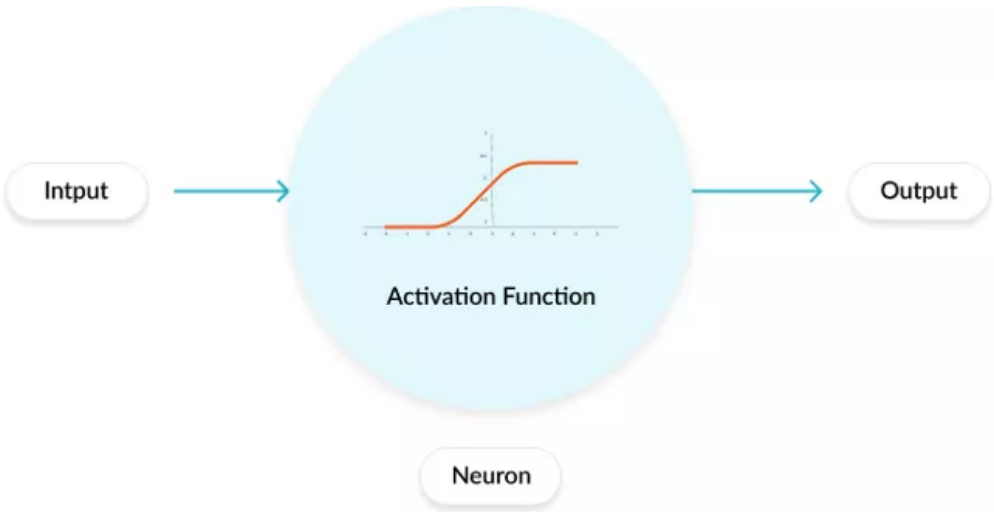
深度學習最常用的10個激活函數！（數學原理+優缺點）

AI蝸牛車 昨天

Datawhale乾貨

作者：Sukanya Bag，來源：機器之心

> 激活函數是神經網絡模型重要的組成部分，本文作者Sukanya Bag從激活函數的數學原理出發，詳解了十種激活函數的優缺點。



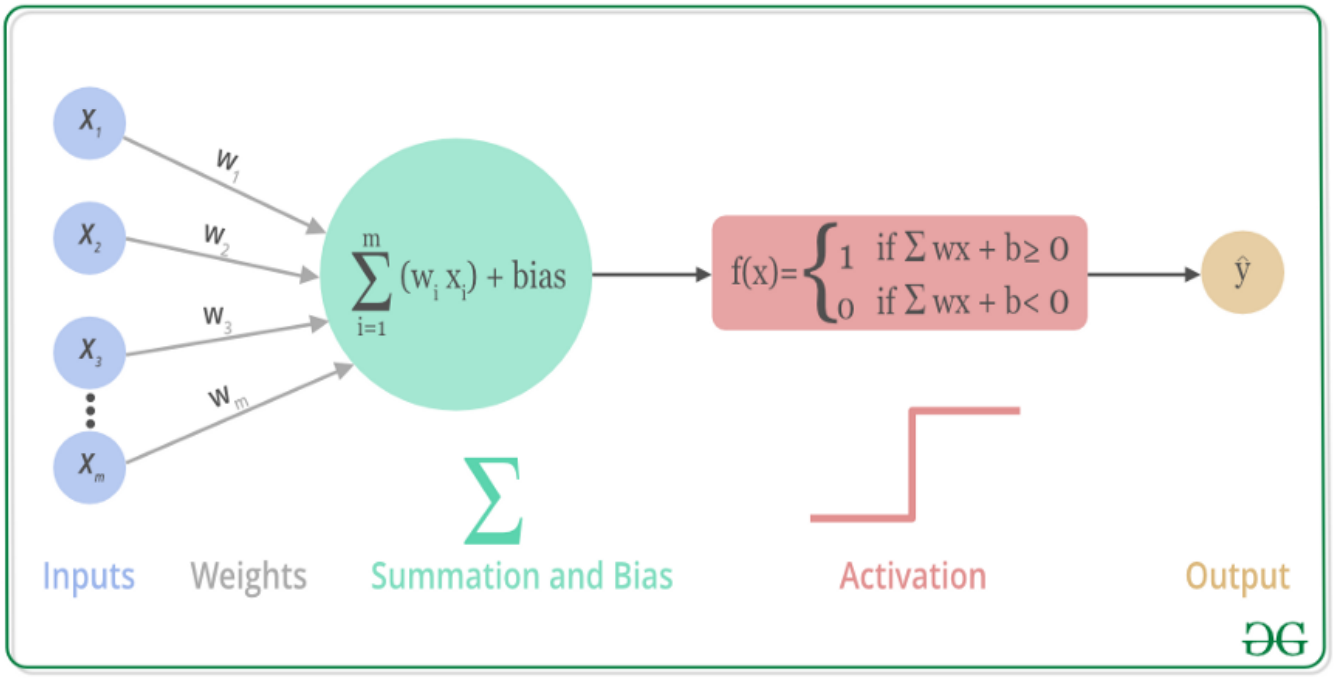
激活函數（Activation Function）是一種添加到人工神經網絡中的函數，旨在幫助網絡學習數據中的複雜模式。類似於人類大腦中基於神經元的模型，激活函數最終決定了要發射給下一個神經元的內容。

