機器與深度學習基礎知識初探
-導傳遞

黃志勝 Chih-Sheng (Tommy) Huang

義隆電子人工智慧研發部

國立陽明交通大學AI學院合聘助理教授

基礎機器學習與深度學習

- 1. 神經網路如何運作
- 2. 梯度下降法
- 3. 神經網路如何利用導傳遞找解
- 4. 神經網路太深層的問題
- 5. Batch Normalization在幹什麼

基礎機器學習與深度學習

- 神經網路如何利用導傳遞找解
- 神經網路太深層的問題:

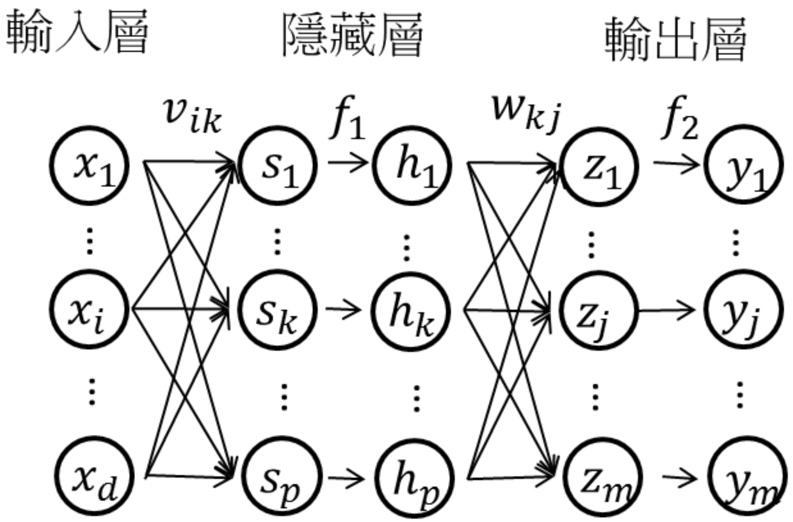
梯度消失問題(Vanishing gradient)/梯度爆炸問題（exploding gradient problem）。

Activation Function為什麼要採用ReLU，而不是用Sigmoid。

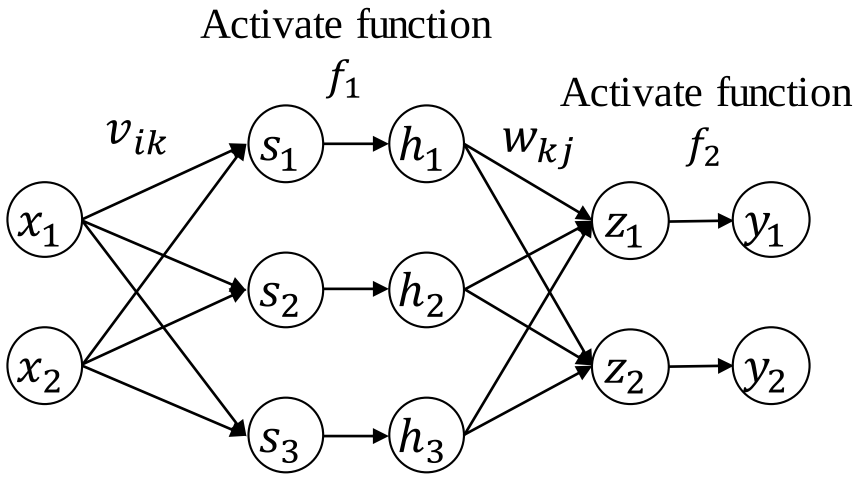
Residual block克服神經網路不能太深層的問題。

神經網路如何利用導傳遞找解

- 下圖為一個三層的MLP



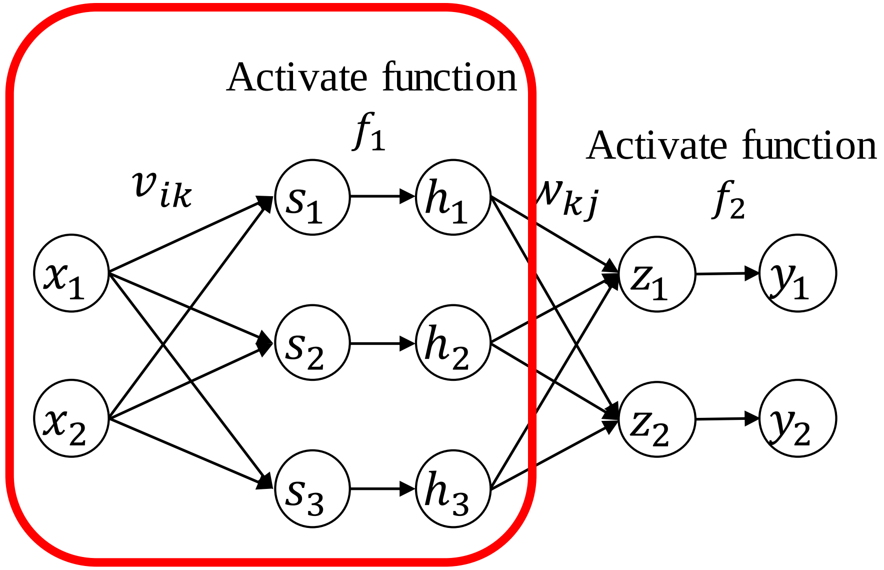
簡化



$i = 1,2$
$k = 1,2,3$
$j = 1,2$

Forward propagation

輸入層→隱藏層



$i = 1,2$
$k = 1,2,3$
$\mathrm{j} = 1,2$

$$s_1 = v_{11}x_1 + v_{21}x_2 = \boldsymbol{v}_1^T\boldsymbol{x}$$
$$s_2 = v_{12}x_1 + v_{22}x_2 = \boldsymbol{v}_2^T\boldsymbol{x}$$
$$s_3 = v_{13}x_1 + v_{23}x_2 = \boldsymbol{v}_3^T\boldsymbol{x}$$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \boldsymbol{v}_k = \begin{bmatrix} v_{1k} \\ v_{2k} \end{bmatrix}, k = 1,2,3$$

