**強化學習RL懶人包**

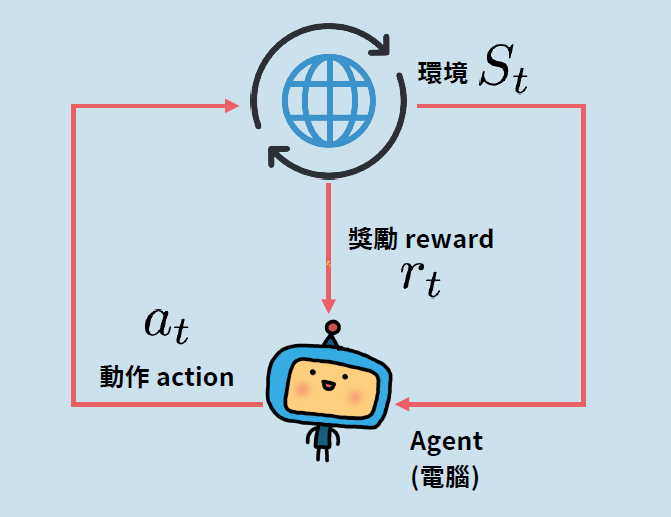
近日鴻海發表了三款電動車，產業分析師們除了讚頌鴻海的執行效率外，一個疑惑也浮上檯面，也就是足發展電動車硬體外，那另一重要的軟體自動駕駛系統呢?

機械學習在自動駕駛系統中扮演相當大的角色，這也是市場領先者(特斯拉)的賣點之一，支持這領域發展的是強化學習(Reinforcement learning) RL。

RL能解決許多開發應用上面臨的艱難決策問題，包括產業自動化、自主駕駛、電玩競技遊戲以及機器人等，以下就來介紹RL的原理吧。

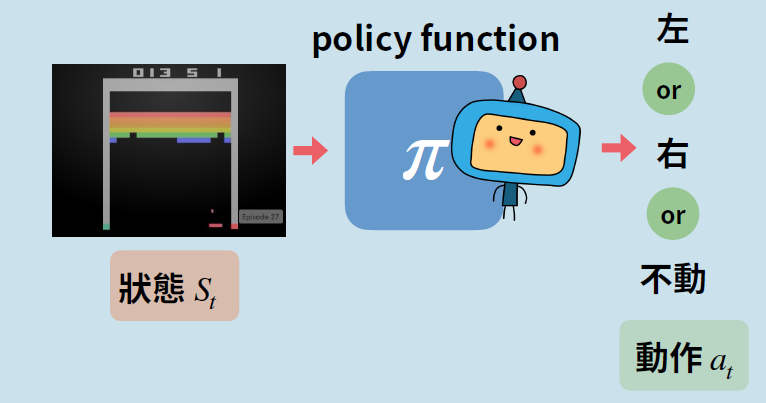
**強化學習 Reinforcement Learning(RL)**

強化學習是機器學習(Machine learning)的一種，電腦透過與一個動態(dynamic)環境不斷重複地互動，來學習正確地執行一項任務。這種嘗試錯誤(trial-and-error)的學習方法，使電腦在沒有人類干預、沒有被寫入明確的執行任務程式下，就能夠做出一系列的決策。最著名的強化學習案例就是AlphaGo。



