

金門大學資訊工程系



人工智能與神經網路

(還有深度學習的進展)

陳鍾誠

2018年4月20日

人工智慧

- 想讓電腦做有智慧的事情
- 這些事情通常是人才會做的...

而且

- 常常和人體器官有密切關係 . . .

像是

- 影像辨識 – 眼睛
- 語音辨識 – 耳朵
- 機器翻譯 – 嘴巴、耳朵、手
- 機器人 – 眼耳手腳

人工智慧的方法

- 1、比對法
- 2、推理法
- 3、搜尋法
- 4、統計法
- 5、優化法

【筆記】

S代表解空間

s代表解空間中的其中一個解

令 $f(s) \rightarrow R$

意即分數函數 f 將其中一個解映射到一個實數

優化即

把解空間 S 中的所有解全部探索一遍

找到最好的那一個

Über
die
Optimierung
von
Funktionen

比對法

- 紀錄問題與答案配對後，
直接從表格內查出來

像是 Eliza 交談系統



看起來好像很神奇

- 其實用的就只是比對法而已 ...

```
var qaList = [  
    { Q: "謝謝", A: "不客氣!" },  
    { Q: "對不起 | 抱歉 | 不好意思", A: "別說抱歉 !|別客氣，儘管說 !" },  
    { Q: "可否 | 可不可以", A: "你確定想*?" },  
    { Q: "我想", A: "你為何想*?" },  
    { Q: "我要", A: "你為何要*?" },  
    { Q: "你是", A: "你認為我是*?" },  
    { Q: "認為 | 以為", A: "為何說*?" },  
    { Q: "感覺", A: "常有這種感覺嗎?" },  
    { Q: "為何不", A: "你希望我*!" },  
    { Q: "是否", A: "為何想知道是否*?" },  
    { Q: "不能", A: "為何不能*?|你試過了嗎?|或許你現在能*了呢?" },  
    { Q: "我是", A: "你好，久仰久仰!" },  
    { Q: "甚麼 | 什麼 | 何時 | 誰 | 哪裡 | 如何 | 為何 | 因何", A: "為何這樣問?|為何你對這問題有興趣?|你認為答案是甚麼呢?|你認為如何呢?|你常問這類問題嗎?|這真的是你想知道的嗎?|為何不問問別人?|你曾有過類似的問題嗎?|你問這問題的原因是甚麼呢?" },  
    { Q: "原因", A: "這是真正的原因嗎?|還有其他原因嗎?" },  
    { Q: "理由", A: "這說明了甚麼呢?|還有其他理由嗎?" },  
    { Q: "你好 | 嗨 | 您好", A: "你好，有甚麼問題嗎?" },  
    { Q: "或許", A: "你好像不太確定?" }]
```

推理法

- 撰寫規則後，電腦根據規則推論

專家系統用的就是推理法

哺乳類 \Leftarrow 有毛.
哺乳類 \Leftarrow 泌乳.
鳥類 \Leftarrow 有羽毛.
鳥類 \Leftarrow 會飛 & 生蛋.
食肉類 \Leftarrow 哺乳類 & 吃肉.
食肉類 \Leftarrow 有爪 & 利齒 & 兩眼前視.
有蹄類 \Leftarrow 哺乳類 & 有蹄.
偶蹄類 \Leftarrow 哺乳類 & 反芻.
獵豹 \Leftarrow 哺乳類 & 吃肉 & 斑點.
老虎 \Leftarrow 哺乳類 & 吃肉 & 條紋.
長頸鹿 \Leftarrow 有蹄類 & 長腿 & 斑點.
斑馬 \Leftarrow 有蹄類 & 條紋.
鴕鳥 \Leftarrow 鳥類 & 長腿.

```
facts=[]
?- 有毛
addFact(有毛)
addFact(哺乳類)
facts=[ "有毛", "哺乳類"]
?- 吃肉
addFact(吃肉)
addFact(食肉類)
facts=[ "有毛", "哺乳類", "吃肉", "食肉類"]
?- 條紋
addFact(條紋)
addFact(老虎)
facts=[ "有毛", "哺乳類", "吃肉", "食肉類", "條紋", "老虎"]
?- 
```

可以幫助你進行決策

哺乳類 \Leftarrow 有毛.

哺乳類 \Leftarrow 泌乳.

鳥類 \Leftarrow 有羽毛.

鳥類 \Leftarrow 會飛 & 生蛋.

食肉類 \Leftarrow 哺乳類 & 吃肉.

食肉類 \Leftarrow 有爪 & 利齒 & 兩眼前視.

有蹄類 \Leftarrow 哺乳類 & 有蹄.

偶蹄類 \Leftarrow 哺乳類 & 反芻.

獵豹 \Leftarrow 哺乳類 & 吃肉 & 斑點.

老虎 \Leftarrow 哺乳類 & 吃肉 & 條紋.

長頸鹿 \Leftarrow 有蹄類 & 長腿 & 斑點.

斑馬 \Leftarrow 有蹄類 & 條紋.

鶲鳥 \Leftarrow 鳥類 & 長腿.

?- 有毛

哺乳類 \Leftarrow 有毛 .

?- 吃肉

食肉類 \Leftarrow 哺乳類 & 吃肉 .

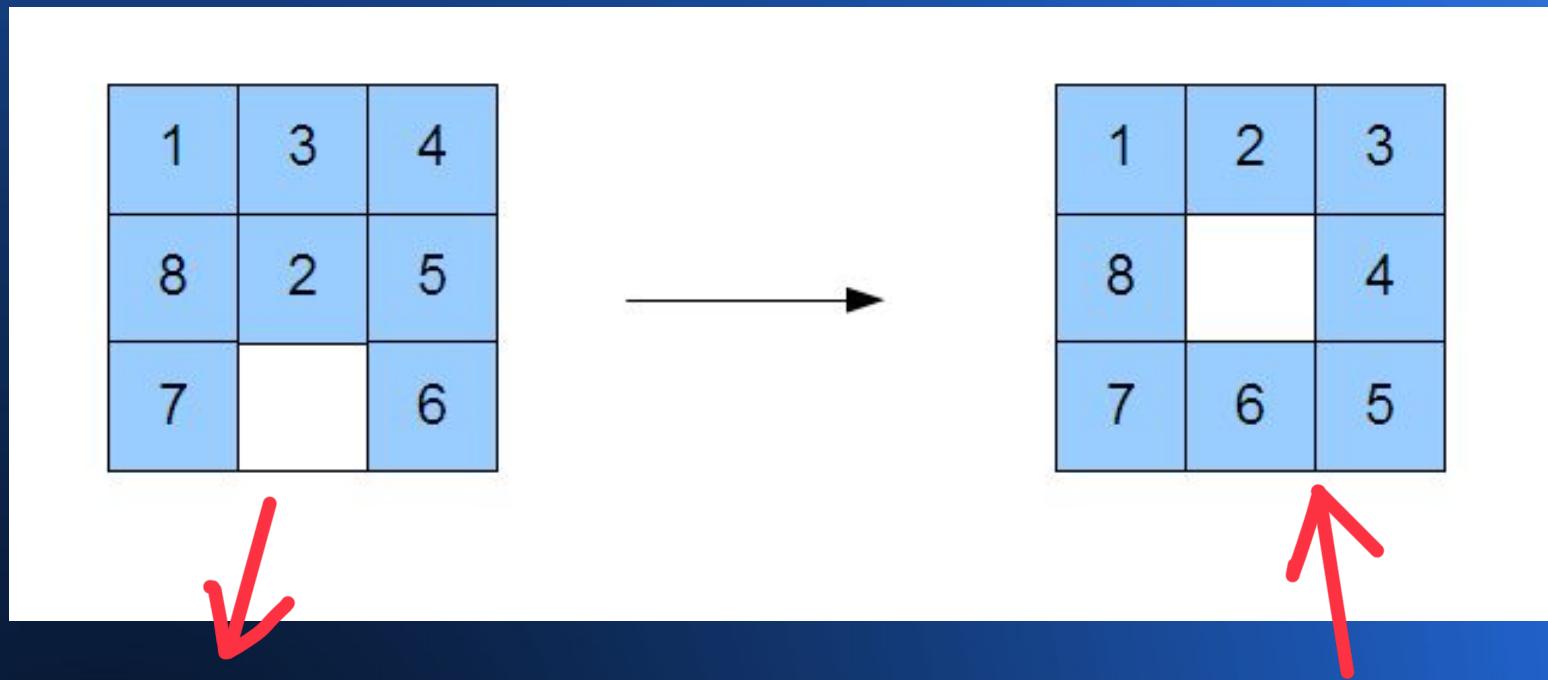
? - 條紋

老虎 \Leftarrow 哺乳類 & 吃肉 & 條紋 .

搜尋法

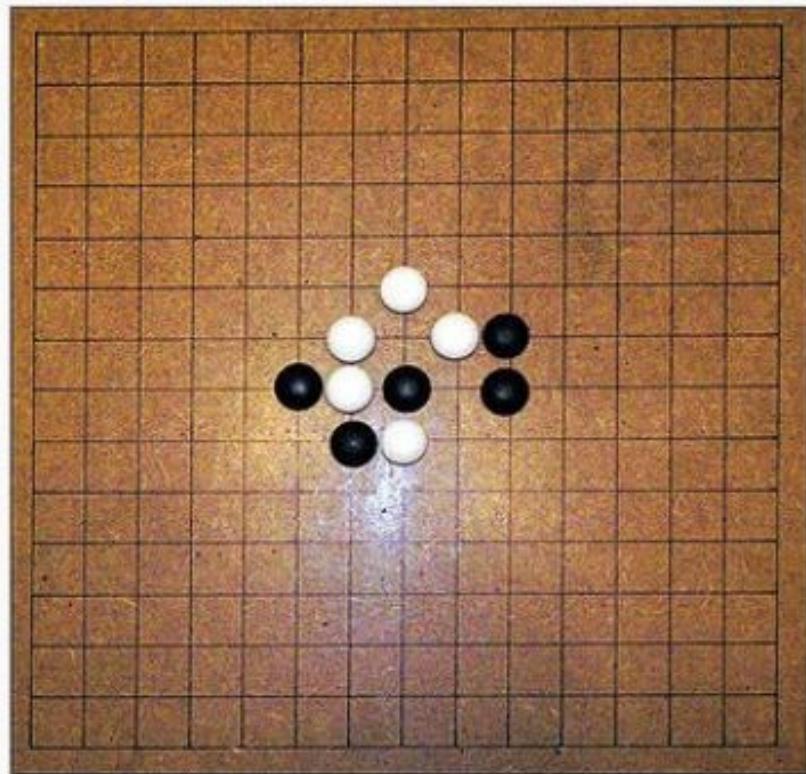
- 對所有可能的結果進行系統式的列舉，然後看看有沒有答案！

像是拼圖

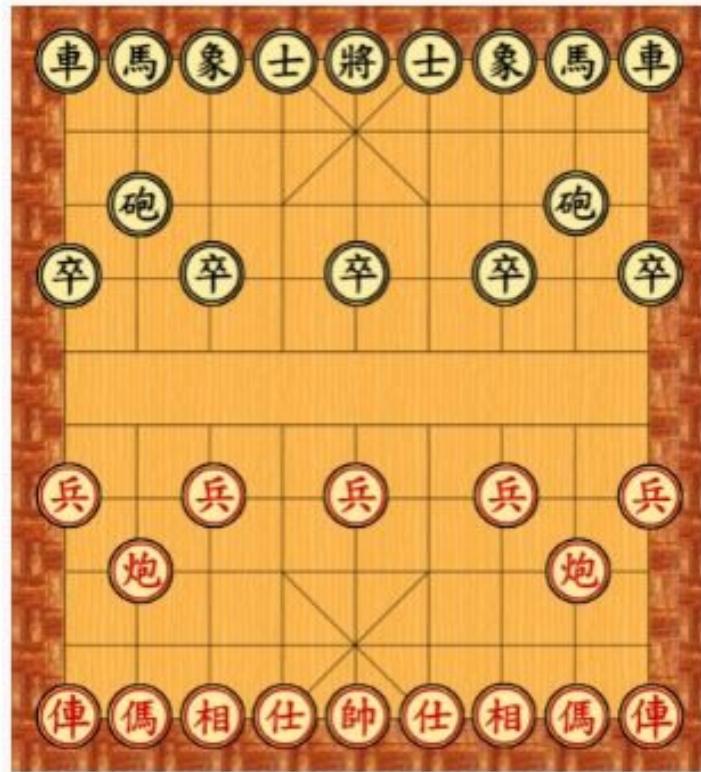


1,3,4	1,3,4	1,3,4	1,3,0	1,0,3	1,2,3
8,2,5	=> 8,2,5	=> 8,2,0	=> 8,2,4	=> 8,2,4	=> 8,0,4
7,0,6	7,6,0	7,6,5	7,6,5	7,6,5	7,6,5

還有下棋



(a) 五子棋 / 圍棋



(b) 象棋

【筆記】

以五子棋為例，擬定一個計分表如下：

attackScores = [0, 3, 10, 100, 500]

