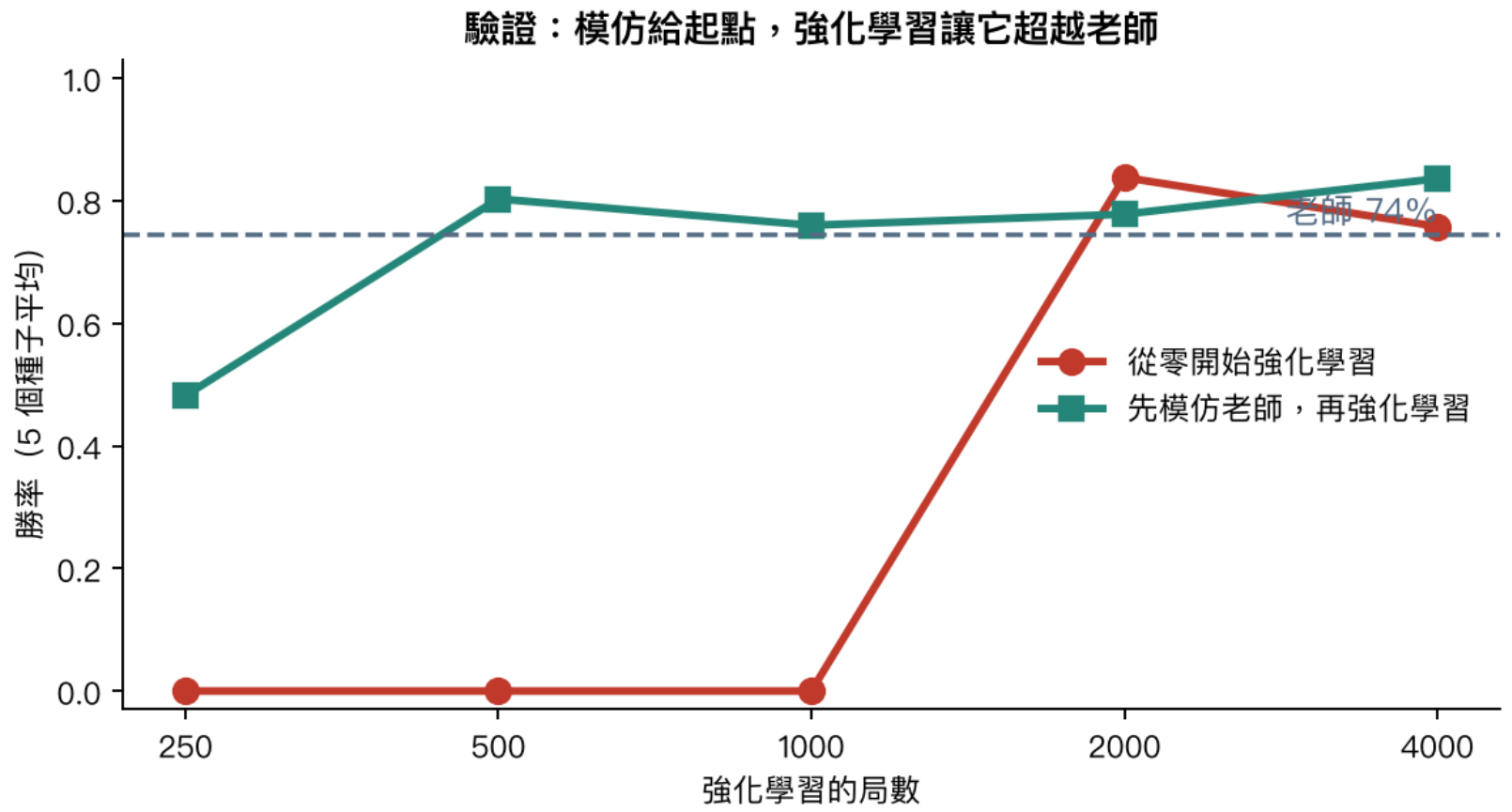
圖解 專家做得比較多的事，提高它在獎勵裡的分數；Actor 做得比較多的事，降低分數



資料來源：Ng & Russell（美國，UC Berkeley），ICML 2000；GAIL：Ho & Ermon（美國，史丹佛大學），NeurIPS 2016