在人工神經網絡中，一個節點的激活函數定義了該節點在給定的輸入或輸入集合下的輸出。標準的計算機芯片電路可以看作是根據輸入得到開（1）或關（0）輸出的數字電路激活函數。因此，激活函數是確

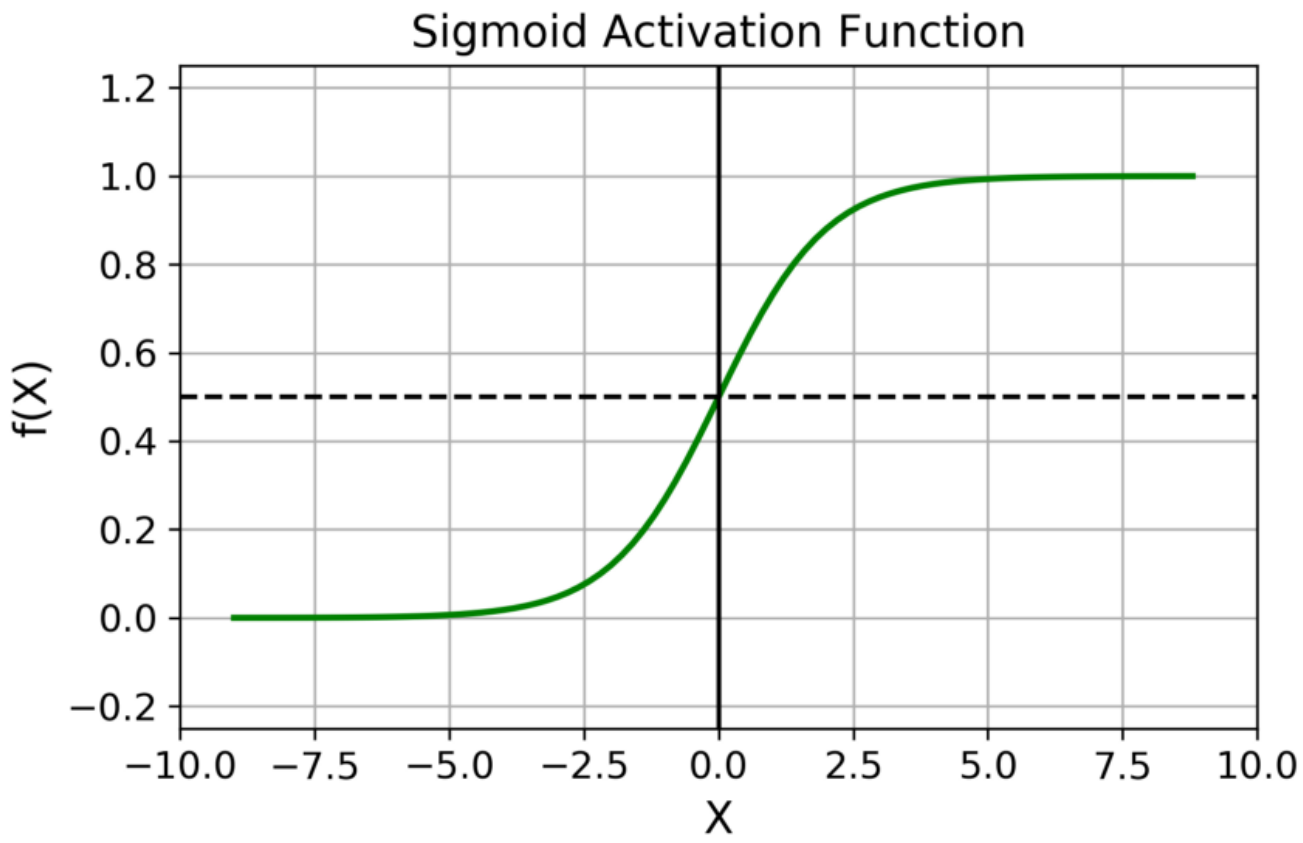
定神經網絡輸出的數學方程式，本文概述了深度學習中常見的十種激活函數及其優缺點。

首先我們來了解一下人工神經元的工作原理，大致如下：

上述過程的數學可視化過程如下圖所示：



1. Sigmoid 激活函數



Sigmoid 函數的圖像看起來像一個S 形曲線。

函數表達式如下：

$$f(z) = 1/(1+ e^{-z})$$

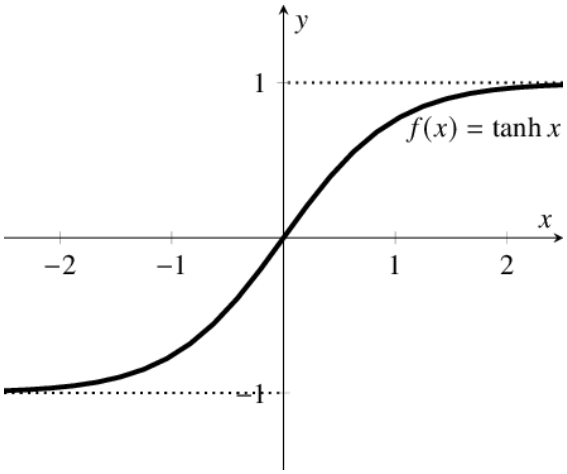
在什麼情況下適合使用Sigmoid 激活函數呢？

- Sigmoid 函數的輸出範圍是0 到1。由於輸出值限定在0 到1，因此它對每個神經元的輸出進行了歸一化；