攻擊分數：分別代表連成1子、2子、3子、4子、5子的分數

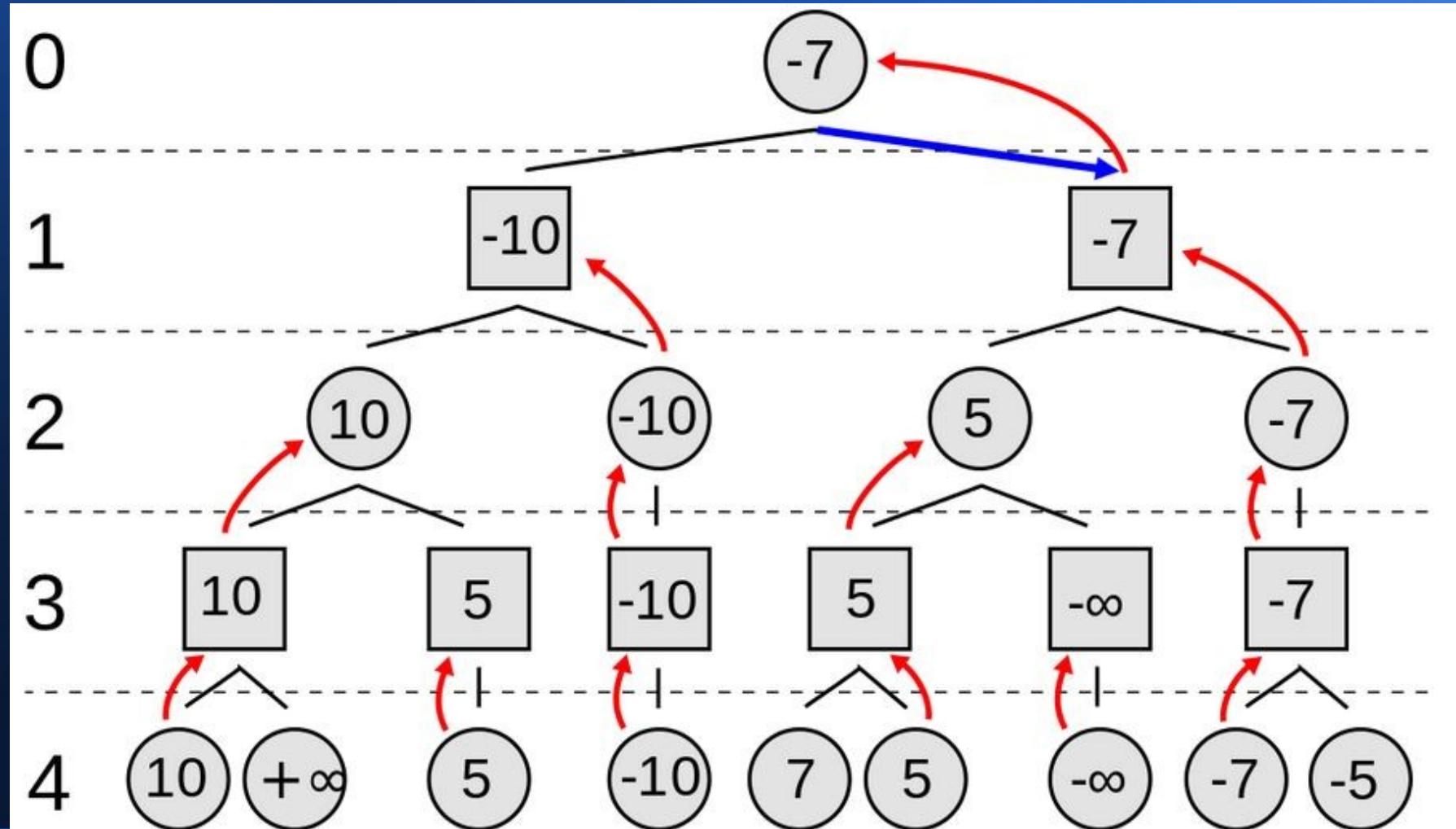
guardScores = [0, 2, 9, 25, 90, 400]

防守分數：分別代表阻斷對手連成1子、2子、3子、4子、5子的分數

演算法：

遍歷棋盤上可以落子的每一個位置，計算在此位置的攻擊分數加防守分數總和，取分數最高的位置落子。

用的通常就是搜尋法



統計法

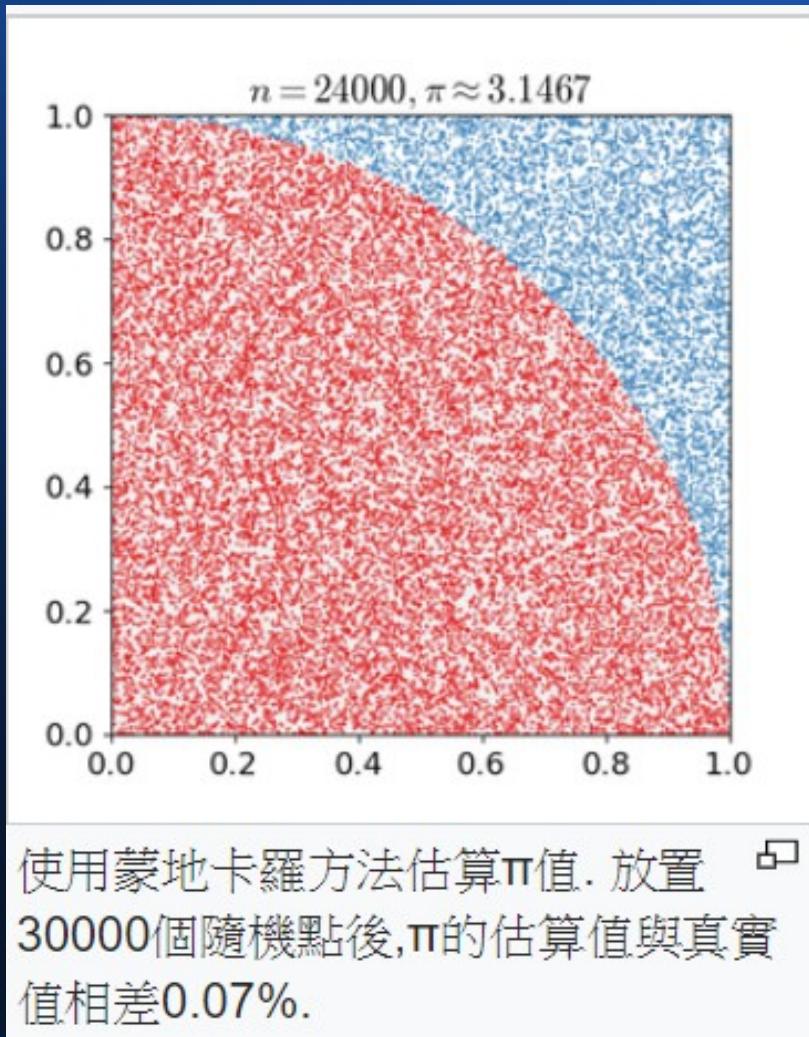
- 根據統計資訊，找出最有可能的那個答案
- 有可能會結合推理搜尋...

統計的目的

- 通常是尋找機率最大的那個解答

$$\begin{aligned}\sum_z P(Z = z|x, h) L(x, Z = z|h) &= \sum_z \frac{P(x, Z = z, h)}{P(x, h)} \log P(x, Z = z|h) \\ &= \frac{1}{P(x, h)} \sum_z P(x, Z = z, h) \log P(x, Z = z|h) \\ &= \frac{1}{P(x, h)} H(x, Z|h)\end{aligned}$$

例如我們可以用統計法算圓周率



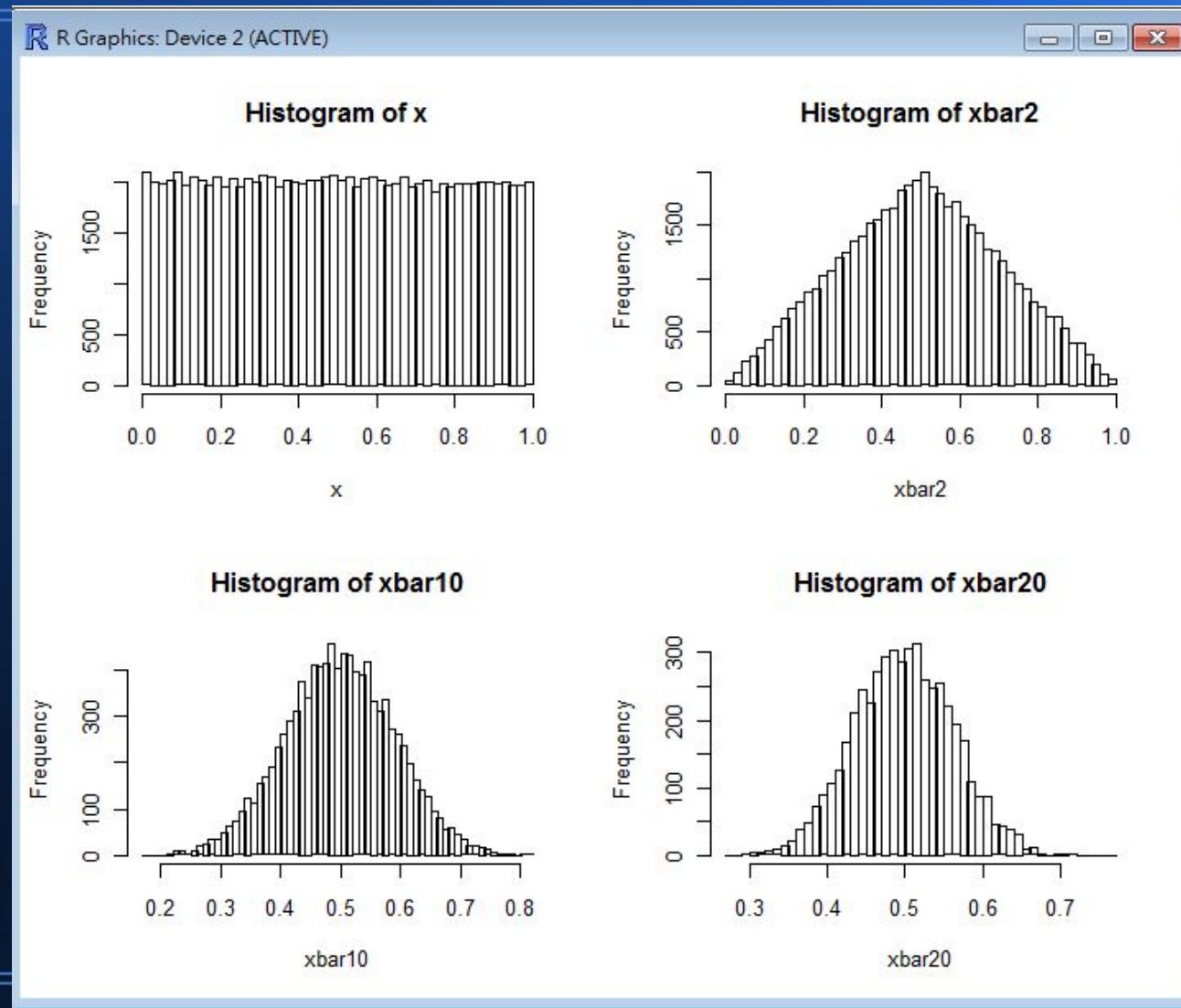
如果樣本是用亂數產生的，
通常會被稱為是蒙地卡羅法 ...

我們可以輕易用電腦亂數

驗證中央極限定理

【筆記】

→ 對獨立的樣本取平均值會越來越趨向常態分布



或用貝氏定理

- 進行機率式推論 . . .

貝氏定理是關於隨機事件A和B的條件機率的一則定理。

$$P(A|B) = \frac{P(A) \times P(B|A)}{P(B)}$$

其中 $P(A|B)$ 是在B發生的情況下A發生的可能性。

很多語音辨識

- 還有機器翻譯系統
- 都是用統計法

最後一個方法

- 是所謂的優化法 . . .

優化法

- 對每個可能的解答，都會給一個分數，尋找分數最好的那個解答！

優化法涵蓋面很廣

可以說

- 一切皆優化 . . .

只要

- 有辦法判斷解答的好壞
- 就可以使用優化法
- 因為我們總希望選擇最好的答案！

不管是

- 電腦下棋
- 機器翻譯
- 影像辨識
- 語音辨識

【筆記】

只要能夠解決通用的優化問題，就幾乎解決了人工智慧的所有問題。但所有的優化問題都是NP-complete (non-deterministic polynomial，非決定式的多項式集合)，有時候甚至比NP-complete更難。

我們都會希望

- 電腦給我們最好的答案！

【筆記】

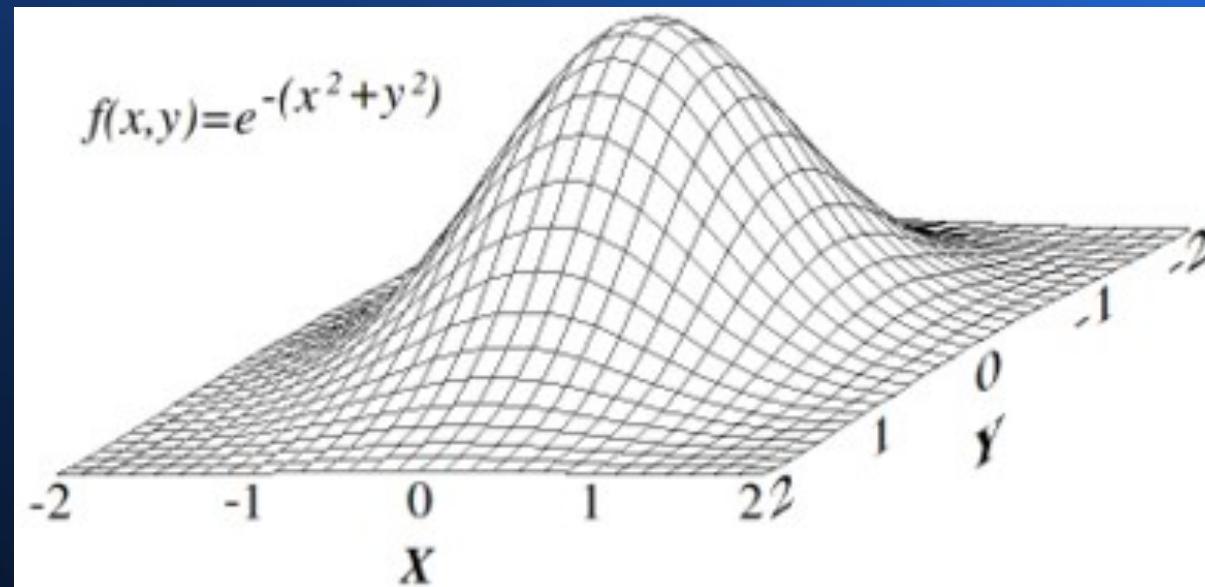
真實世界中的問題，常有一個很大的解空間，我們要去搜尋出最好的結果，但是解空間太大了，不可能全部取完，所以歷史上出現了許多找最好答案的算法。

所以前輩們設計了下列方法

- 爬山演算法
- 遺傳演算法
- 期望 - 最大化算法 (EM)
- 神經網路 ...