找獎勵讓專家贏 → Actor 用它學 → 再找新獎勵，一直重複；獎勵換成分類器，就是下週的 GAN。

驗證：先模仿、再強化，超越老師



老師（不完美的專家）勝率 75%；從零開始要練到 2000 局才學會；先模仿再強化，500 局就有 80%，4000 局 84%。

第四段　練習：自己動手訓練

章節導覽

上一段

強化學習、模仿學習、先模仿再強化。

這一段要回答

同一套方法，換到不同的世界會學到什麼？

1. 坦克：獎勵設計、模仿與強化
2. 冰湖：地板會滑的世界
3. 21 點：AI 學出來的策略



先預測，再按 ▶：猜錯的地方，就是今天真正學到的地方。

實作 A、B：手算與坦克獎勵設計

▶ 動手做・打開 Colab

動手做

①強化學習 ②怎麼學 ③獎勵示範 ④練習 ⑤期中

建議 30 分鐘　Colab 步驟

- ☐ 必做 A1、A2：手算回報與 Q 值更新
- ☐ 必做 B0：一步一步看獎勵怎麼算
- ☐ 必做 B1：依序看三個陷阱獎勵的動畫
- ☐ 必做 B2：設計自己的獎勵，衝排行榜



完成標準

勝率 ≥ 80%、平均 ≤ 25 步，並寫出每一項獎勵為什麼這樣給

每改一次，先猜它會學到什麼，再按 ▶。

冰湖：地板會滑的世界

練習

- 4×4 的冰湖：走到寶藏 +1，掉進冰洞結束
- 地板會滑：有一定機率滑到左右兩側
- 不會滑時：最短路徑 6 步就好
- 會滑時：AI 學到的路線很奇怪——
- 它常常故意朝牆走！