- 用於將預測概率作為輸出的模型。由於概率的取值範圍是0 到1，因此Sigmoid 函數非常合適；
- 梯度平滑，避免「跳躍」的輸出值；
- 函數是可微的。這意味著可以找到任意兩個點的sigmoid 曲線的斜率；
- 明確的預測，即非常接近1 或0。

Sigmoid 激活函數有哪些缺點？

- 傾向於梯度消失；
- 函數輸出不是以0 為中心的，這會降低權重更新的效率；
- Sigmoid 函數執行指數運算，計算機運行得較慢。

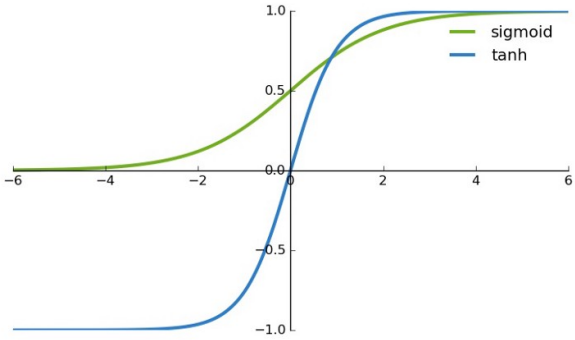
2. Tanh / 雙曲正切激活函數



tanh 激活函數的圖像也是S 形，表達式如下：

$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

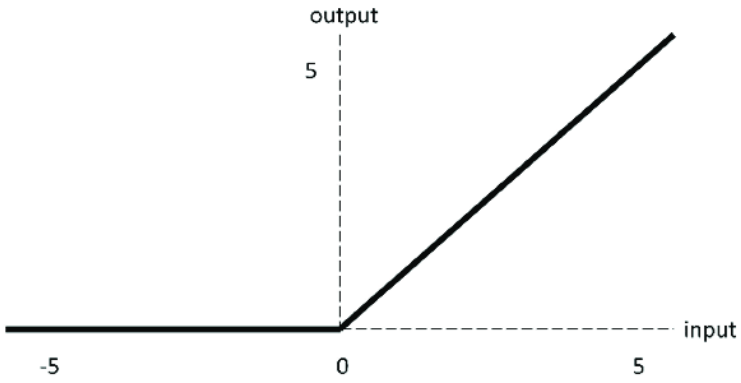
tanh 是一個雙曲正切函數。tanh 函數和sigmoid 函數的曲線相對相似。但是它比sigmoid 函數更有一些優勢。