爬山演算法

- 是一種很簡單的優化方法



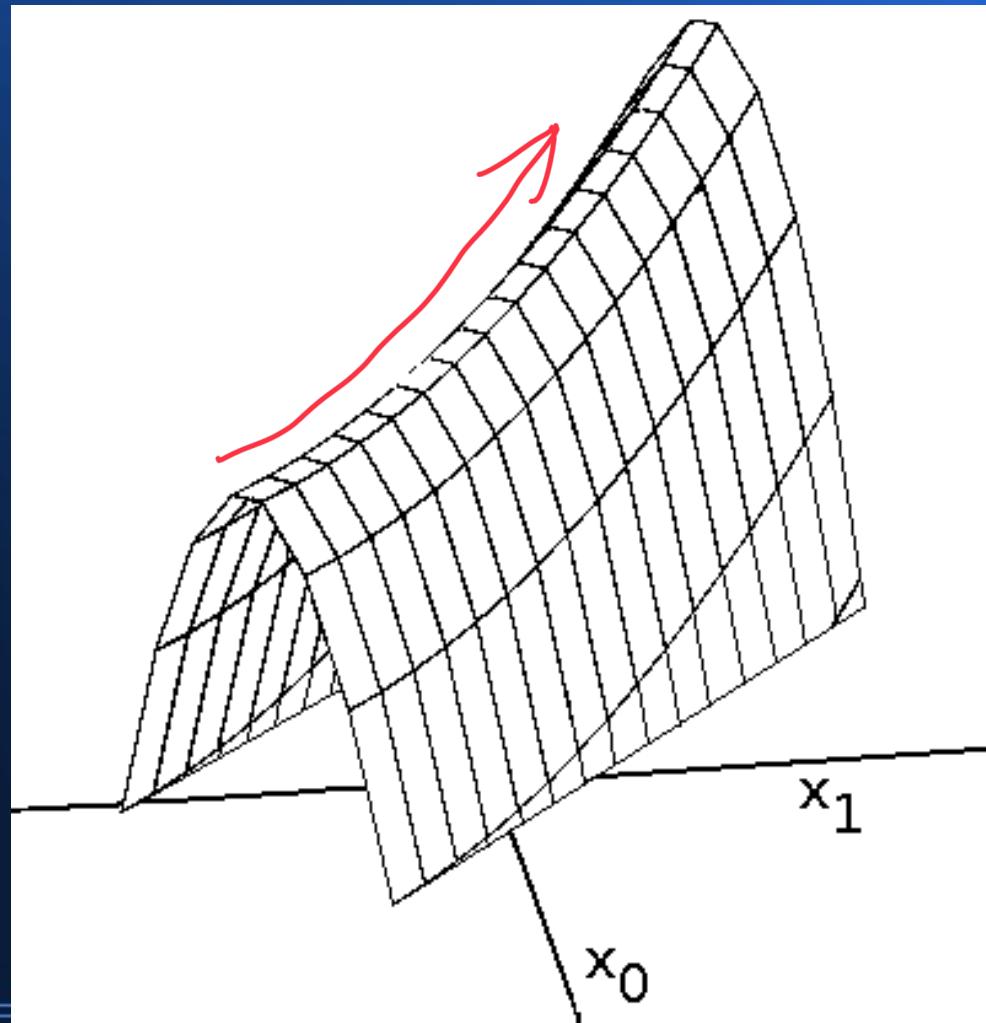
方法是

- 看到附近有更高的就往那邊爬

```
Algorithm HillClimbing(f, x)
    x = 隨意設定一個解。高度
    while (x 有鄰居  $x'$  比 x 更高)
        x =  $x'$ ;
    end
    return x;
end
```

如果是山脊

- 那就會沿著陵線爬...



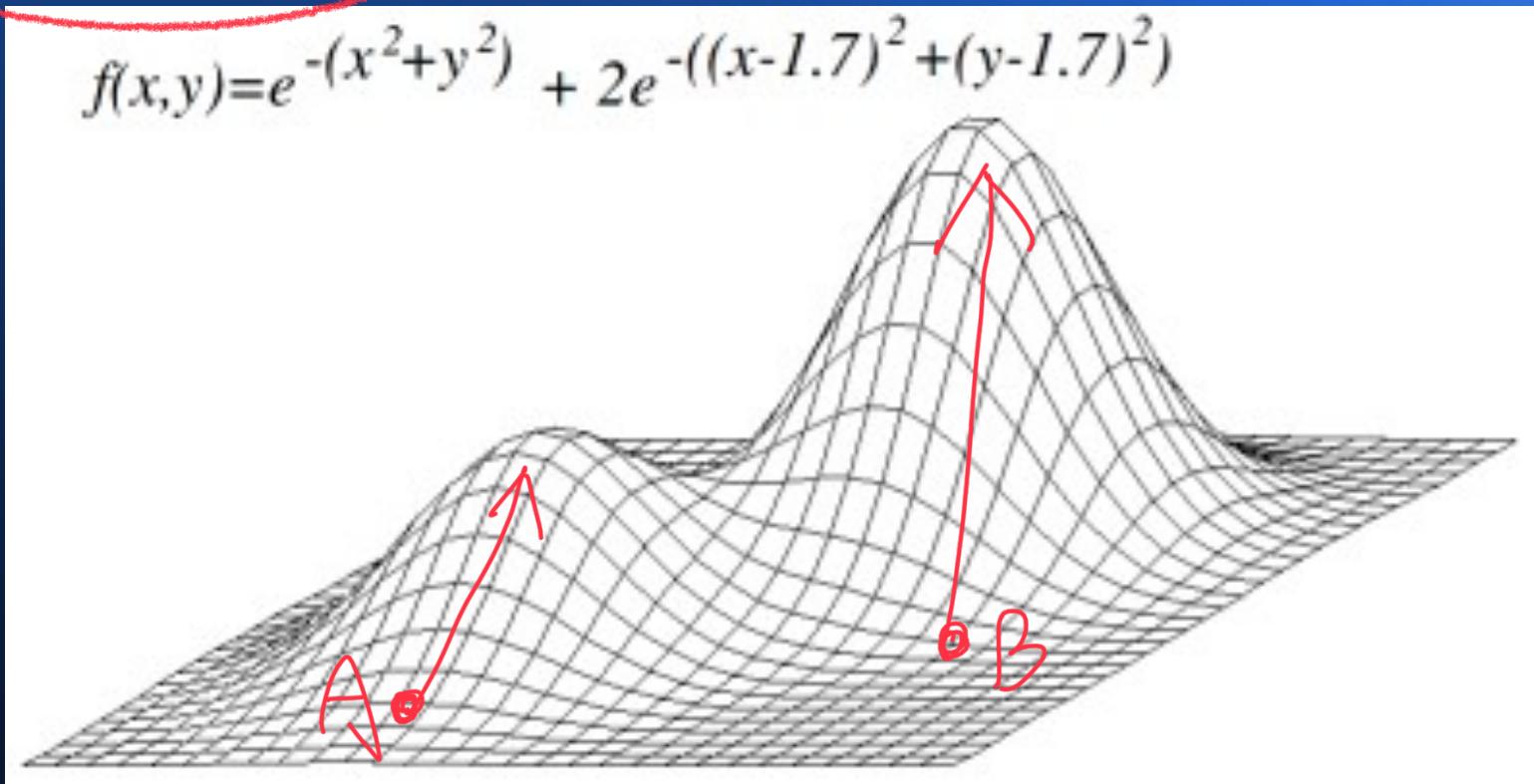
但是如果有很多山峰

爬山演算法不一定會到最高的山峰（會停在任何山頂）

【筆記】Notes

要看起始點在哪

最近的那個相對高點



但是

- 對於很多優化問題
- 爬山演算法常常就很夠用了！

神經網路

- 也是一種優化法

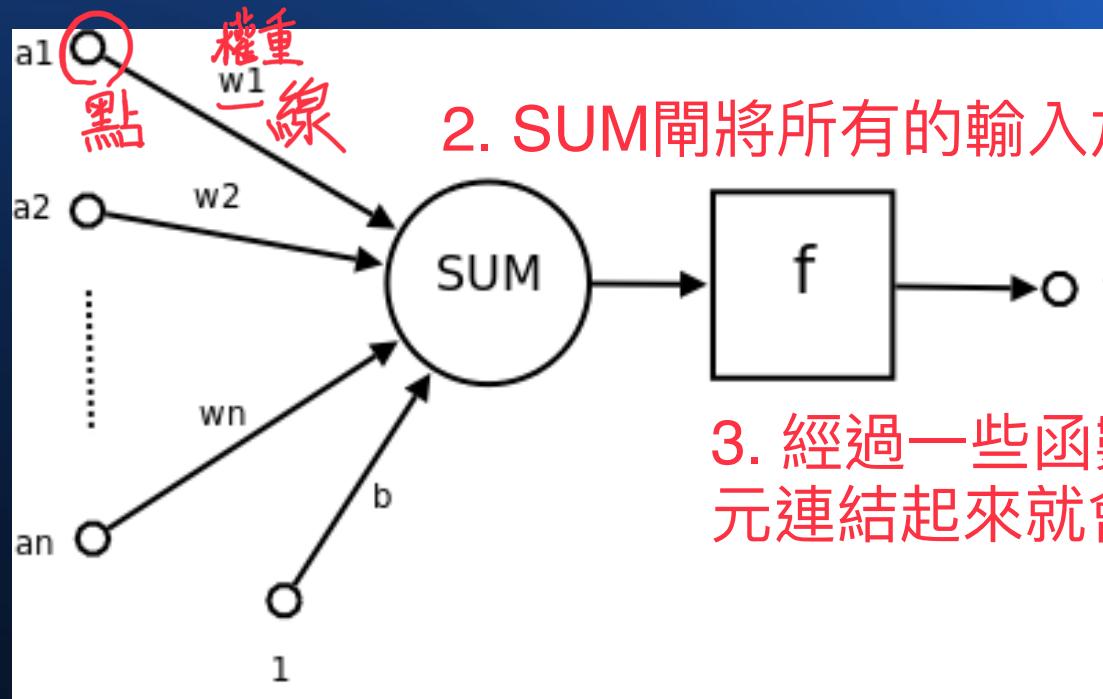
模仿人腦神經網路結構

【筆記】

將人類的神經簡化再簡化之後就剩下圖形理論。

- 但簡化了很多，只留下《點 + 線 + 閘》 . . .

1. 將點的刺激強度乘以權重作為輸入



2. SUM閘將所有的輸入加起來得到加總

3. 經過一些函數，把很多的神經元連結起來就會形成神經網路。

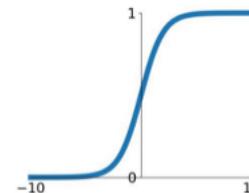
每個神經元

- 都會用開關函數來控制

Activation Functions

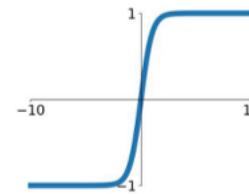
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



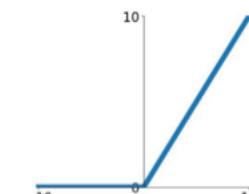
tanh

$$\tanh(x)$$



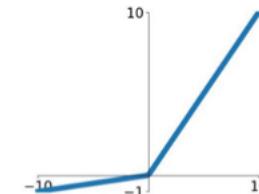
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

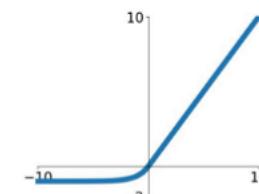


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

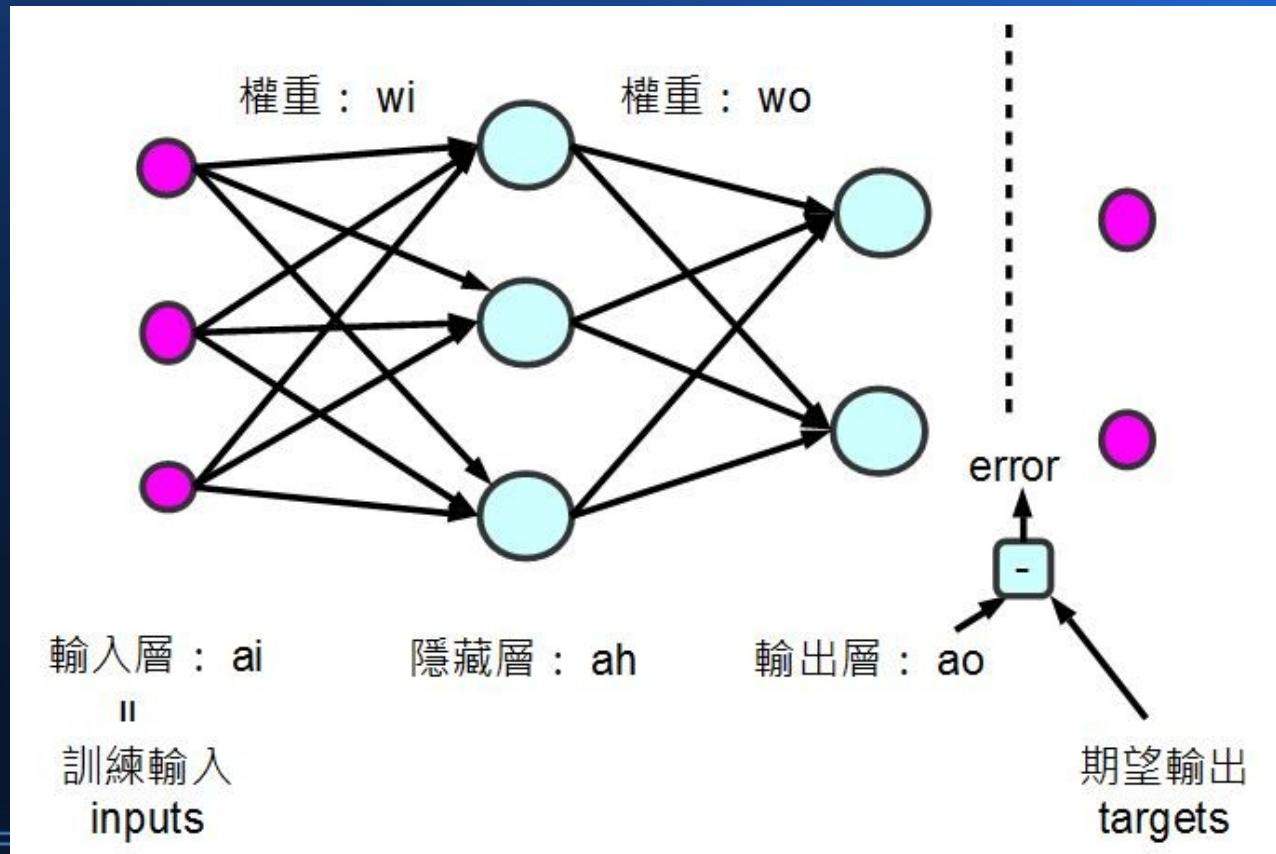
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