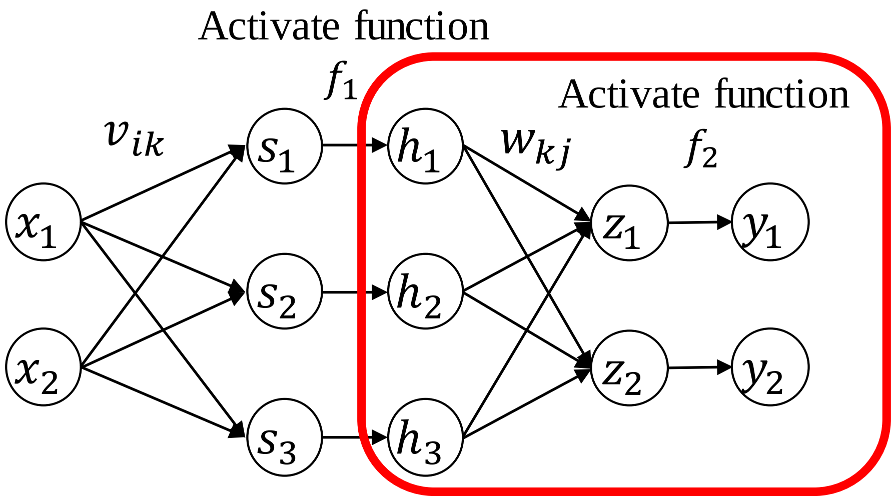
$$s_k = \sum_{i=1}^{2} v_{ik}x_i$$
$$\boldsymbol{s} = \boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x}$$
$$\boldsymbol{s} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix}$$
$$\boldsymbol{v} = [\boldsymbol{v}_1 \quad \boldsymbol{v}_2 \quad \boldsymbol{v}_2] = \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix}$$

Activate function

$$\boldsymbol{h} = f_1(\boldsymbol{s})$$
$$\boldsymbol{h} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

Forward propagation

隱藏層→輸出層



$i = 1,2$

$k = 1,2,3$

$\mathrm{j} = 1,2$

$$z_1 = w_{11}h_1 + w_{21}h_2 + w_{31}h_3 = \boldsymbol{w}_1^T\boldsymbol{h}$$
$$z_2 = w_{12}h_1 + w_{22}h_2 + w_{32}h_3 = \boldsymbol{w}_2^T\boldsymbol{h}$$

$$\boldsymbol{w}_j = \begin{bmatrix} w_{1j} \\ w_{2j} \\ w_{3j} \end{bmatrix}, j = 1,2$$

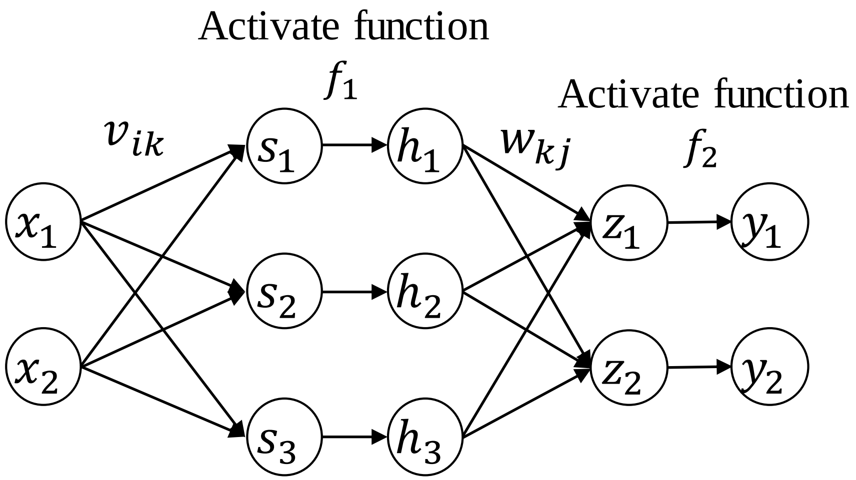
$$\boldsymbol{z} = \boldsymbol{w}^T\boldsymbol{h}$$
$$\boldsymbol{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$\boldsymbol{w} = [\boldsymbol{w}_1 \quad \boldsymbol{w}_2] = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix}$$

Activate function

$$\boldsymbol{y} = f_2(\boldsymbol{z})$$
$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

Forward propagation



$i = 1,2$
$k = 1,2,3$
j = 1,2

輸入層→隱藏層

$$\boldsymbol{s} = \boldsymbol{v}^T \boldsymbol{x}$$
$$\boldsymbol{h} = f_1(\boldsymbol{s})$$

隱藏層→輸出層

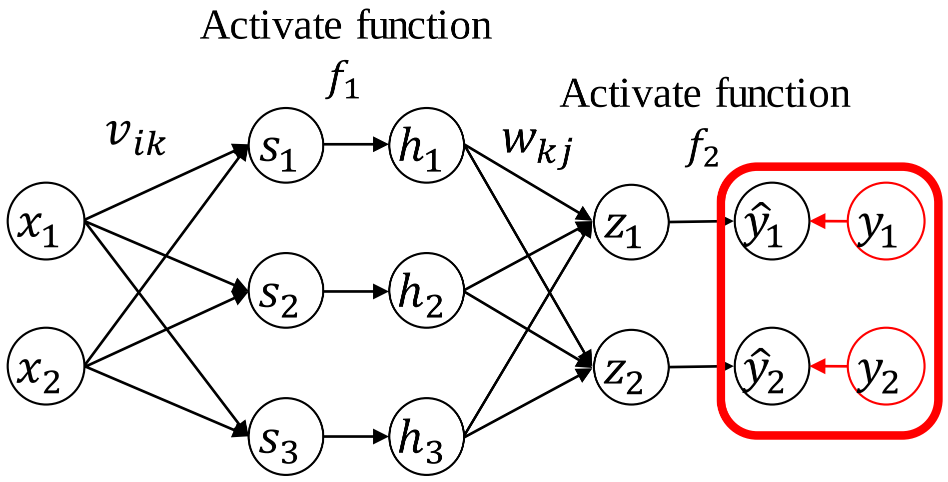
$$\boldsymbol{z} = \boldsymbol{w}^T \boldsymbol{h}$$
$$\boldsymbol{y} = f_2(\boldsymbol{z})$$

=

$$\boldsymbol{y} = f_2\left(\boldsymbol{w}^T f_1(\boldsymbol{v}^T \boldsymbol{x})\right)$$