- 首先，當輸入較大或較小時，輸出幾乎是平滑的並且梯度較小，這不利於權重更新。二者的區別在於輸出間隔，tanh 的輸出間隔為1，並且整個函數以0 為中心，比sigmoid 函數更好；
- 在tanh 圖中，負輸入將被強映射為負，而零輸入被映射為接近零。

注意：在一般的二元分類問題中，tanh 函數用於隱藏層，而sigmoid 函數用於輸出層，但這並不是固定的，需要根據特定問題進行調整。

3. ReLU 激活函數



ReLU 激活函數圖像如上圖所示，函數表達式如下：

$$\sigma(x) = \begin{cases} max(0, x) & , x >= 0 \\ 0 & , x < 0 \end{cases}$$

ReLU 函數是深度學習中較為流行的一種激活函數，相比於sigmoid 函數和tanh 函數，它具有如下優點：

- 當輸入為正時，不存在梯度飽和問題。
- 計算速度快得多。ReLU 函數中只存在線性關係，因此它的計算速度比sigmoid 和tanh 更快。

當然，它也有缺點：

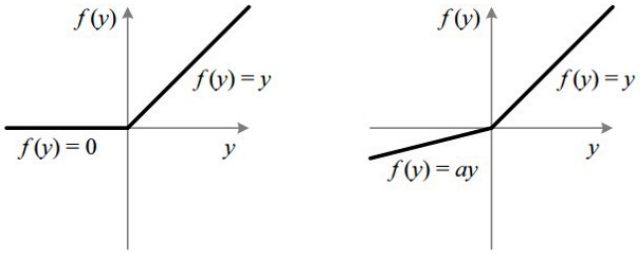
1. Dead ReLU 問題。當輸入為負時，ReLU 完全失效，在正向傳播過程中，這不是問題。有些區域很敏感，有些則不敏感。但是在反向傳播過程中，如果輸入負數，則梯度將完全為零，sigmoid

函數和tanh 函數也具有相同的問題；

2. 我們發現ReLU 函數的輸出為0 或正數，這意味著ReLU 函數不是以0 為中心的函數。

4. Leaky ReLU

它是一種專門設計用於解決Dead ReLU 問題的激活函數：



ReLU vs Leaky ReLU

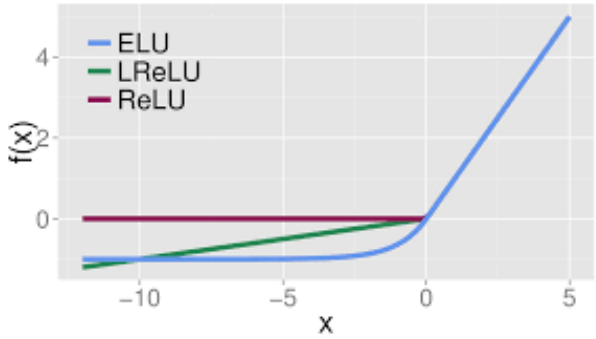
為什麼Leaky ReLU 比ReLU 更好?

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \leq 0 \end{cases}.$$

1. Leaky ReLU 通過把x 的非常小的線性分量給予負輸入（0.01x）來調整負值的零梯度（zero gradients）問題；
2. leak 有助於擴大ReLU 函數的範圍，通常a 的值為0.01 左右；
3. Leaky ReLU 的函數範圍是（負無窮到正無窮）。

注意：從理論上講，Leaky ReLU具有ReLU的所有優點，而且Dead ReLU不會有任何問題，但在實際操作中，尚未完全證明Leaky ReLU總是比ReLU更好。

5. ELU



ELU vs Leaky ReLU vs ReLU

ELU 的提出也解決了ReLU 的問題。與ReLU 相比，ELU 有負值，這會使激活的平均值接近零。均值激活接近於零可以使學習更快，因為它們使梯度更接近自然梯度。