傳統神經網路

- 以《多層感知器》為主要模型



採用《梯度下降法 + 反傳遞演算法》

☆ 當神經元、參數很多的時候會變得很慢

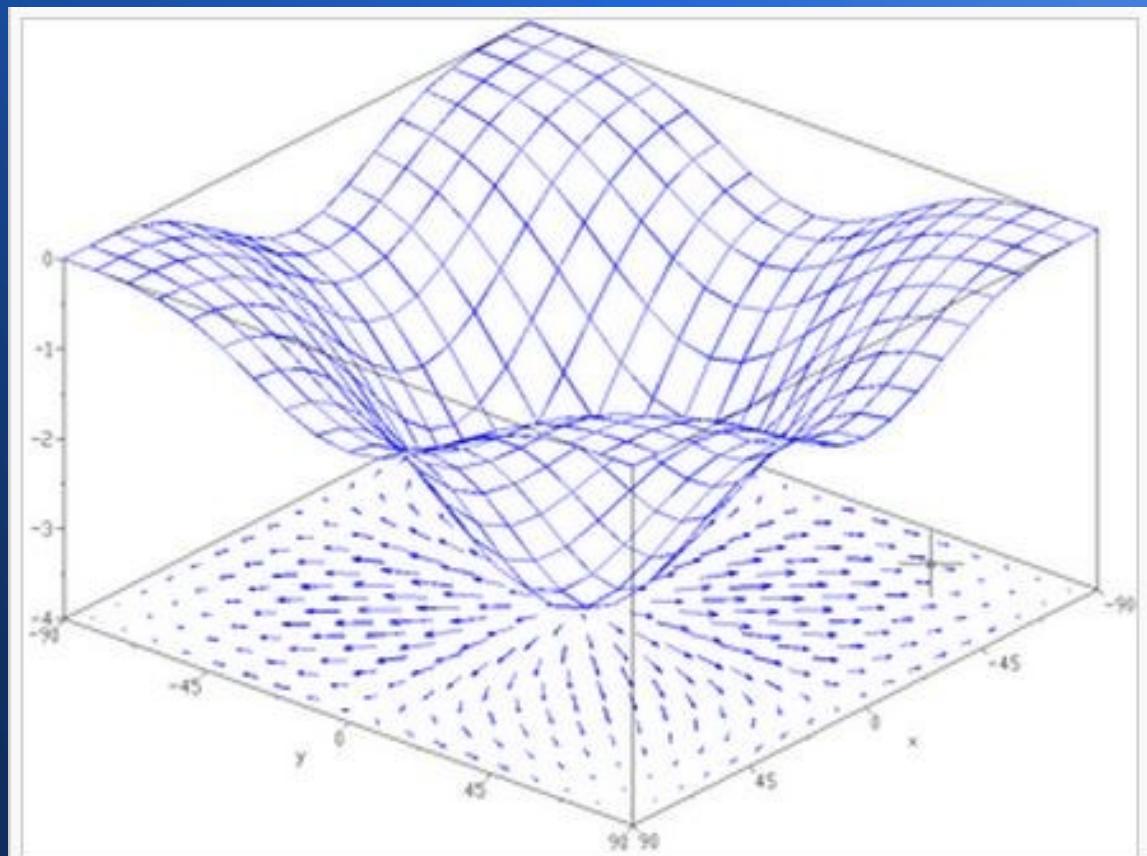
☆ 較快速的、但會用到很深的微積分



所謂的梯度

- 就是斜率最大的那個方向

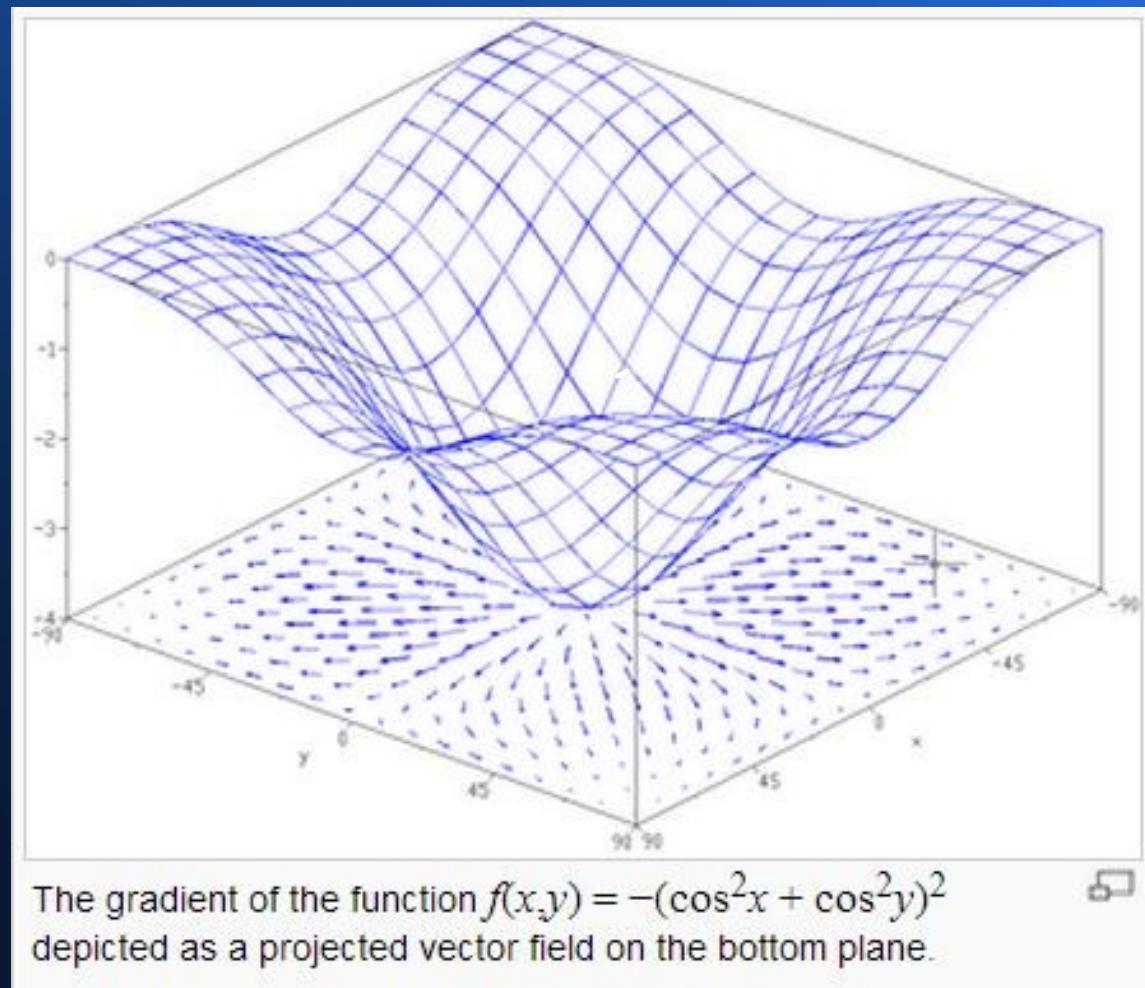
$$\nabla f = \frac{\partial f}{\partial x_1} \vec{e}_1 + \cdots + \frac{\partial f}{\partial x_n} \vec{e}_n$$



The gradient of the function $f(x,y) = -(\cos^2 x + \cos^2 y)^2$ depicted as a projected vector field on the bottom plane.

梯度下降法

就是利用微積分計算梯度，每次都朝最陡的方向下坡



★ 往逆梯度的方向走，就會走到山谷
(相反則為山峰)



★ 使用梯度下降法

→ 雖較爬山演算法準確

→ 但解決的問題必須要能算梯度

→ 連續可微函數才能算梯度

→ 不容易將一個問題變成連續可微函數

★ 所以有時爬山演算法更好用

而反傳遞算法

- 則是套用微積分裡的鏈鎖規則

$$\frac{\partial f(q, z)}{\partial x} = \frac{\partial q(x, y)}{\partial x} \frac{\partial f(q, z)}{\partial q}$$

- 從後一層反推前一層的梯度
這樣才能調整前一層的參數

梯度下降法

- 仍然是在優化
- 目標是尋找能量的最低點

$$Y^* = \operatorname{argmin}_{Y \in \mathcal{Y}} E(Y, X).$$

能量通常就是錯誤率

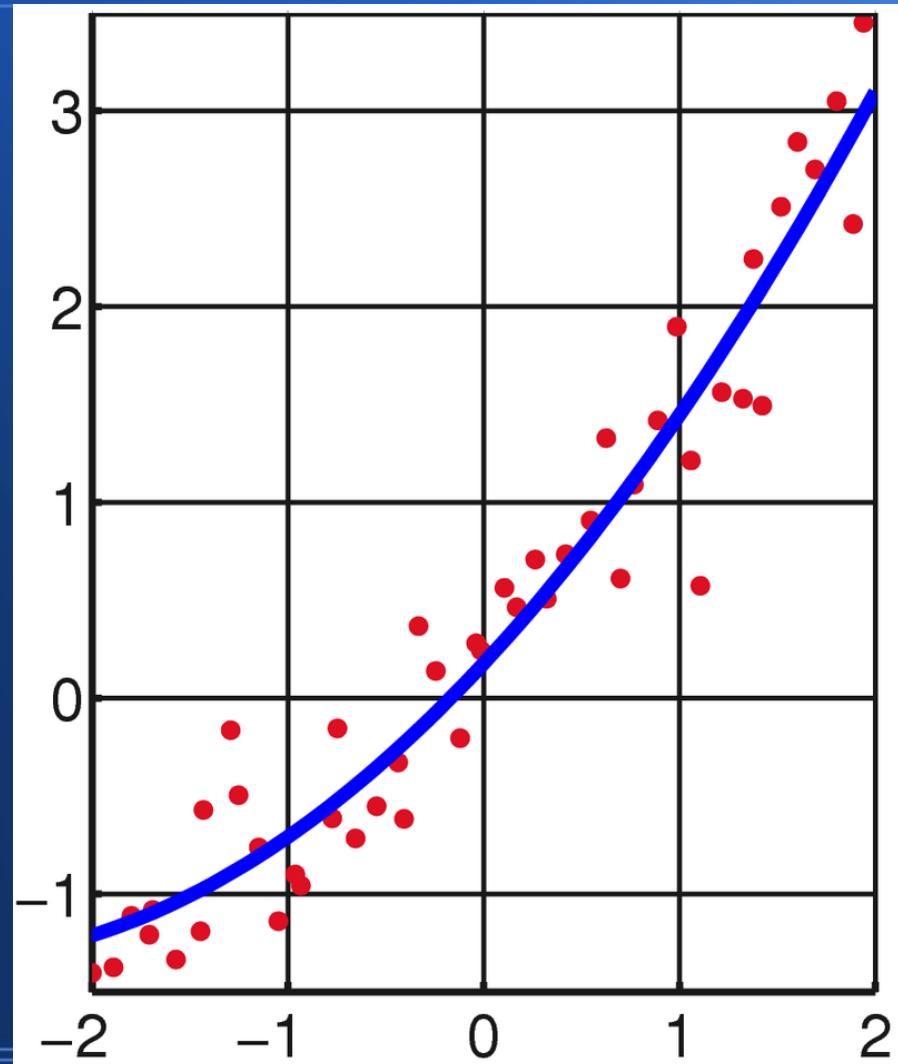
以神經網路而言，梯度下降法用於找跟正確答案之間的差異。

★有兩種可能性：

1. 最後一層使用回歸的方式 → 找最小平方法
2. 最後一層如果使用softmax → 猜正確答案

統計和線性代數裏

- 有個最小平方法
- 其實就是一種
優化問題的解法



神經網路

- 也可以透過反傳遞算法
- 尋找最符合的線型 ...

新一代的神經網路

- 由於層數較多較深，因此被稱為《深度學習》！

《深度學習一詞的由來》

新舊神經網路有模型上的差異

舊的：多層感知器

新的：

- 卷積神經網路 CNN
- 循環神經網路 RNN

從這兩個新的模型開始，

人們發現還可以把層數拉的更深，

把卷積、relu、pool等等串接在一起後發現效果很好，
可以再做出更多改良的模型。



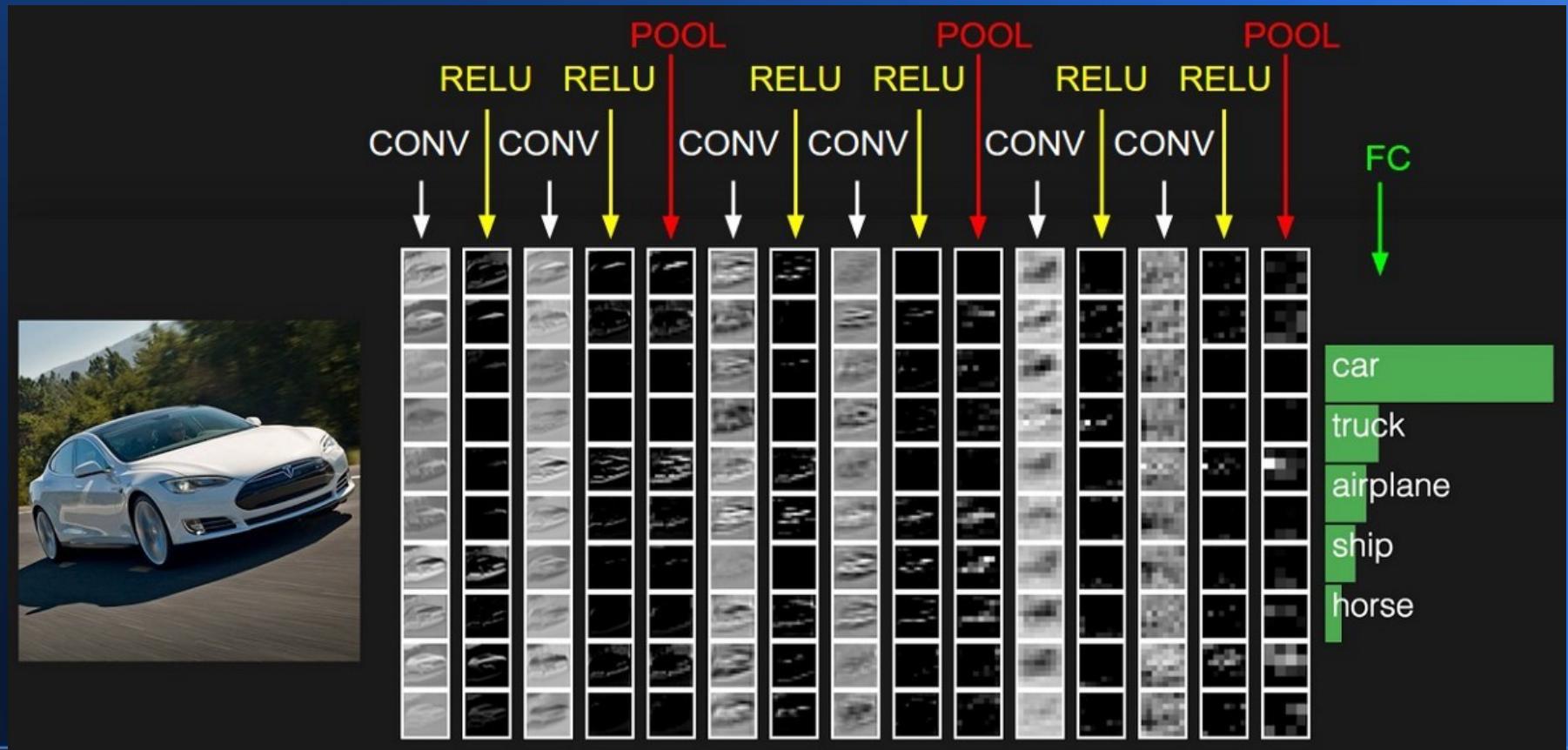
深度學習的神經網路

- 除了多層感知器之外，還加入了：
 - 卷積神經網路 CNN
 - 循環神經網路 RNN, LSTM
(長短期記憶網路)
 - 生成對抗網路 GAN
 - 偽造者
 - 辨識者(一直偽造到辨識者辨識不出來)
 - 強化學習機制 Reinforcement Learning