參數為$\boldsymbol{v}$和$\boldsymbol{w}$。

Back-propagation



$$\hat{\boldsymbol{y}} = f_2\left(\boldsymbol{w}^T f_1(\boldsymbol{v}^T \boldsymbol{x})\right) \qquad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

我們用SSE來當作loss function

$$loss(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{2}(\boldsymbol{y} - \hat{\boldsymbol{y}})^T(\boldsymbol{y} - \hat{\boldsymbol{y}})$$

利用gradient descent找最佳參數解(參數只有$\boldsymbol{w}$和$\boldsymbol{v}$)

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \alpha \nabla \boldsymbol{w}^{(t)}$$
$$\boldsymbol{v}^{(t+1)} = \boldsymbol{v}^{(t)} - \alpha \nabla \boldsymbol{v}^{(t)}$$

輸入層→隱藏層

$$\boldsymbol{s} = \boldsymbol{v}^T \boldsymbol{x}$$
$$\boldsymbol{h} = f_1(\boldsymbol{s})$$

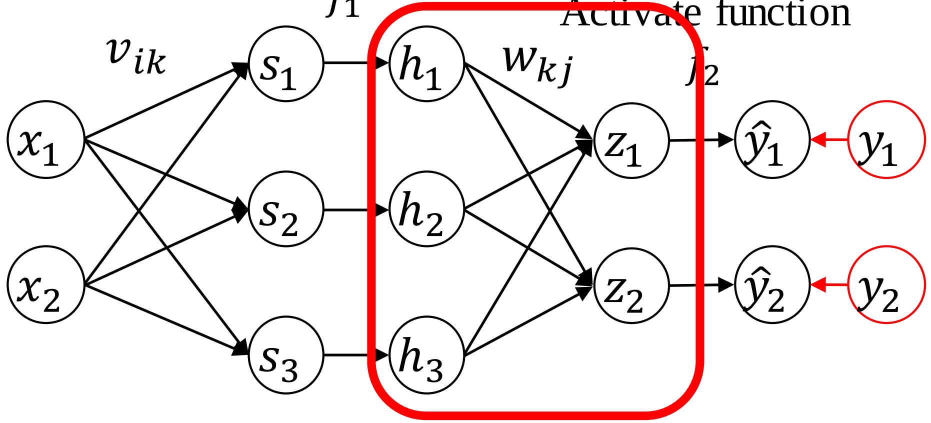
隱藏層→輸出層

$$\boldsymbol{z} = \boldsymbol{w}^T \boldsymbol{h}$$
$$\boldsymbol{y} = f_2(\boldsymbol{z})$$

$\nabla \boldsymbol{w} = \dfrac{\partial loss(\boldsymbol{y}, \hat{\boldsymbol{y}})}{\partial \boldsymbol{w}}$ 輸出層→隱藏層參數為$\boldsymbol{w}$，對loss進行$\boldsymbol{w}$的偏微分

$\nabla \boldsymbol{v} = \dfrac{\partial loss(\boldsymbol{y}, \hat{\boldsymbol{y}})}{\partial \boldsymbol{v}}$ 隱藏層→輸入層參數為$\boldsymbol{v}$，對loss進行$\boldsymbol{v}$的偏微分

Back-propagation
輸出層→隱藏層



隱藏層→輸出層

$$\boldsymbol{z} = \boldsymbol{w}^T\boldsymbol{h}$$

$$\boldsymbol{y} = f_2(\boldsymbol{z})$$

參數為$\boldsymbol{w}$，因此我們對loss進行$\boldsymbol{w}$的偏微分

$$loss(\boldsymbol{y},\hat{\boldsymbol{y}}) = \frac{1}{2}(\boldsymbol{y}-\hat{\boldsymbol{y}})^T(\boldsymbol{y}-\hat{\boldsymbol{y}})$$

$$= \frac{1}{2}\left(\boldsymbol{y}-f_2(\boldsymbol{z})\right)^T\left(\boldsymbol{y}-f_2(\boldsymbol{z})\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y}-f_2(\boldsymbol{w}^T\boldsymbol{h})\right)^T\left(\boldsymbol{y}-f_2(\boldsymbol{w}^T\boldsymbol{h})\right)$$

Chain rule

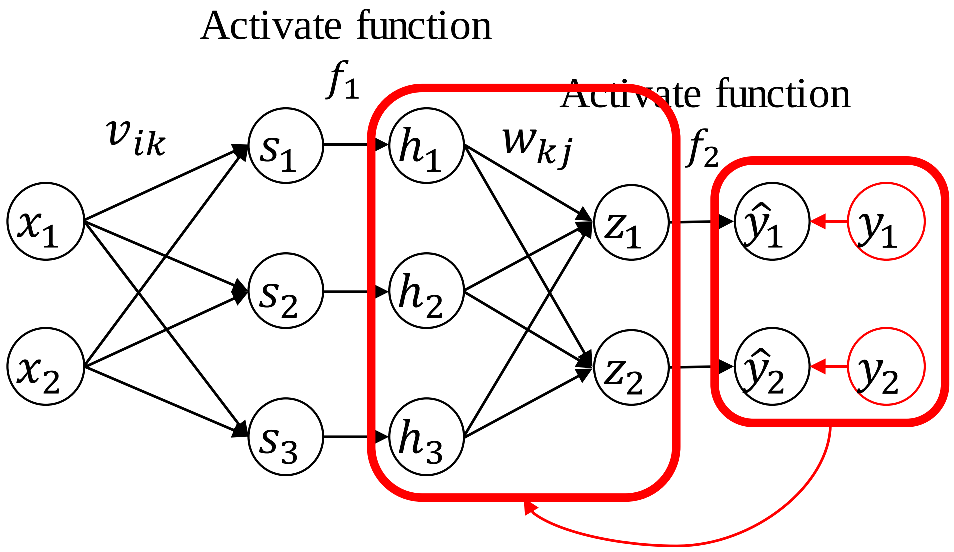
$$\Delta\boldsymbol{w} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{w}} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{w}}$$

$$\frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{z}} = \frac{\frac{1}{2}\left(\boldsymbol{y}-f_2(\boldsymbol{z})\right)^T\left(\boldsymbol{y}-f_2(\boldsymbol{z})\right)}{\partial\boldsymbol{z}}$$

$$= \left(\boldsymbol{y}-f_2(\boldsymbol{z})\right)\frac{\partial f_2(\boldsymbol{z})}{\partial\boldsymbol{z}} = (\boldsymbol{y}-\hat{\boldsymbol{y}})f_2'(\boldsymbol{z})$$

$$\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{w}} = \frac{\partial\boldsymbol{w}^T\boldsymbol{h}}{\partial\boldsymbol{w}} = \boldsymbol{h}$$