環境有隨機性時，「最短」不一定「最好」。

想一想⑧

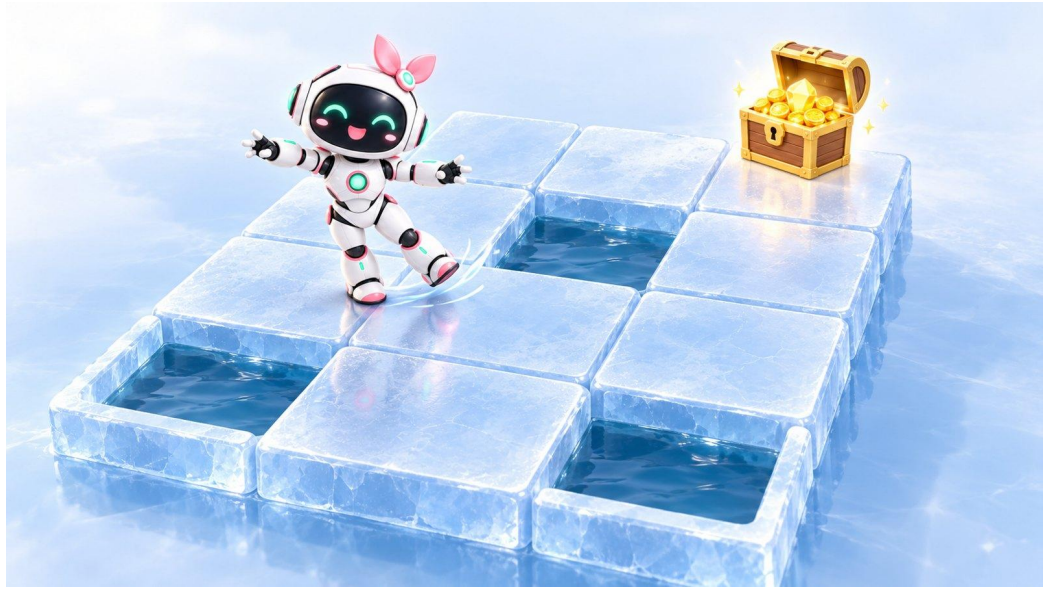
思考題

冰湖會滑時，AI 學到在某些格子「故意朝牆走」。為什麼？

A　撞牆不會動，只會往旁邊滑，比較不會掉進冰洞

B　AI 學壞了

C　牆旁邊有獎勵



在 Colab C 把滑倒機率改成 0，再看一次路線。

想一想⑧：解答

思考題

冰湖會滑時，AI 學到在某些格子「故意朝牆走」。為什麼？

A　撞牆不會動，只會往旁邊滑，比較不會掉進冰洞　✓

B　AI 學壞了

C　牆旁邊有獎勵



朝牆走時，「成功」就是原地不動，「滑倒」才會往左右移動；選一面牆，讓滑倒的方向剛好避開冰洞，就能安全地慢慢靠近寶藏。

朝牆走時，「成功」就是原地不動，「滑倒」才會往左右移動；選一面牆，讓滑倒的方向剛好避開冰洞，就能安全地慢慢靠近寶藏。

實作 B4、C、D：模仿 + 強化、冰湖、21 點

▶ 動手做．打開 Colab

動手做

建議 15 分鐘　Colab 步驟

- ☐ 必做 B4：先模仿、再強化，看能不能超越老師
- ☐ C 冰湖：改滑倒機率，看路線怎麼變
- ☐ D 21 點：先自己填策略，再和 AI 比
- ☐ 示範（課後再跑）：B3 模仿學習的陷阱



完成標準

B4 找到超越老師勝率的局數；C、D 擇一，說出 AI 的策略和你的直覺哪裡不同

每個練習都先寫下你的預測，再按 ▶。

第五段　期中專題時間

章節導覽

上一段

今天：獎勵、示範、分布偏移、先模仿再強化。

這一段要回答

你的坦克，現在卡在哪裡？



1. 全班現況
2. 用今天的觀念看你的坦克
3. 截止前檢查

今天學的每一個觀念，都能拿來診斷你的期中坦克。

期中現況（11/5 更新）

期中

①強化學習 ②怎麼學 ③獎勵示範 ④練習 **⑤期中**

- （11/5 由網站資料自動更新）
- 已繳交模型人數
- 天梯前段分數分布
- 各機體、模組的勝率
- 常見的上傳失敗原因



（11/5 更新）

用今天的觀念看你的坦克

期中

①強化學習 ②怎麼學 ③獎勵示範 ④練習 **⑤期中**

先找出你的坦克是哪一種，再對症下藥

現象	今天的觀念	可以怎麼做
卡牆、卡角落	分布偏移：規則策略很少卡住，示範裡沒有「脫困」	加脫困規則再錄；或讓模型自己打、再請規則策略標註（DAgger）
開火太少	資料不平衡：開火只佔很少的步數	提高 FIRE_WEIGHT；多錄交戰的局面
只會躲	鑽漏洞：躲著也能拿血量分	規則策略主動找敵人；計分裡擊殺比重高
學不像規則策略	觀測裡看不到規則策略用的資訊	疊幀或 LSTM 補記憶（Class 3）

模型學得不好時，先問：是資料沒教到，還是根本看不到？

截止前檢查清單

期中

①強化學習 ②怎麼學 ③獎勵示範 ④練習 **⑤期中**

上場版本	繳交網站「③ 繳交」：有一個「上場中」的 model.pt
規則策略	「③ 繳交」的規則策略那一格是綠色（網站錄過，或另外上傳）
期中報告	PDF 已上傳，4 個表格都填了
天梯	看一下排行榜：你的坦克有出現、場數有在增加
最後上傳	截止前一小時不要再大改，留時間給天梯跑

截止時間以繳交網站顯示的為準。

討論：用分數帶人

討論 1 如果你是老師，用什麼獎勵讓學生「真的學會」，而不是只為了分數？

討論 2 期中的計分方式，有哪些地方可能被鑽漏洞？你會怎麼改？

討論 3 公司的 KPI 是不是一種獎勵？你看過哪些「為了 KPI 而 KPI」的例子？

延伸 一個指標一旦變成目標，就不再是好指標（古德哈特定律）

延伸 過度嚴格的獎懲，不如精妙的機制設計

強化學習的難題，也是管理與教育的難題。

本堂重點

總結

強化學習	三步驟：Actor、讓回報最大、最佳化；回報可以從最後往回算，越遠的打越多折（γ 決定看多遠）。
怎麼學	Policy Gradient = 加權的分類（A 的版本 0→3）；Critic 給局面打分（MC、TD），用來估 A；Q 學習：只靠 Critic 選動作。
獎勵設計	獎勵的最佳解可以先算；鑽漏洞是獎勵的問題，不是 AI 笨；中途提示用位能型。
寫不出獎勵	模仿學習（期中）要小心分布偏移（DAgger）；逆強化學習從示範反推獎勵。
模仿 + 強化	先模仿給起點、強化學習超越老師：AlphaGo、虛擬果蠅、你的下一步。

下週：生成模型——GAN 與擴散模型。GAN 的鑑別器，就是今天逆強化學習裡那個「分類器」。