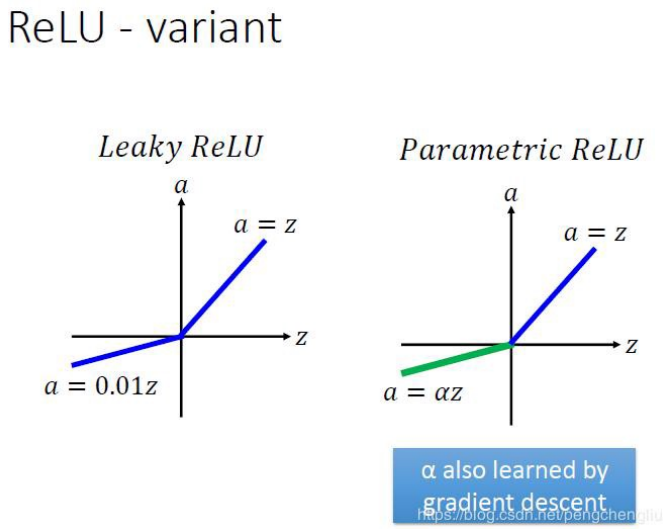
$$g(x) = \mathrm{ELU}(x) = \begin{cases} x, & x > 0 \\ \alpha(e^x - 1), & x \leqslant 0 \end{cases}$$

顯然，ELU 具有ReLU 的所有優點，並且：

- 沒有Dead ReLU 問題，輸出的平均值接近0，以0 為中心；
- ELU 通過減少偏置偏移的影響，使正常梯度更接近於單位自然梯度，從而使均值向零加速學習；
- ELU 在較小的輸入下會飽和至負值，從而減少前向傳播的變異和信息。

一個小問題是它的計算強度更高。與Leaky ReLU 類似，儘管理論上比ReLU 要好，但目前在實踐中沒有充分的證據表明ELU 總是比ReLU 好。

6. PReLU（Parametric ReLU）



PReLU 也是ReLU 的改進版本：

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \leq 0 \end{cases}$$

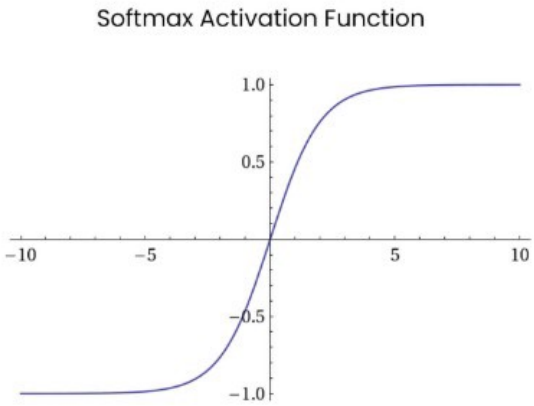
看一下PReLU 的公式：參數α通常為0 到1 之間的數字，並且通常相對較小。

- 如果a_i= 0，則f 變為ReLU
- 如果a_i> 0，則f 變為leaky ReLU
- 如果a_i 是可學習的參數，則f 變為PReLU

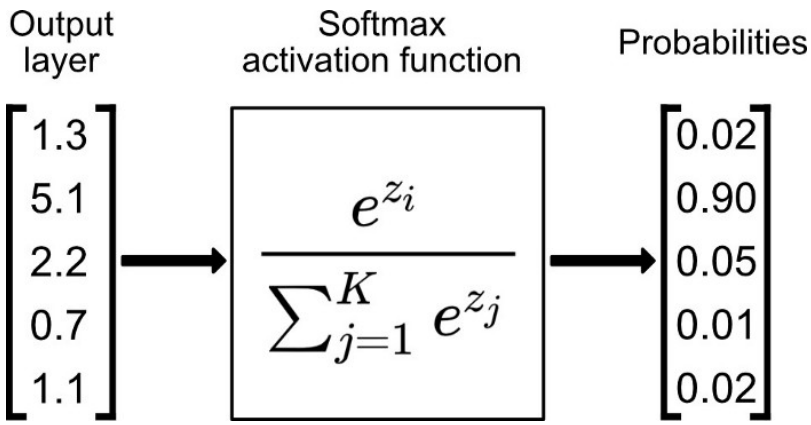
PReLU 的優點如下：

1. 在負值域，PReLU 的斜率較小，這也可以避免Dead ReLU 問題。
2. 與ELU 相比，PReLU 在負值域是線性運算。儘管斜率很小，但不會趨於0。