Back-propagation
輸出層→隱藏層



隱藏層→輸出層

$$\boldsymbol{z} = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{h}$$

$$\boldsymbol{y} = f_2(\boldsymbol{z})$$

參數為$\boldsymbol{w}$，因此我們對loss進行$\boldsymbol{w}$的偏微分

$$loss(\boldsymbol{y},\hat{\boldsymbol{y}}) = \frac{1}{2}(\boldsymbol{y}-\hat{\boldsymbol{y}})^{\boldsymbol{T}}(\boldsymbol{y}-\hat{\boldsymbol{y}})$$

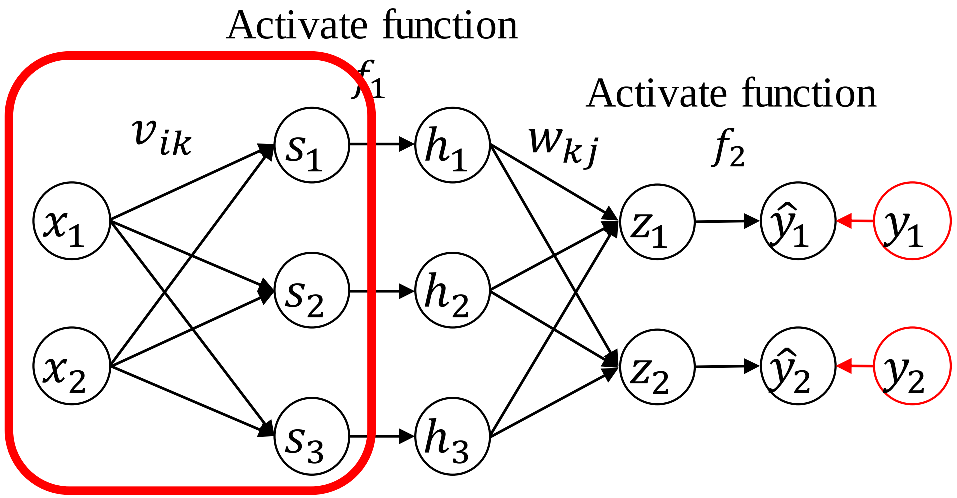
$$= \frac{1}{2}\left(\boldsymbol{y}-f_2(\boldsymbol{z})\right)^{\boldsymbol{T}}\left(\boldsymbol{y}-f_2(\boldsymbol{z})\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y}-f_2\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{h}\right)\right)^{\boldsymbol{T}}\left(\boldsymbol{y}-f_2\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{h}\right)\right)$$

$$\Delta\boldsymbol{w} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{w}} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{w}}$$

$$= (\boldsymbol{y}-\hat{\boldsymbol{y}})f_2{}'(\boldsymbol{z})\boldsymbol{h}$$

Back-propagation
隱藏層→輸入層



$$loss(\boldsymbol{y},\hat{\boldsymbol{y}}) = \left(\boldsymbol{y} - f_2(\boldsymbol{z})\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2(\boldsymbol{z})\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{s})\right)\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{s})\right)\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x})\right)\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x})\right)\right)$$

Chain rule

$$\Delta\boldsymbol{v} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{v}} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{s}}\frac{\partial\boldsymbol{s}}{\partial\boldsymbol{v}} = \boxed{\frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{z}}}\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{s}}\frac{\partial\boldsymbol{s}}{\partial\boldsymbol{v}}$$

前面解過了

$$\frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{z}} = (\boldsymbol{y} - \hat{\boldsymbol{y}}) f_2{}'(\boldsymbol{z})$$

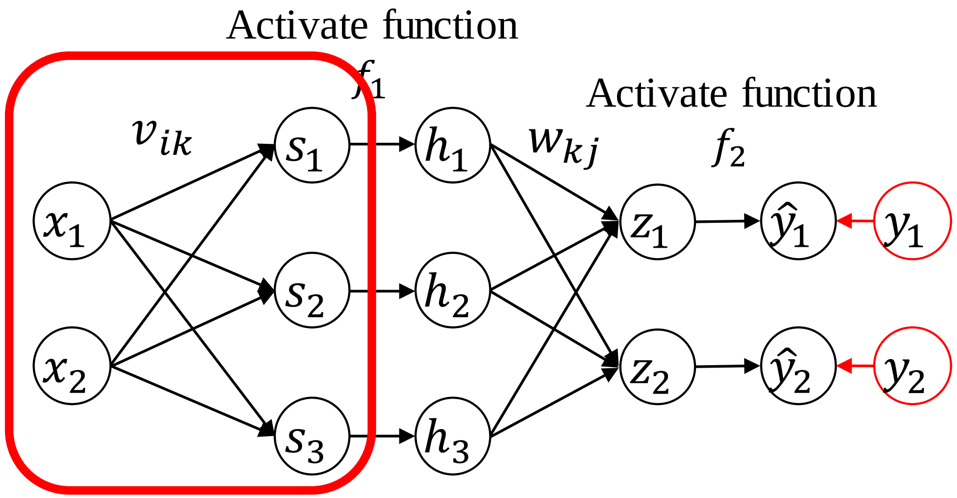
隱藏層→輸入層

$$\boldsymbol{s} = \boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x}$$

$$\boldsymbol{h} = f_1(\boldsymbol{s})$$

參數為$\boldsymbol{v}$，因此我們對loss進行$\boldsymbol{v}$的偏微分

Back-propagation
隱藏層→輸入層



$$loss(\boldsymbol{y},\hat{\boldsymbol{y}}) = \left(\boldsymbol{y} - f_2(\boldsymbol{z})\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2(\boldsymbol{z})\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{s})\right)\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{s})\right)\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x})\right)\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x})\right)\right)$$