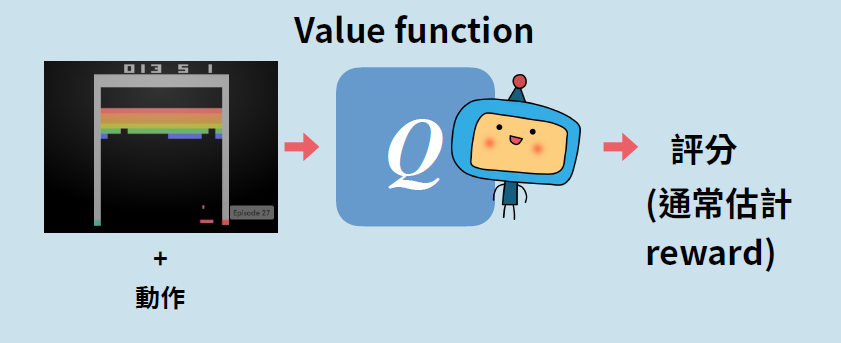
我們以教電腦玩打磚塊遊戲為例，可能的方式有兩種。

第一種是Policy Based

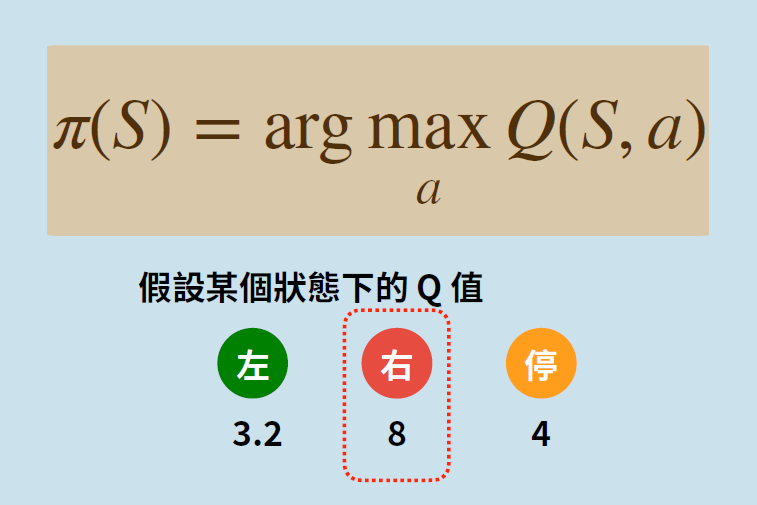
****

這個policy function很難準備資料，需要事先窮舉各種可能性，這好比我們初入門學寫程式時，用程式語言寫五子棋遊戲一樣，要將各種棋步判定規則撰寫於程式中。可以了解這不是好方法。