卷積神經網路 CNN

- 通常被用來《辨認影像》

像是這樣



其中的卷積層 CONV

- 是一種遮罩函數，檢查有沒有符合遮罩的點

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input Image



0	0	1
1	0	0
0	1	1

Feature
Detector

=

0				

Feature Map

像是有些遮罩可以取出邊緣

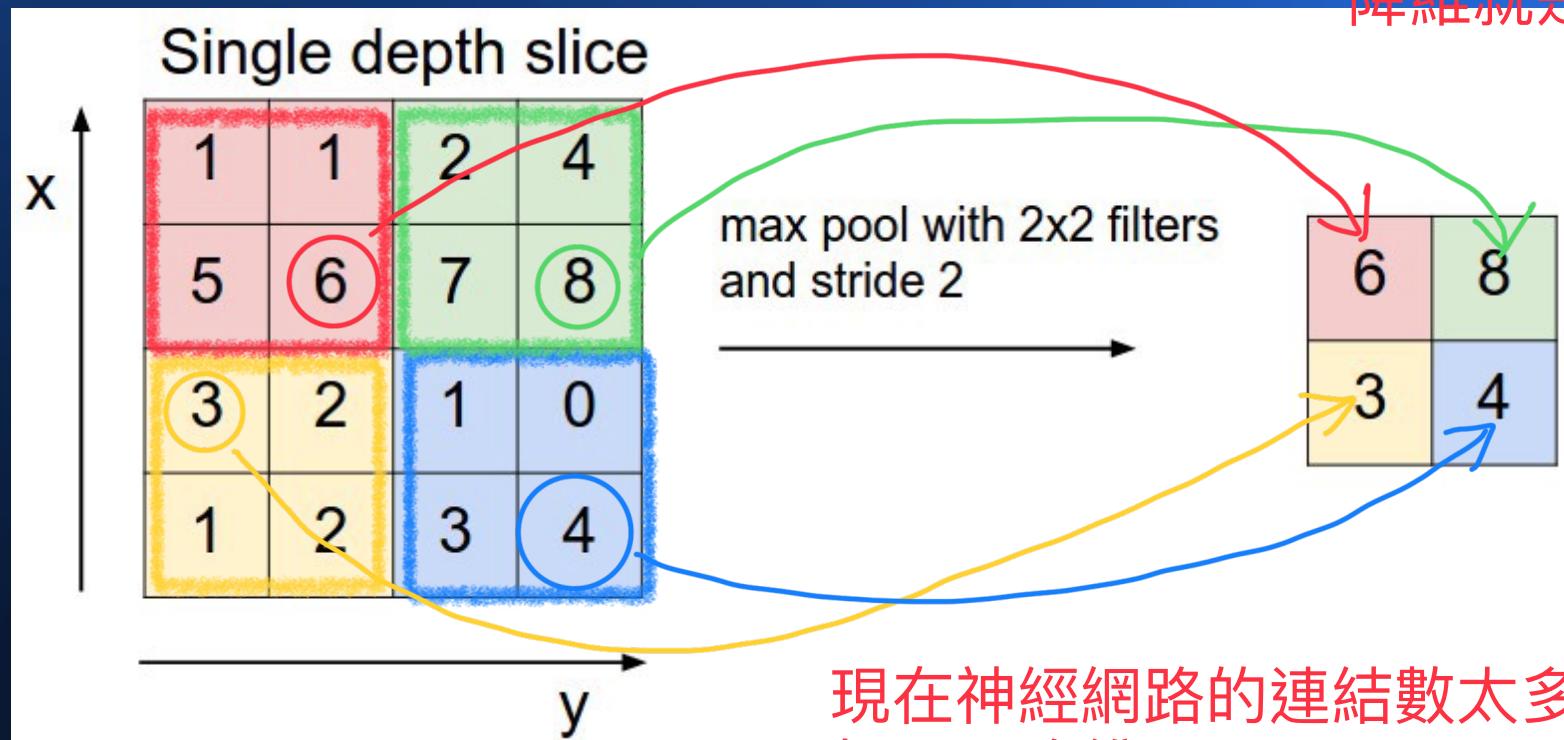


有些遮罩可以區分深淺



而池化層 Pool

- 其實就只是取最大值，可以用來降低維度
降維就是池化



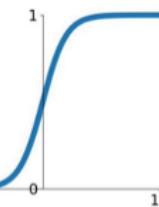
卷積網路中的 ReLU 層

則是一種開關，讓高於某個門檻的訊號通過

Activation Functions

Sigmoid

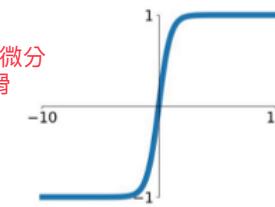
$$\sigma(x) = \frac{1}{1+e^{-x}}$$



tanh

$$\tanh(x)$$

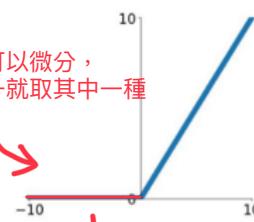
為了要可以微分
所以平滑



ReLU

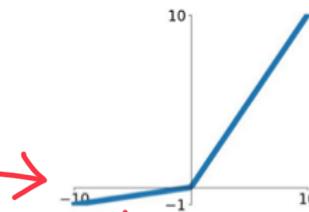
$$\max(0, x)$$

轉折點還是可以微分，
兩邊斜率不一就取其中一種



Leaky ReLU

$$\max(0.1x, x)$$



增加斜率
避免神經網路不再學習

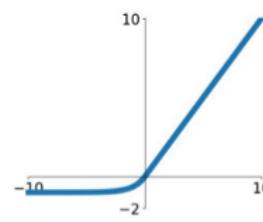
Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

演變
 \cup
åö

ELU

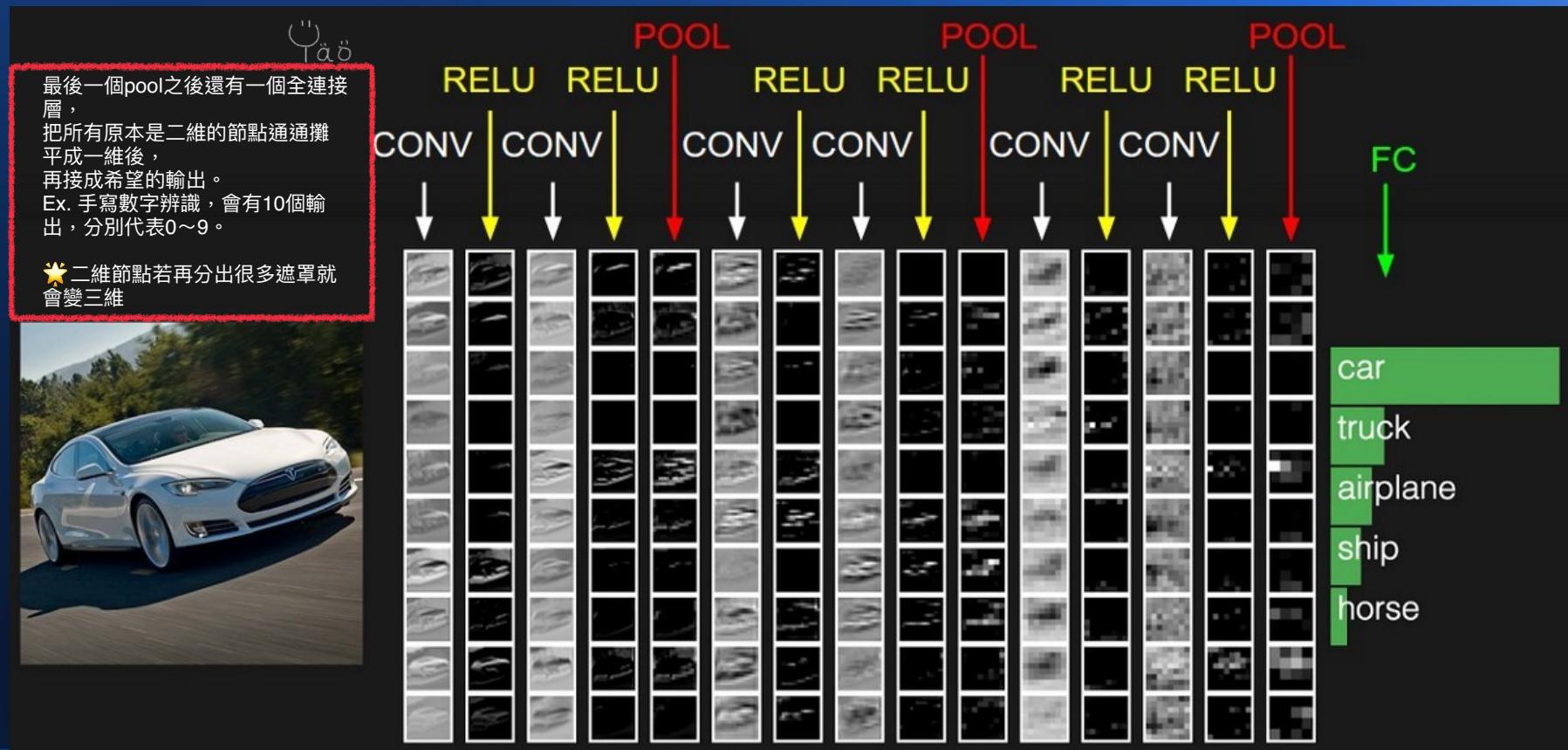
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



平的地方有梯度消失的問題，神經網路停止學習

最後加上傳統的全連接層

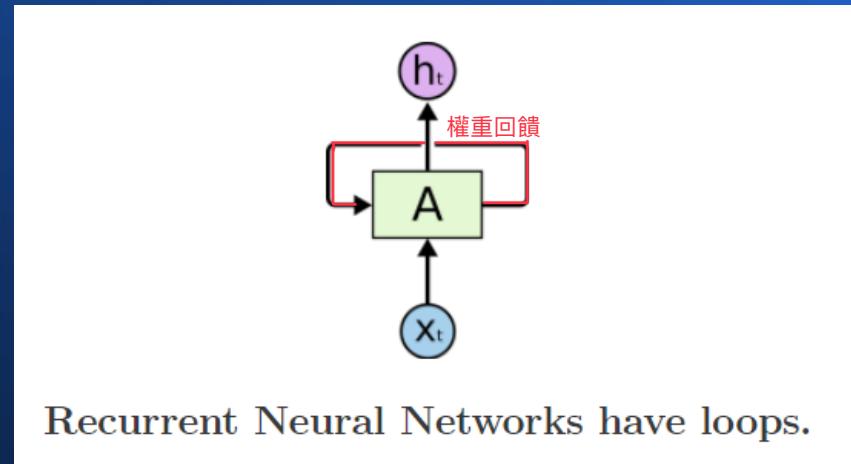
卷積網路就可以辨識影像達到頗高的正確率



循環神經網路 RNN

- 最常被用來處理語言
- 像是機器翻譯系統

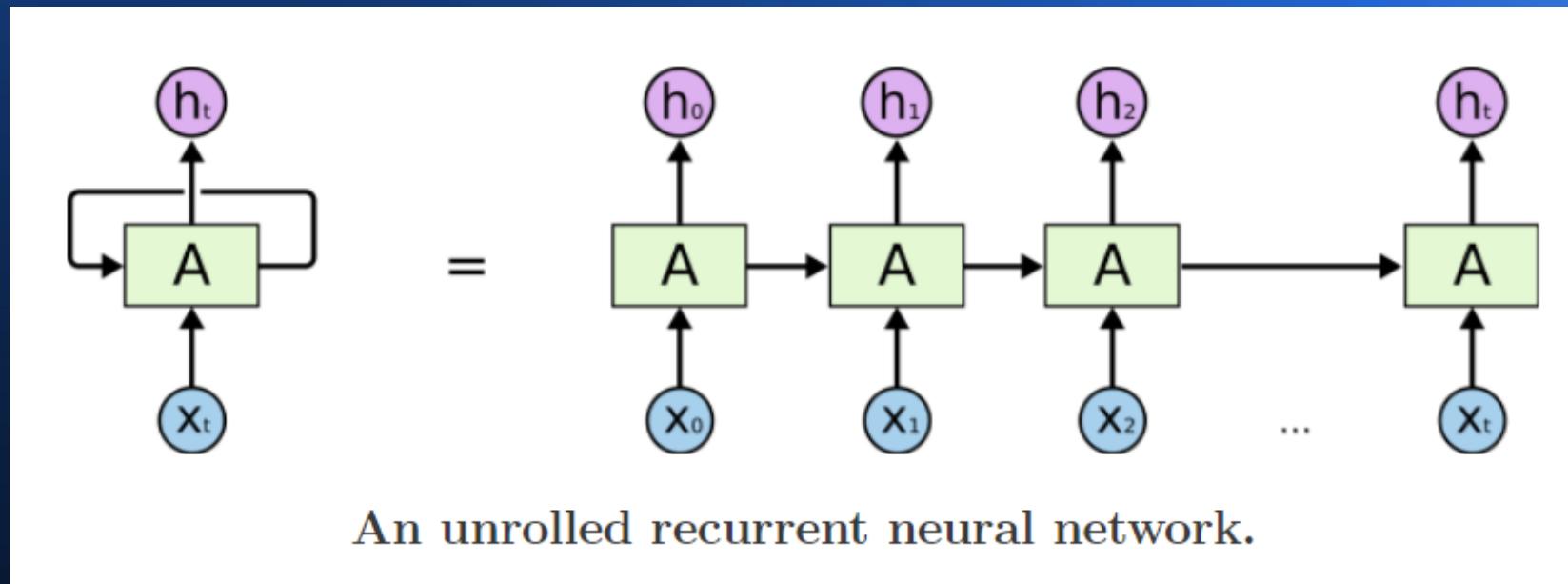
RNN 的神經網路有循環



- 可以用來儲存一些《狀態》
- 於是相同的輸入，不見得會有相同的輸出
- 因為和目前狀態有關

如果把網路根據時間展開

- 就可以看出售這種概念的用途



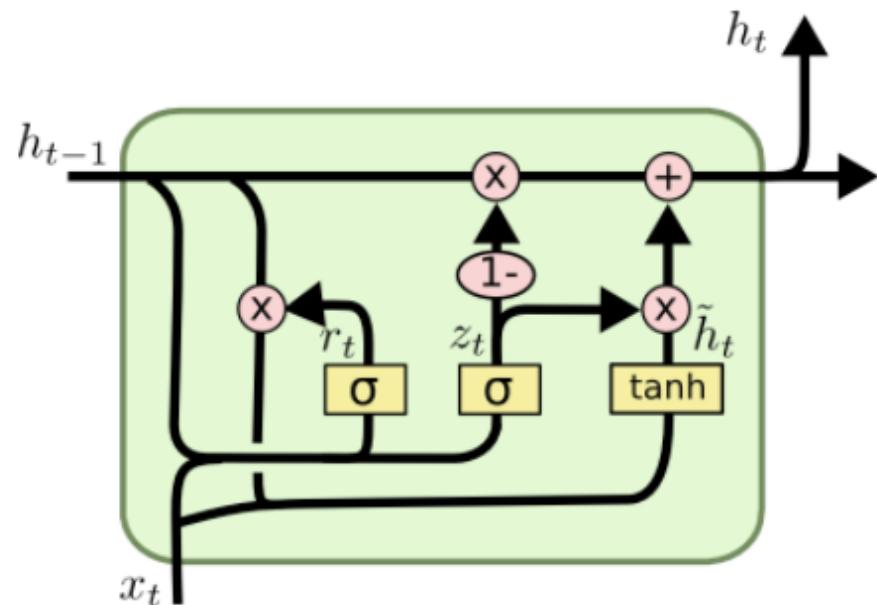
基本上就是可以處理時間序列

但是簡單的 RNN 不穩定

- 只能記住短期的事情 . . .