Chain rule

$$\Delta\boldsymbol{v} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{v}} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{s}}\frac{\partial\boldsymbol{s}}{\partial\boldsymbol{v}} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{z}}\boxed{\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{s}}}\boxed{\frac{\partial\boldsymbol{s}}{\partial\boldsymbol{v}}}$$

$$\boxed{\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{s}} = \frac{\partial\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{s})}{\partial\boldsymbol{s}} = \boldsymbol{w}^{\boldsymbol{T}} f_1{}'(\boldsymbol{s})}$$

$$\boxed{\frac{\partial\boldsymbol{s}}{\partial\boldsymbol{v}} = \frac{\partial\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x}}{\partial\boldsymbol{v}} = \boldsymbol{x}}$$

隱藏層→輸入層

$$\boldsymbol{s} = \boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x}$$

$$\boldsymbol{h} = f_1(\boldsymbol{s})$$

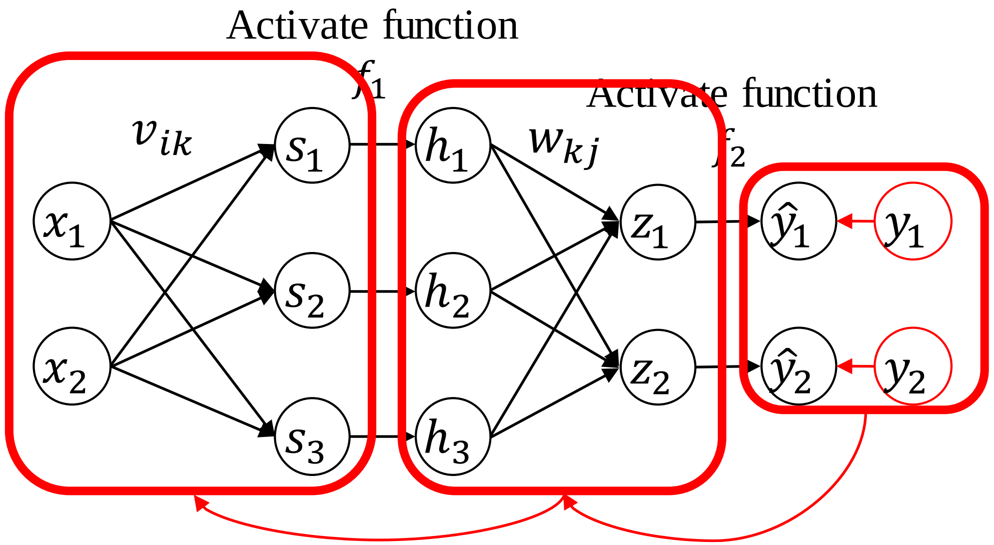
參數為$\boldsymbol{v}$，因此我們對loss進行$\boldsymbol{v}$的偏微分

Back-propagation
隱藏層→輸入層

$$loss(\boldsymbol{y},\hat{\boldsymbol{y}}) = \left(\boldsymbol{y} - f_2(\boldsymbol{z})\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2(\boldsymbol{z})\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{s})\right)\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{s})\right)\right)$$

$$= \frac{1}{2}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x})\right)\right)^{\boldsymbol{T}}\left(\boldsymbol{y} - f_2\left(\boldsymbol{w}^{\boldsymbol{T}} f_1(\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x})\right)\right)$$



$$\Delta\boldsymbol{v} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial \boldsymbol{v}} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial \boldsymbol{s}}\frac{\partial \boldsymbol{s}}{\partial \boldsymbol{v}} = \frac{\partial loss(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial \boldsymbol{z}}\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{s}}\frac{\partial \boldsymbol{s}}{\partial \boldsymbol{v}}$$

$$= \left[(\boldsymbol{y} - \hat{\boldsymbol{y}}) f_2{}'(\boldsymbol{z})\right]\left[\boldsymbol{w}^{\boldsymbol{T}} f_1{}'(\boldsymbol{s})\right]\boldsymbol{x}$$

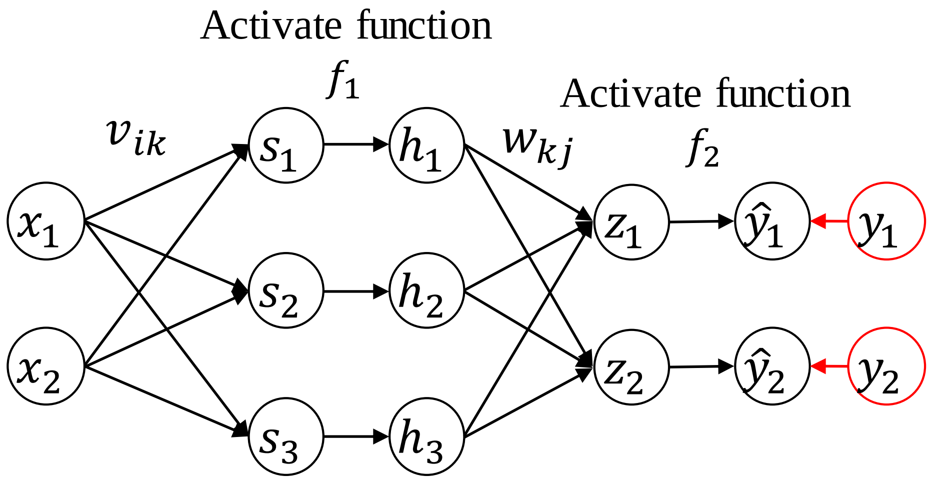
隱藏層→輸入層

$$\boldsymbol{s} = \boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x}$$

$$\boldsymbol{h} = f_1(\boldsymbol{s})$$

參數為$\boldsymbol{v}$，因此我們對loss進行$\boldsymbol{v}$的偏微分

Back-propagation



$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \alpha \Delta \boldsymbol{w}^{(t)}$$

$$\boldsymbol{v}^{(t+1)} = \boldsymbol{v}^{(t)} - \alpha \Delta \boldsymbol{v}^{(t)}$$

輸出層→隱藏層參數為$\boldsymbol{w}$

$$\Delta \boldsymbol{w} = \boxed{(\boldsymbol{y} - \hat{\boldsymbol{y}}) f_2{}'(\boldsymbol{z})}\, \boldsymbol{h}$$

隱藏層→輸入層參數為$\boldsymbol{v}$

$$\Delta \boldsymbol{v} = \boxed{[(\boldsymbol{y} - \hat{\boldsymbol{y}}) f_2{}'(\boldsymbol{z})]} \boxed{[\boldsymbol{w}^T f_1{}'(\boldsymbol{s})]}\, \boldsymbol{x}$$