7. Softmax



Softmax 是用於多類分類問題的激活函數，在多類分類問題中，超過兩個類標籤則需要類成員關係。對於長度為K 的任意實向量，Softmax 可以將其壓縮為長度為K，值在（0，1）範圍內，並且向量中元素的總和為1 的實向量。



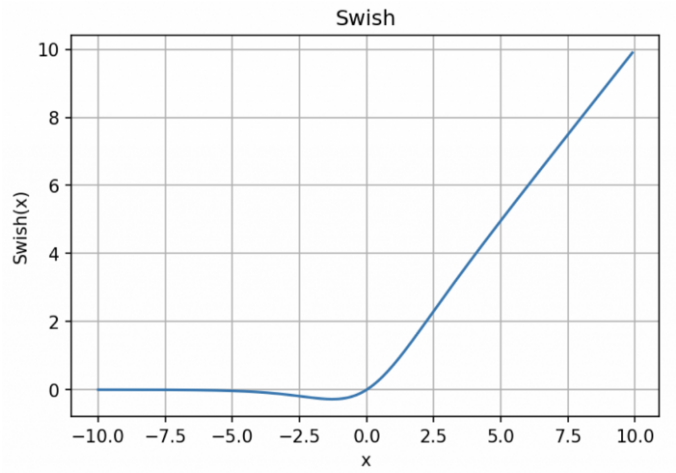
Softmax 與正常的max 函數不同：max 函數僅輸出最大值，但Softmax 確保較小的值具有較小的概率，並且不會直接丟棄。我們可以認為它是argmax 函數的概率版本或「soft」版本。

Softmax 函數的分母結合了原始輸出值的所有因子，這意味著Softmax 函數獲得的各種概率彼此相關。

Softmax 激活函數的主要缺點是：

1. 在零點不可微；
2. 負輸入的梯度為零，這意味著對於該區域的激活，權重不會在反向傳播期間更新，因此會產生永不激活的死亡神經元。

8. Swish



函數表達式：y = x * sigmoid (x)

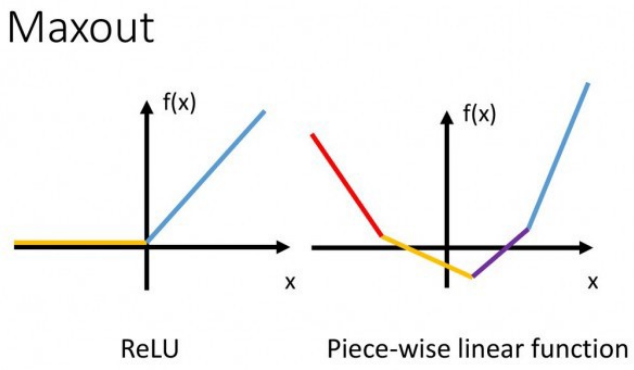
Swish 的設計受到了LSTM 和高速網絡中gating 的sigmoid 函數使用的啟發。我們使用相同的gating 值來簡化gating 機制，這稱為self-gating。

self-gating 的優點在於它只需要簡單的標量輸入，而普通的gating 則需要多個標量輸入。這使得諸如Swish 之類的self-gated 激活函數能夠輕鬆替換以單個標量為輸入的激活函數（例如ReLU），而無需更改隱藏容量或參數數量。

Swish 激活函數的主要優點如下：

- 「無界性」有助於防止慢速訓練期間，梯度逐漸接近0並導致飽和；（同時，有界性也是有優勢的，因為有界激活函數可以具有很強的正則化，並且較大的負輸入問題也能解決）；
- 導數恆> 0；
- 平滑度在優化和泛化中起了重要作用。