所以有人發展出長短期記憶網路

- 簡稱為 LSTM



$$z_t = \sigma (W_z \cdot [h_{t-1}, x_t])$$

$$r_t = \sigma (W_r \cdot [h_{t-1}, x_t])$$

$$\tilde{h}_t = \tanh (W \cdot [r_t * h_{t-1}, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

循環神經網路

- 很常被用來學習語言的語法
- 學完之後該網路就會《亂講話》
但是卻講得《人模人樣》的...

如果看完莎士比亞之後 就會產生像莎士比亞風格的文章

VIOLA:

Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:

O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.

看完一堆 Tex 格式的論文

就會產生一堆很像 Tex 格式的論文

```
\begin{proof}
We may assume that $\mathcal{I}$ is an abelian sheaf on $\mathcal{C}$.
\item Given a morphism $\Delta : \mathcal{F} \rightarrow \mathcal{I}$
is an injective and let $\mathfrak{q}$ be an abelian sheaf on $X$.
Let $\mathcal{F}$ be a fibered complex. Let $\mathcal{F}$ be a category.
\begin{enumerate}
\item \hyperref[setain-construction-phantom]{Lemma}
\label{lemma-characterize-quasi-finite}
Let $\mathcal{F}$ be an abelian quasi-coherent sheaf on $\mathcal{C}$.
Let $\mathcal{F}$ be a coherent $\mathcal{O}_X$-module. Then
$\mathcal{F}$ is an abelian catenary over $\mathcal{C}$.
\item The following are equivalent
\begin{enumerate}
\item $\mathcal{F}$ is an $\mathcal{O}_X$-module.
\end{enumerate}
\end{enumerate}
\end{proof}
```

而且還可以用 LaTeX 轉換成 PDF

看起來比我寫的論文還要好 . . .

For $\bigoplus_{n=1,\dots,m}$ where $\mathcal{L}_{m\bullet} = 0$, hence we can find a closed subset \mathcal{H} in \mathcal{H} and any sets \mathcal{F} on X , U is a closed immersion of S , then $U \rightarrow T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \text{Spec}(R) = U \times_X U \times_X U$$

and the comparicoly in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \rightarrow V$. Consider the maps M along the set of points \mathcal{Sch}_{fppf} and $U \rightarrow U$ is the fibre category of S in U in Section ?? and the fact that any U' affine, see Morphisms, Lemma ???. Hence we obtain a scheme S and any open subset $W \subset U$ in $\mathcal{Sh}(G)$ such that $\text{Spec}(R') \rightarrow S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S . We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \rightarrow \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\text{GL}_{S'}(x'/S'')$ and we win. \square

To prove study we see that $\mathcal{F}|_U$ is a covering of X' , and \mathcal{T}_i is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and \mathcal{F}_p exists and let \mathcal{F}_i be a presheaf of \mathcal{O}_X -modules on \mathcal{C} as a \mathcal{F} -module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{\mathcal{M}}^\bullet = \mathcal{I}^\bullet \otimes_{\text{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1} \mathcal{F}$$

is a unique morphism of algebraic stacks. Note that

$$\text{Arrows} = (\mathcal{Sch}/S)_{fppf}^{\text{opp}}, (\mathcal{Sch}/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longrightarrow (U, \text{Spec}(A))$$

is an open subset of X . Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S .

Proof. See discussion of sheaves of sets. \square

The result for prove any open covering follows from the less of Example ???. It may replace S by $X_{\text{spaces},\text{étale}}$ which gives an open subspace of X and T equal to S_{Zar} , see Descent, Lemma ???. Namely, by Lemma ?? we see that R is geometrically regular over S .

Lemma 0.1. Assume (3) and (3) by the construction in the description.

Suppose $X = \lim |X|$ (by the formal open covering X and a single map $\underline{\text{Proj}}_X(\mathcal{A}) = \text{Spec}(B)$ over U compatible with the complex

$$\text{Set}(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X,\mathcal{O}_X}).$$

When in this case of to show that $\mathcal{Q} \rightarrow \mathcal{C}_{Z/X}$ is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are catenary. If T is surjective we may assume that T is connected with residue fields of S . Moreover there exists a closed subspace $Z \subset X$ of X where U in X' is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem

(1) f is locally of finite type. Since $S = \text{Spec}(R)$ and $Y = \text{Spec}(R)$.

Proof. This is form all sheaves of sheaves on X . But given a scheme U and a surjective étale morphism $U \rightarrow X$. Let $U \cap U = \coprod_{i=1,\dots,n} U_i$ be the scheme X over S at the schemes $X_i \rightarrow X$ and $U = \lim_i X_i$. \square

The following lemma surjective restrocomposes of this implies that $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X},\dots,0}$.

Lemma 0.2. Let X be a locally Noetherian scheme over S , $E = \mathcal{F}_{X/S}$. Set $\mathcal{I} = \mathcal{J}_1 \subset \mathcal{I}'_n$. Since $\mathcal{I}'_n \subset \mathcal{I}^n$ are nonzero over $i_0 \leq p$ is a subset of $\mathcal{J}_{n,0} \circ \bar{A}_2$ works.

Lemma 0.3. In Situation ???. Hence we may assume $q' = 0$.

Proof. We will use the property we see that p is the next functor (??). On the other hand, by Lemma ?? we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