前面層的gradient
會是由後面所有層的gradient和現在這層的疊乘。

輸入層→隱藏層

$$\boldsymbol{s} = \boldsymbol{v}^T \boldsymbol{x}$$
$$\boldsymbol{h} = f_1(\boldsymbol{s})$$

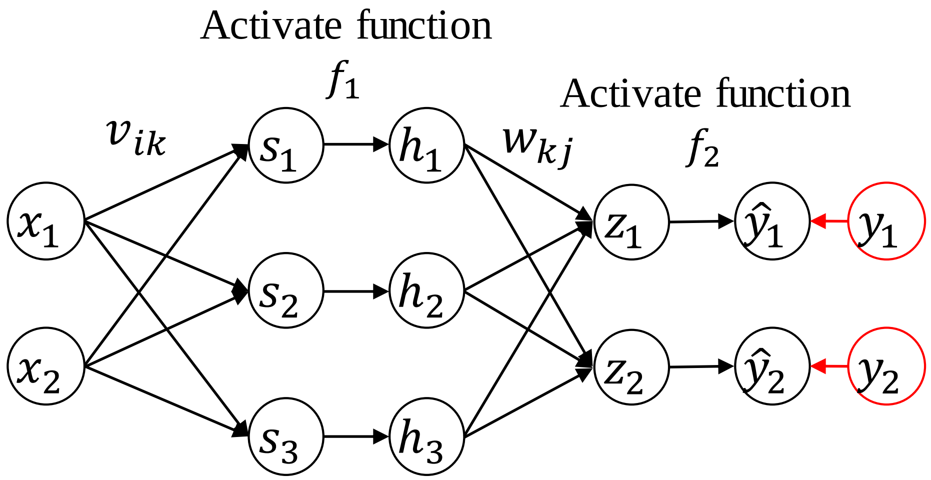
隱藏層→輸出層

$$\boldsymbol{z} = \boldsymbol{w}^T \boldsymbol{h}$$
$$\boldsymbol{y} = f_2(\boldsymbol{z})$$

神經網路太深層的問題

神經網路太深層的問題

Sigmoid在神經網路太深層的問題



輸出層→隱藏層參數為$\boldsymbol{w}$

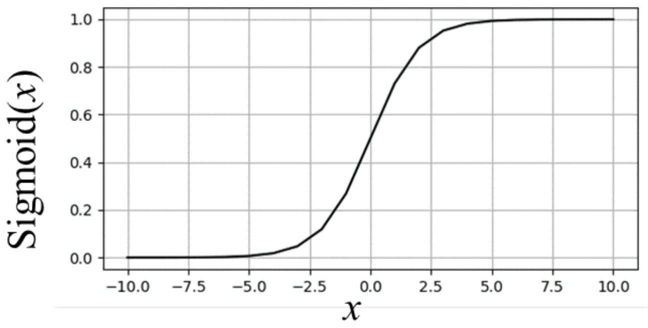
$$\Delta \boldsymbol{w} = \boxed{(\boldsymbol{y}-\hat{\boldsymbol{y}}) f_2{}'(\boldsymbol{z})}\, \boldsymbol{h}$$

隱藏層→輸入層參數為$\boldsymbol{v}$

$$\Delta \boldsymbol{v} = \boxed{[(\boldsymbol{y}-\hat{\boldsymbol{y}}) f_2{}'(\boldsymbol{z})]}\boxed{[\boldsymbol{w}^{\boldsymbol{T}} f_1{}'(\boldsymbol{s})]}\, \boldsymbol{x}$$

$$\text{Sigmoid}: f(x) = \frac{1}{1+e^{-x}}$$

$$f' = f(x)(1-f(x))$$



輸入層→隱藏層

$$\boldsymbol{s} = \boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{x}$$
$$\boldsymbol{h} = f_1(\boldsymbol{s})$$

隱藏層→輸出層

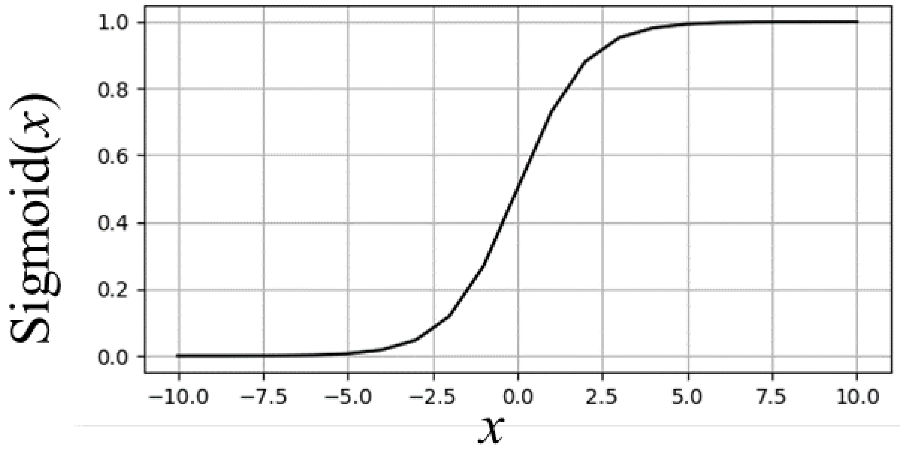
$$\boldsymbol{z} = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{h}$$
$$\boldsymbol{y} = f_2(\boldsymbol{z})$$