9. Maxout

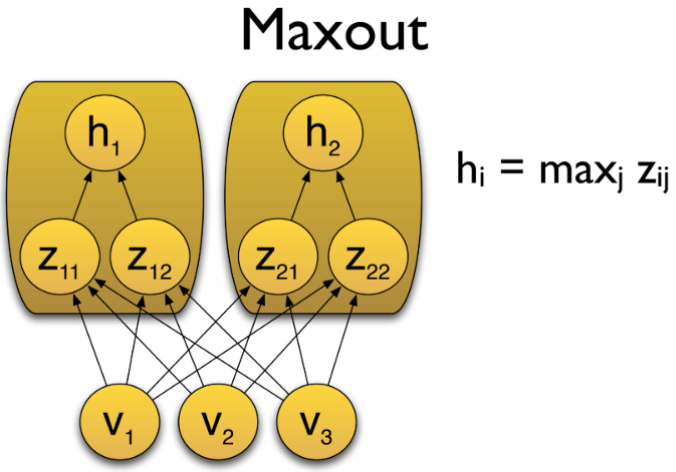


在Maxout 層，激活函數是輸入的最大值，因此只有2 個maxout 節點的多層感知機就可以擬合任意的凸函數。

單個Maxout 節點可以解釋為對一個實值函數進行分段線性近似(PWL)，其中函數圖上任意兩點之間的線段位於圖（凸函數）的上方。

$$ReLU = \max(0, x),\ \mathrm{abs}(x) = \max(x,\ -x)$$

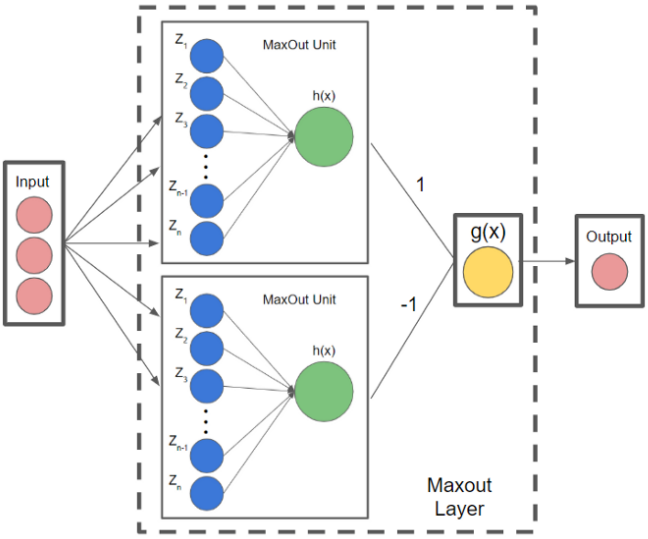
Maxout 也可以對d 維向量（V）實現：



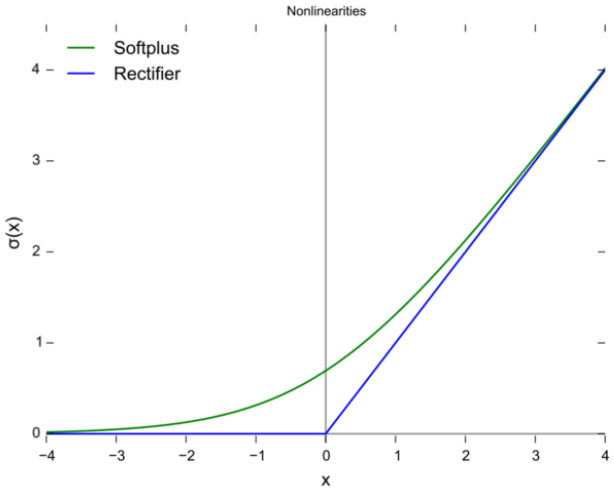
假設兩個凸函數h_1(x) 和h_2(x)，由兩個Maxout 節點近似化，函數g(x) 是連續的PWL 函數。

$$g(x) = h_1(x) - h_2(x)$$

因此，由兩個Maxout 節點組成的Maxout 層可以很好地近似任何連續函數。



10. Softplus



Softplus 函數：f（x）= ln（1 + exp x）

Softplus 的導數為

f ‘(x)=exp(x) / (1+exp x)

= 1/ (1 +exp(−x))

，也稱為logistic / sigmoid 函數。

Softplus 函數類似於ReLU 函數，但是相對較平滑，像ReLU 一樣是單側抑制。它的接受範圍很廣：(0, + inf)。

原文鏈接：https://sukanyabag.medium.com/activation-functions-all-you-need-to-know-355a850d025e

更多精彩內容（請點擊圖片進行閱讀）