where K is an F -algebra where δ_{n+1} is a scheme over S . \square

接著看完 Linux 作業系統的原始碼

結果就可以寫出一堆看來很像樣的 C 語言程式了

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
    int error;
    if (fd == MARN_EPT) {
        /*
         * The kernel blank will coeld it to userspace.
         */
        if (ss->segment < mem_total)
            unblock_graph_and_set_blocked();
        else
            ret = 1;
        goto bail;
    }
    segaddr = in_SB(in.addr);
    selector = seg / 16;
    setup_works = true;
    for (i = 0; i < blocks; i++) {
        seq = buf[i++];
        bpf = bd->bd.next + i * search;
        if (fd) {
            current = blocked;
```

現在的 Google 翻譯系統

- 使用的就是 LSTM 類型的循環神經網路技術

The screenshot shows the Google Translate interface with two tabs: '中文' (Chinese) and '英文' (English). The English tab is selected. The input text is: "The Unreasonable Effectiveness of Recurrent Neural Networks" dated "May 21, 2015". The output text is: "遞歸神經網絡的不合理有效性" dated "2015年5月21日". The translated text is: "回歸神經網絡（RNN）有一些神奇的東西。我還記得當我訓練我的第一個經常使用圖像標題的網絡時。在幾十分鐘的訓練中，我的第一個嬰兒模型（帶有相當隨意選擇的超參數）開始產生非常好看的圖像描述，這些圖像處於有意義的邊緣。有時，你的模型的簡單性與你從中得到的結果的質量之間的比例超過了你的期望值，這是其中的一種。當時令人震驚的結果是，普遍的看法是RNNs應該很難訓練（我有更多的經驗實際上達到了相反的結論）。大約快一年：我一直在訓練RNN，我已經目睹了他們的力量和魯棒性，但他們的神奇輸出仍然讓我感到有趣。這篇文章是關於與你分享一些魔術。" Below the translated text are several small icons: a star, a square, a double arrow, and a left arrow.

效果比之前的版本好很多

接著讓我們看一種最新型的神經網路

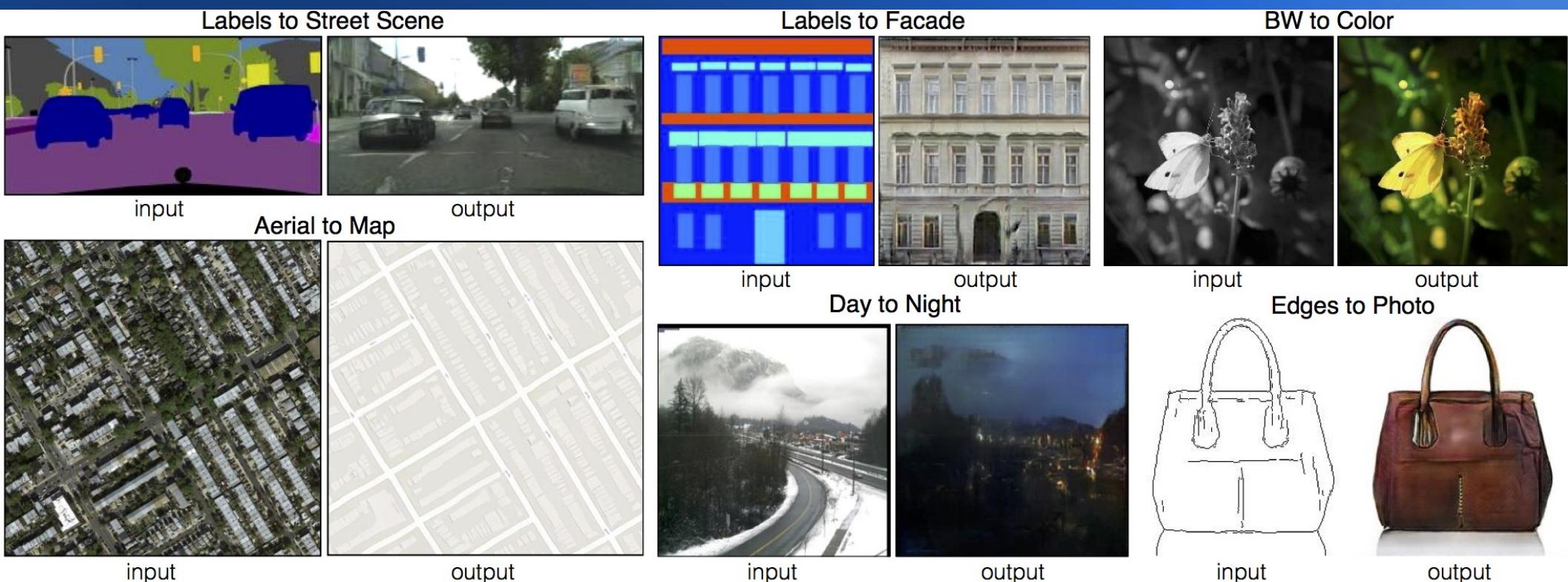
- 稱為《生成對抗網路》
- 簡稱 GAN

生成對抗網路 GAN

- 採用《偽造者與鑑賞家》對抗的方式，讓雙方能夠在對抗過程中能力愈來愈強！

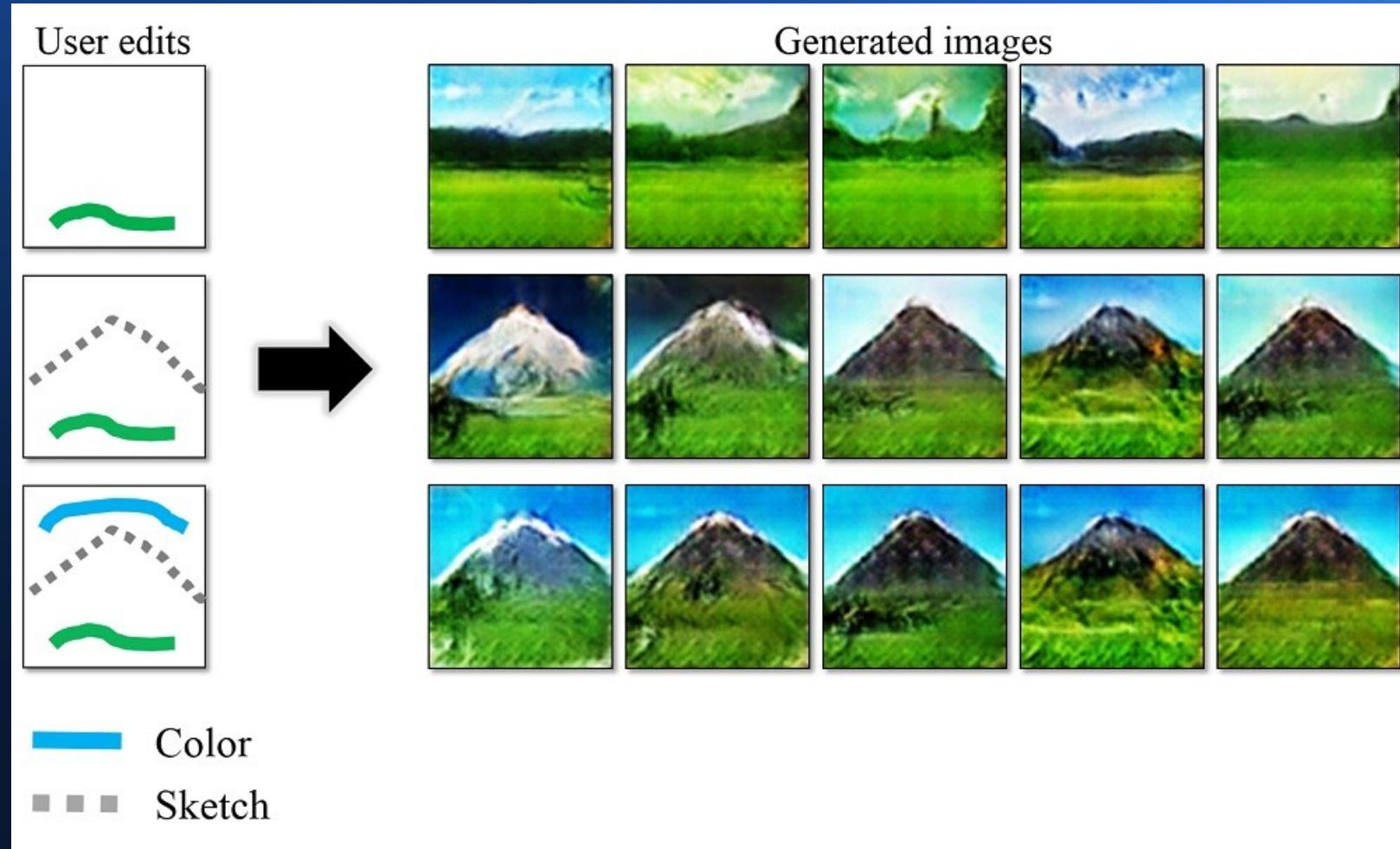
GAN 最擅長《模仿他人的風格》

或者將《素描》轉換成《擬真照片》

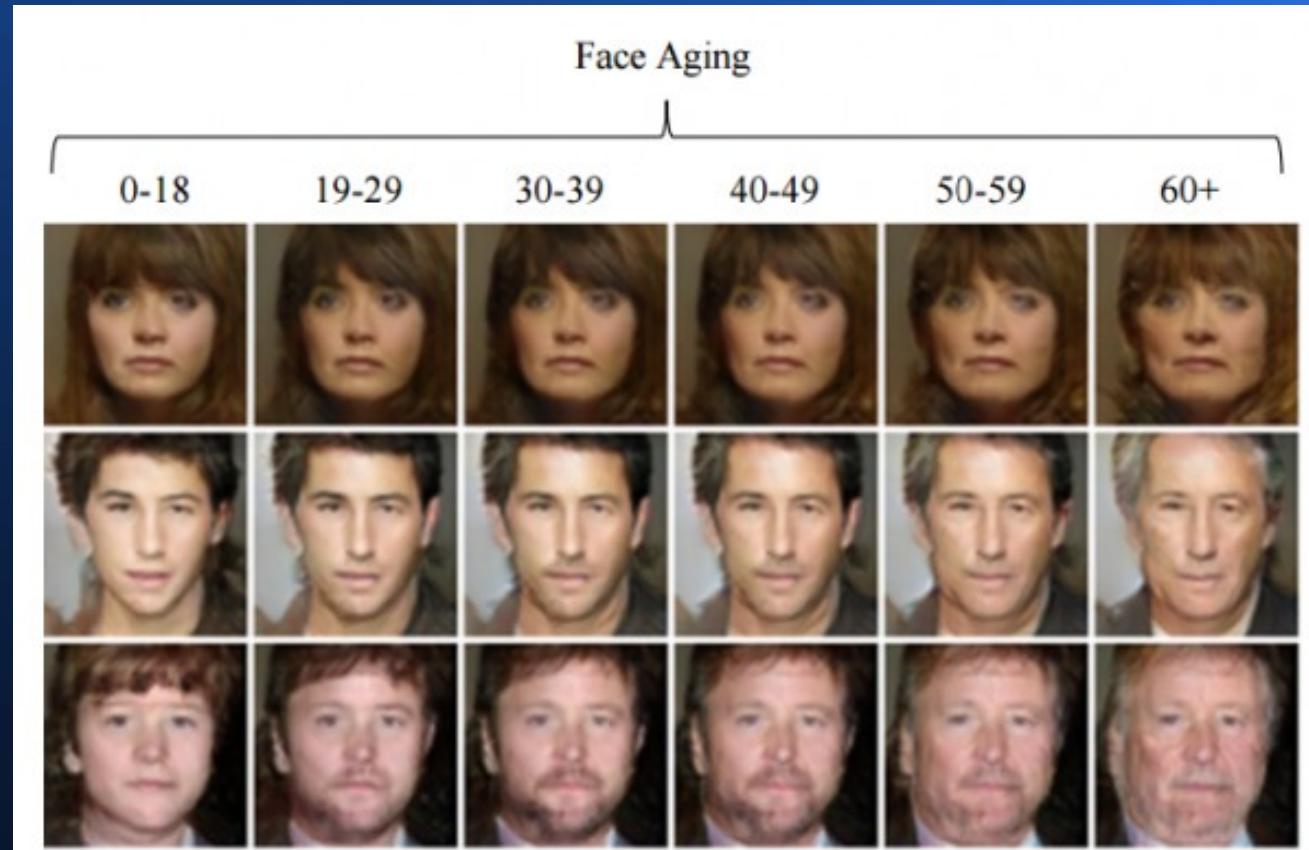


簡單的幾筆畫

也可以被轉換成逼真的風景照



如果讓 GAN 學習人臉



你就可以透過調整參數

Attributes vector file ⓘ
/home/greg/ws/digits/attributes_z_20170202-100209-72af.pkl

Z vector (leave blank for random) ⓘ
6.64713979e-01 9.05301422e-03 9.21479464e-01 7.21585378e-02 3.69947910e-01 1.40126292e-02 1.01437695e-01

Add or remove attributes by filling corresponding box with +1 or -1 (or any other multiplier).

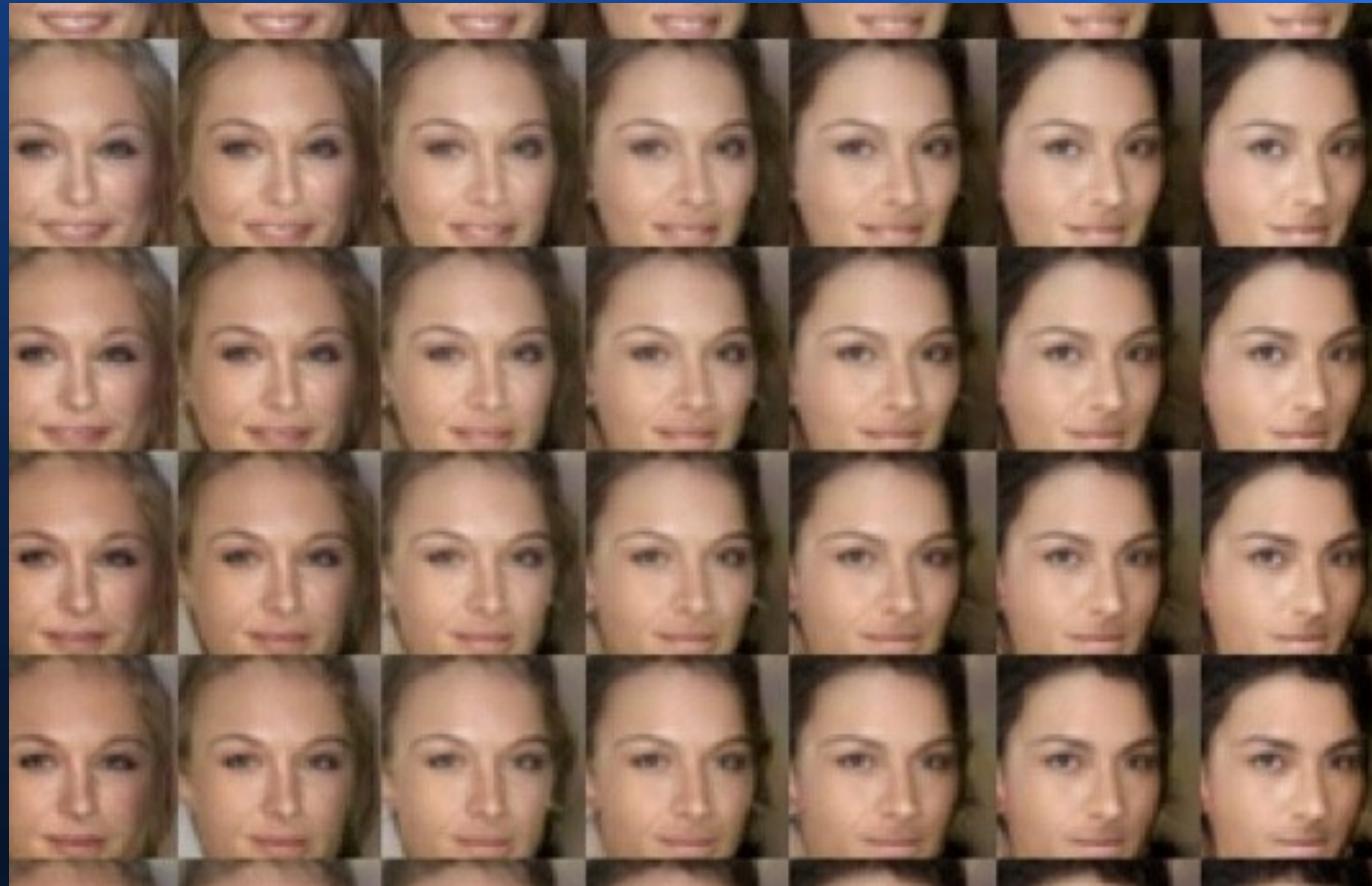
Attributes Params

Bald	Black_Hair	Blond_Hair	Male	Smiling	Wearing_Lipstick	Young
<input type="checkbox"/>						