Sigmoid在神經網路太深層的問題

Sigmoid : $f(x) = \frac{1}{1+e^{-x}}$

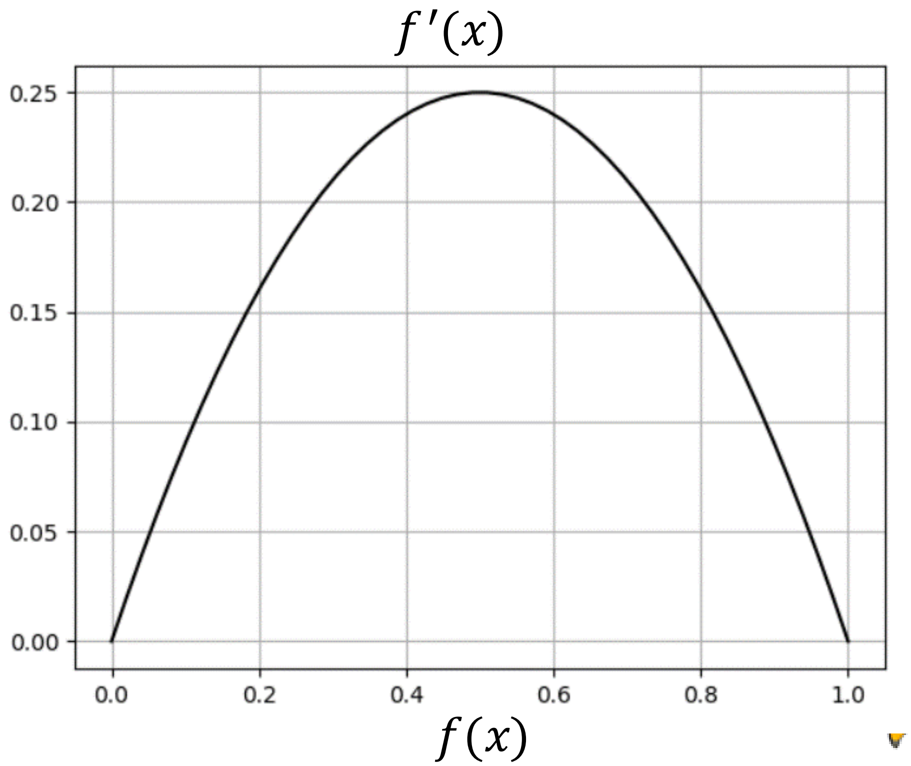
$f'(x) = f(x)\big(1 - f(x)\big)$

$f(x)$輸出介於0~1



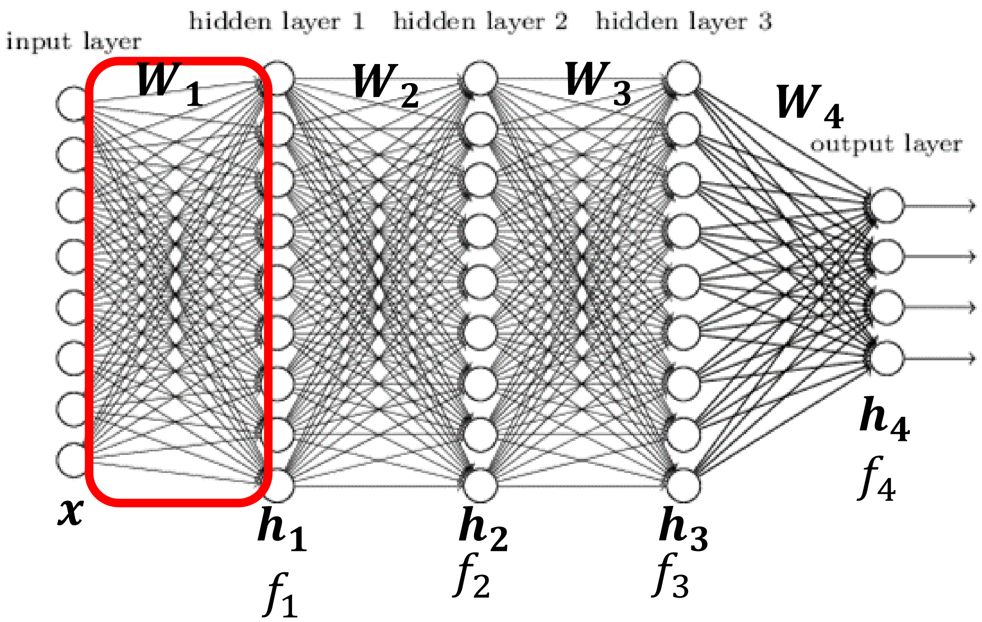
$f'(x)$會更小，最大值為0.25

$f(x) = 0.5$，$f'(x) = 0.5 \times 0.5 = 0.25$



Sigmoid在神經網路太深層的問題

MLP



$$\Delta \boldsymbol{W}_1 = [(\boldsymbol{y}-\hat{\boldsymbol{y}}) f_4'(\boldsymbol{h}_4)][\boldsymbol{W}_3^T f_3'(\boldsymbol{h}_3)][\boldsymbol{W}_2^T f_2'(\boldsymbol{h}_2)][\boldsymbol{W}_1^T f_1'(\boldsymbol{h}_1)]\boldsymbol{x}$$

所以當層數到100層時候，對於第一層的gradient會有100個activate function的導數相乘。

剛剛sigmoid已經說了，其導數最大為0.25。

所以$0.25^{100} \approx 0$

這就是梯度消失問題(Vanishing gradient)

如果導數值單調大於1時，就會發生**梯度爆炸問題（exploding gradient problem）**。

如何舒緩Gradient造成的問題

- 1. 重新設計網路架構: 更少的層。
- 2. Rectified Linear Activation (ReLU)
- 3. Gradient Clipping (Keras預設 clipnorm = 1.0和clipvalue = 0.5。)
- 4. Weight Regularization (L1 or L2 Regularizers)