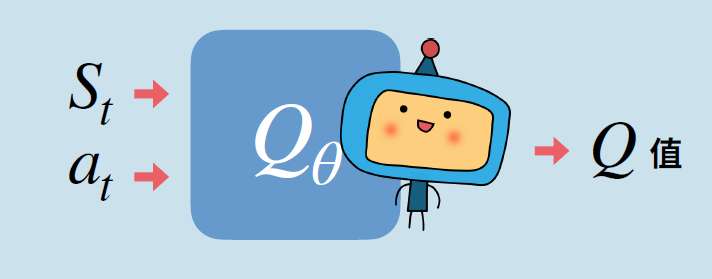
第二種是Value Base



Value Base的原理就是利用點腦演算力量把所有的動作都帶入Q函數，看看哪一個分數最高，以為下一步之決策。



而RL就是Value Base的演算法，最常見的是以Deep Q-learning (DQN)方式來達到目的。

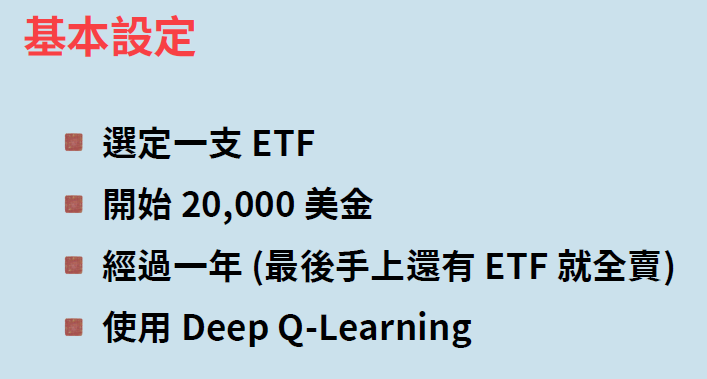


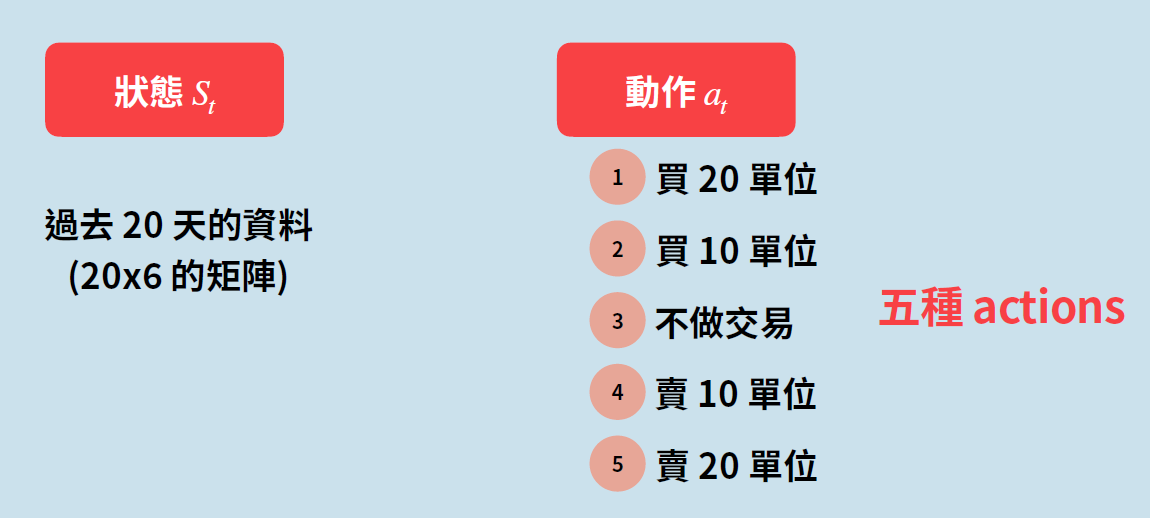
簡單的說就是讓電腦自己去玩，得到初步的Q值，然後用這些Q值當訓練資料，以捲機神經網路，自動生成初版的Q函式。然後不斷地讓電腦依Q函式決定最好的動作(greedy policy)，並更新Q函式，直到結果讓人大致滿意，當然RL強調的是持續學習及修正模型，所以以RL為基礎開發的系統應該會不斷地更新才對。

**RL的案例**

**ETF自動交易系統**

我們以國立政治大學應用數學系陳⾮霆先生106年發布卷積深度Ｑ-學習之ETF自動交易系統研究為例。







**自學走路的機器人**

南加州大學(University of Southern California) Valero Lab的研究人員建造了一個簡單的機械腿，利用以MATLAB®編寫的強化學習演算法，讓它自己教導自己學會如何移動。



Valero Lab的新機械肢。圖片提供：USC。

透過運動蹣跚(motor babbling)與強化學習的結合，系統會試圖進行隨機的動作，並透過執行這些動作產生的結果來學習其動態的屬性。

研究團隊一開始先讓系統進行隨機的運動，或運動蹣跚(motor babble)。只要每一次系統正確地執行了指定任務在跑步機上向前移動，研究人員就會給予系統獎賞。

這個經由學習產生的演算法被稱為G2P(general to particular，從一般到特定的縮寫)。系統透過運動蹣跚建立了自身動態的一般性理解，接著從每一次的經驗或G2P學習，最終開發出達到特定任務的行為。

**總結:**

雖然強化學習RL被視為大幅進化的機器學習，使用時需注意以下三點：

1.樣本效率(sample-efficient)不佳

要達到可接受的表現，需要有大量的訓練。在相對簡單的應用，也可能會花費幾小時、甚至是幾天的訓練的時間。

AlphaGo就是經過了好幾天沒有間斷、幾百萬次比賽的訓練，等同人類幾千年知識的累積。

2.問題的正確設定不易

可能會需要很多的設計決策、相當次數的疊代來達到正確的行為。這些決策包含了選擇合適的類神經網路架構、調整參數、以及規劃獎賞訊號等。

3.訓練過的深度神經網路策略是一個黑箱

網路內部的結構(通常由幾百萬個參數構成)可能複雜到幾乎難以了解及解釋、以及去評估它所做出的決定，需要大量及耐心去驗證成效。

過