<input type="checkbox"/>	+1	-1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
+2	<input type="checkbox"/>					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	-1	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	+2	+2				

Add row

Index	Data
1	
2	
3	
4	
5	

得到合成的改造臉



因為 GAN 的網路參數

- 通常都會對應到某些意義
- 像是有沒有鬍子、男或女、年紀大小、頭髮顏色等等

Attributes vector file [?](#)

/home/greg/ws/digits/attributes_z_20170202-100209-72af.pkl

Z vector (leave blank for random) [?](#)

6.64713979e-01 9.05301422e-03 9.21479464e-01 7.21585378e-02 3.69947910e-01 1.40126292e-02 1.01437695e-01

Add or remove attributes by filling corresponding box with +1 or -1 (or any other multiplier).

Attributes Params

Bald	Black_Hair	Blond_Hair	Male	Smiling	Wearing_Lipstick	Young
<input type="checkbox"/>						
<input type="checkbox"/>	+1	-1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
+2	<input type="checkbox"/>					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	-1	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	+2	+2				

Add row

Index	Data
1	
2	
3	
4	
5	

這種技術還可以用來造假

- 所以即使有影片為證都不太能相信



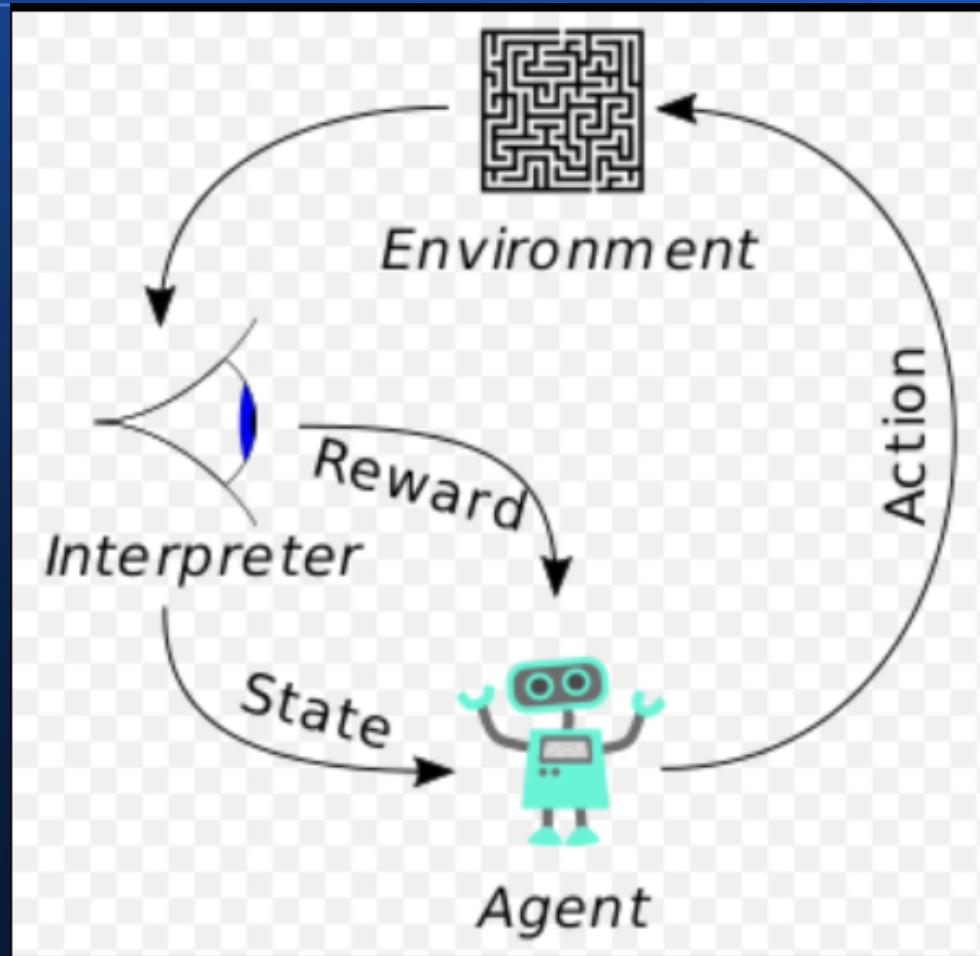
最後讓我們看強化學習機制

強化學習機制

- 常常透過《探索或自我對打》，找到好的策略來進行決策！
- 像是 AlphaGo 圍棋程式就是《機率模型 + 強化學習 + 神經網路》結合得到的結果！

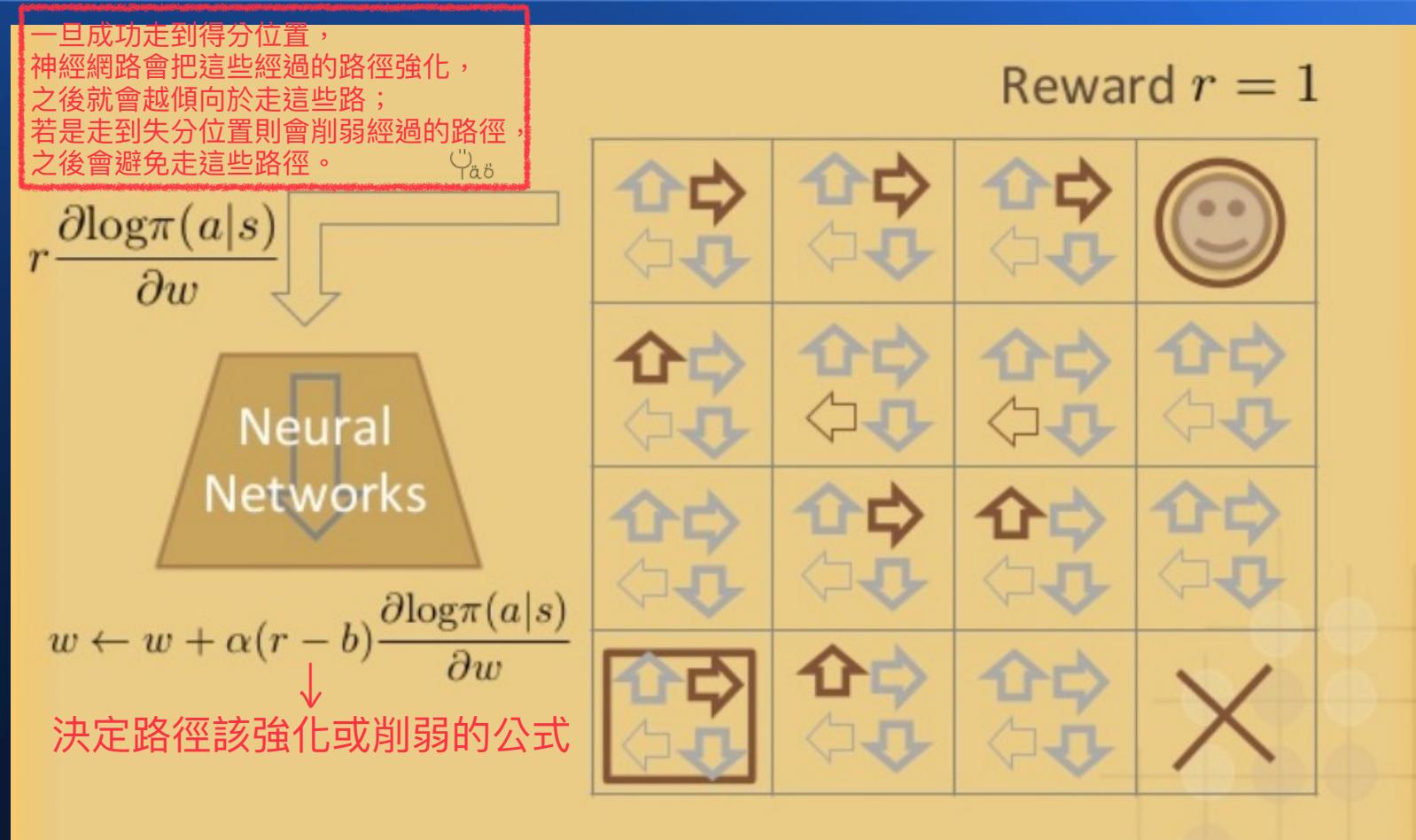
強化學習可以用在機器人上

像是讓掃地機器人學習房間打掃的方式

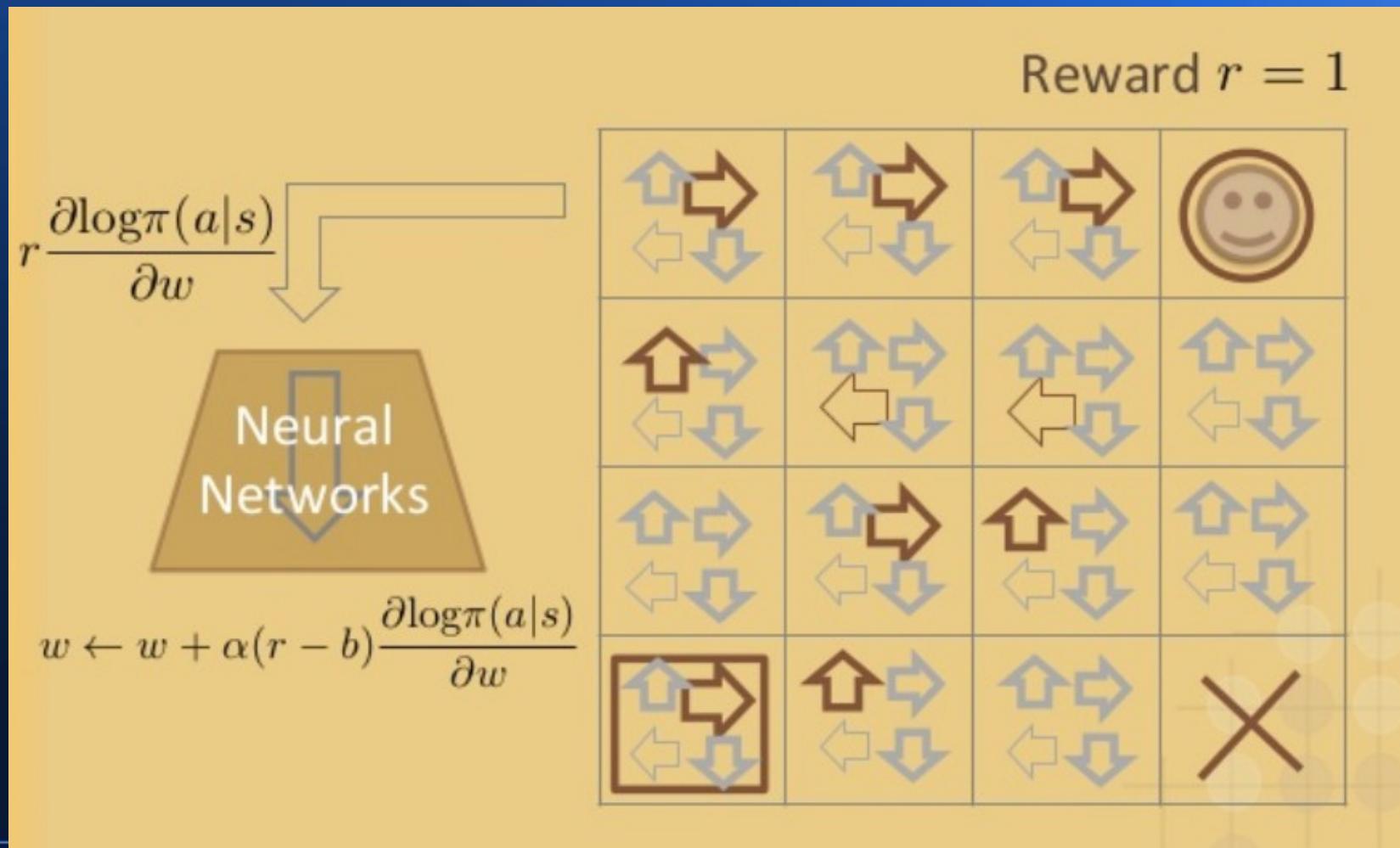


當機器人走到有垃圾的地方

掃到垃圾就算得了一分，這得分會視為獎勵

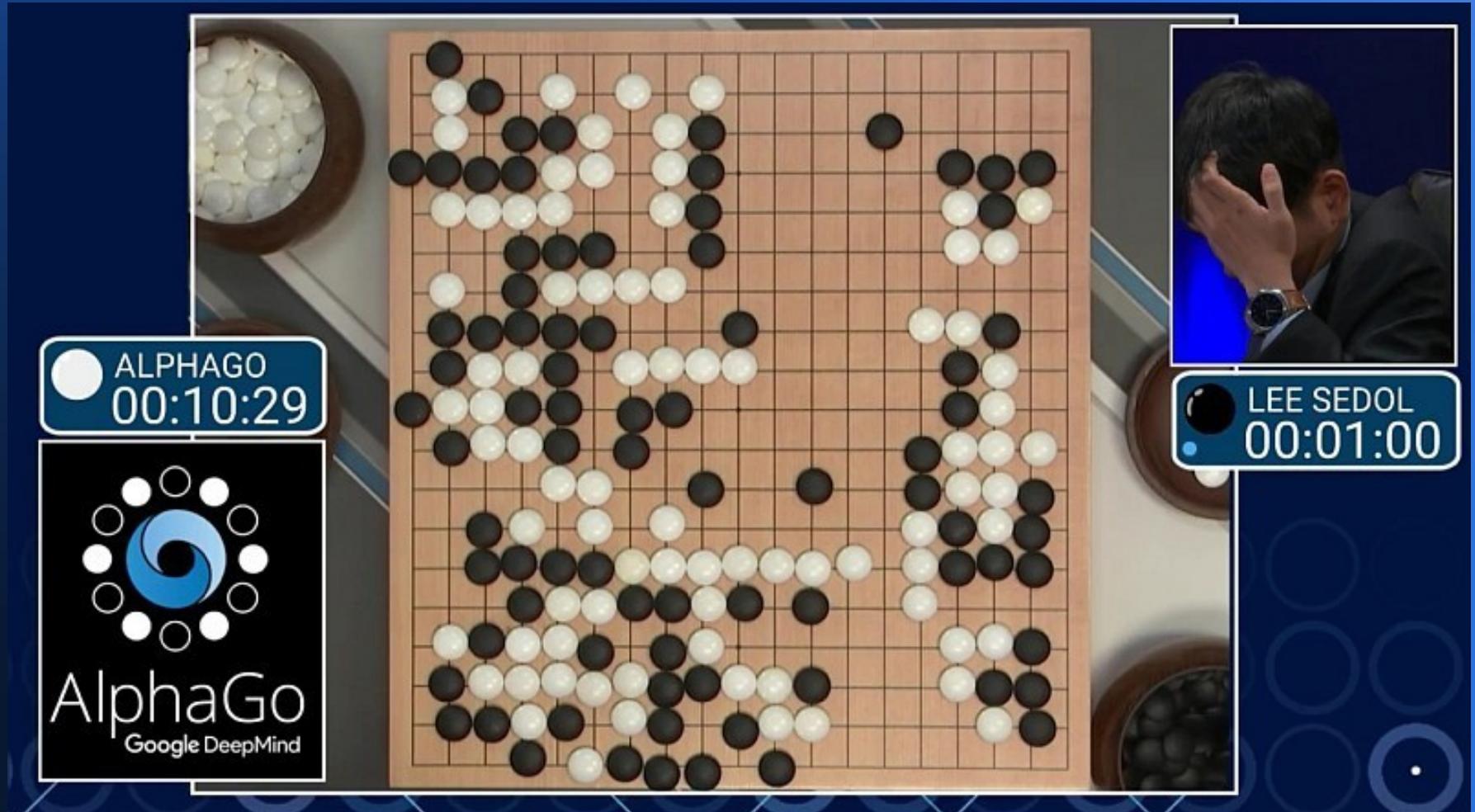


於是會將該路徑走法加分



前年的 AlphaGo 圍棋程式

也是用強化學習 + 神經網路 + 蒙地卡羅搜尋

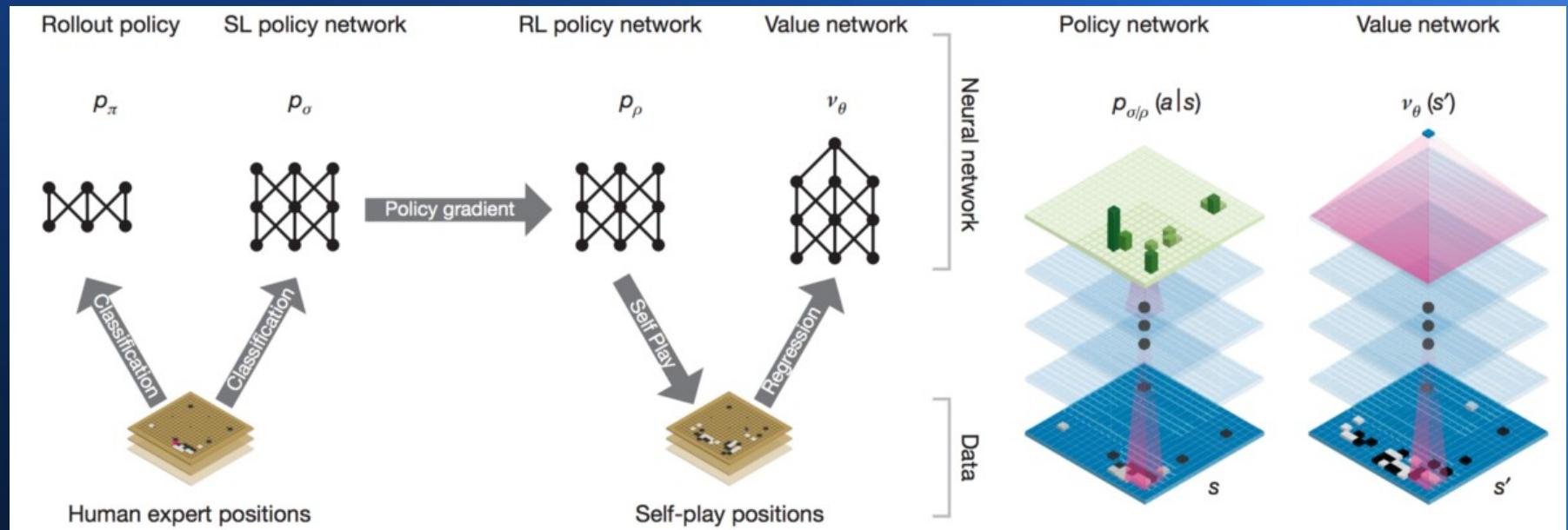


結果五戰四勝贏李世石之後

隔年又三戰全勝大贏世界排名第一的柯潔



AlphaGo 採用了兩組神經網路 策略網路和價值網路



AlphaGo 會向幾千萬局的棋譜學習

- 也有採用自我對打的方式強化

後來的進化版 AlphaGo Zero

- 完全不向棋譜學習
- 只靠自我對打 ...
- 而且只有一套神經網路
- 結果棋力比 AlphaGo 更強 ...

接著 AlphaGo 的創造者 DeepMind 公司

- 又創造出更強的 AlphaZero
- 不只能下圍棋，而且只要有確定規則的棋類，都能透過自我對打成為高手。

這種武功

- 應該算是《左右互博之術》吧！

AlphaGo, AlphaGo Zero, Alpha Zero

- 分別登上了一次 Nature 期刊
- 總共上了三次 …

以上

- 就是神經網路和深度學習的基本概要！

想要知道更多資源請看

- 李宏毅 - 深度學習課程
 - <http://speech.ee.ntu.edu.tw/~tlkagk/courses.html>
- 周莫烦 - Python 深度學習課程
 - <https://www.youtube.com/channel/UCdyjiB5H8Pu7aDTNVXTTpcg>
- Andrej Karpathy - 網誌
 - <https://medium.com/@karpathy>
 - <https://cs.stanford.edu/people/karpathy/>
 - <http://karpathy.github.io/>

謝謝聆聽！