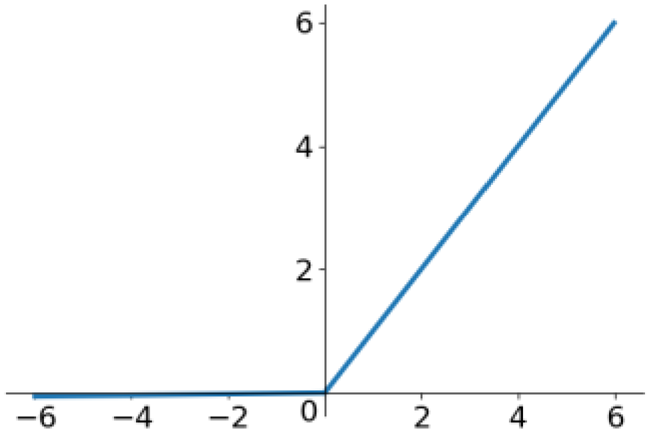
實際解決神經網路太深層的方法: Residual block。

本次介紹不考慮RNN系列的設計。

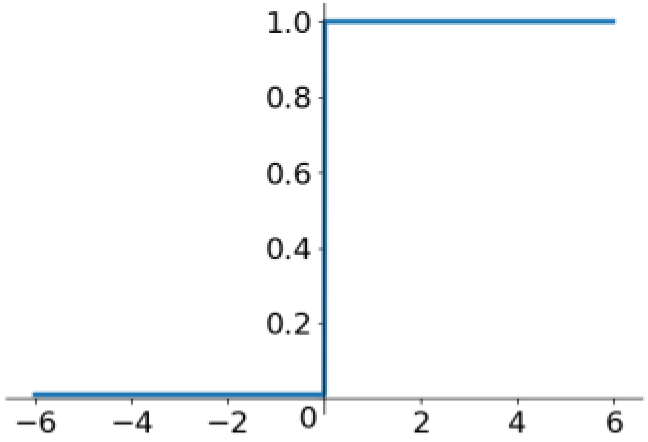
ReLu在神經網路如何舒緩Gradient的問題

- $\text{ReLU}(x) = \max(0, x) = \begin{cases} x & if\ x > 0 \\ 0 & O.W. \end{cases}$
- $\text{ReLU}(x)\text{的導數} = \begin{cases} 1 & if\ x > 0 \\ 0 & O.W. \end{cases}$

ReLU函數並不是全區間皆可微分，但是不可微分的部分可以使用Sub-gradient進行取代



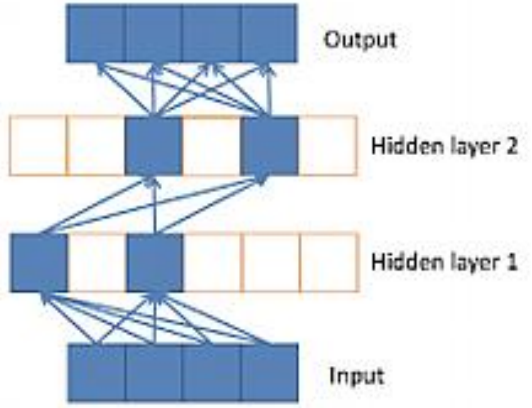
ReLU(x)



ReLU(x)導數

ReLu在神經網路如何舒緩Gradient的問題

- ReLU激勵函數會使負數部分的神經元輸出為0，可以讓網路變得更加多樣性，如同Dropout的概念，可以緩解過擬合(Over fitting)之問題。



- **衍生Dead ReLU的問題**，當某個神經元輸出為0後，就難以再度輸出值，當遇到以下兩種情形時容易導致dead ReLU發生。
- 初始化權重設定為不能被激活的數值。
- 學習率設置過大，在剛開始進行誤差反向傳遞時，容易修正權重值過大，導致權重梯度為0，神經元即再也無法被激活。

ResNet

此Residual block有兩層，第一層權重為w_1，第二層權重為w_1
第一層輸出:

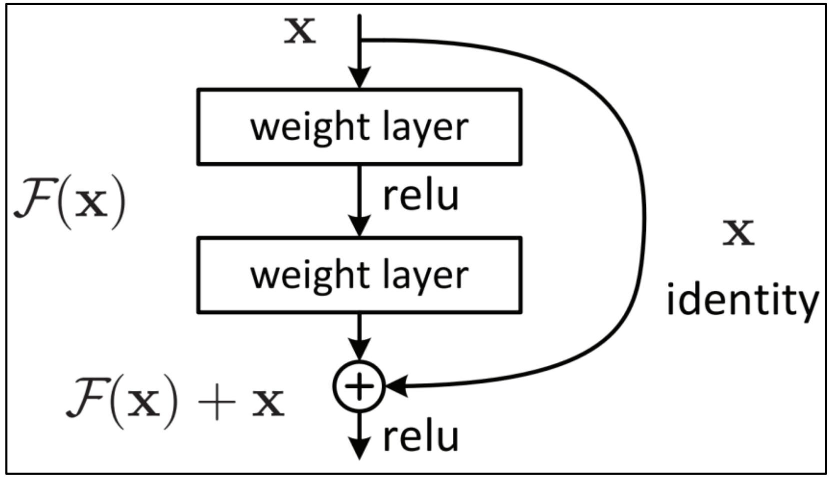
$$\mathrm{relu}(w_1 x)$$

第二層輸出:

$$F(x) = w_2 \mathrm{relu}(w_1 x)$$

最後element-add後結果:

$$H(x) = F(x) + x$$



假設神經網路太深對應用沒有幫助

1. 沒有shortcut connection (residual) ，兩層的輸出結果叫做$H(x)$

最理想狀況就是 $H(x)$ = $\boldsymbol{F(x) = x}$ (identity mapping)

2. 有shortcut connection，$H(x) = F(x) + x \rightarrow \boxed{F(x) = H(x) - x}$

最理想狀況就是 $\boldsymbol{F(x) = 0}$

2的優化找weight解會比1容易。

ResNet

假設我們的residual block只有一層權重為w

1. 沒有shortcut connection(residual)，兩層的輸出結果叫做$H(x)$

$$F(x) = wx = x$$

2. 有**shortcut connection**，$H(x) = F(x) + x \rightarrow F(x) = H(x) - x$

$$F(x) = wx = 0$$

Gradient:

1. $\frac{\partial E}{\partial w} = \frac{(\hat{y}-y)^2}{\partial w} = \frac{(wx-y)^2}{\partial w} = 2(wx-y)x$

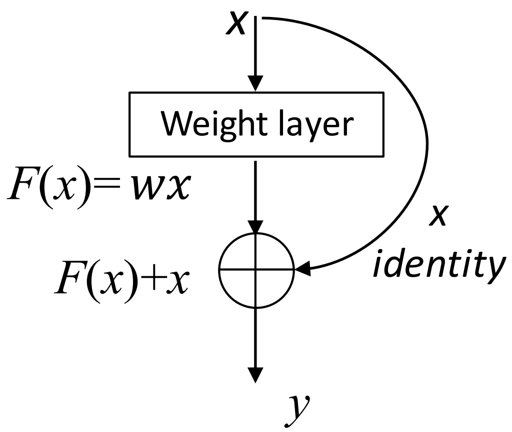
2. $\frac{\partial E}{\partial w} = \frac{(\hat{y}-y)^2}{\partial w} = \frac{(wx+x-y)^2}{\partial w} = 2(wx-y)x + \mathbf{2x^2}$

接近0

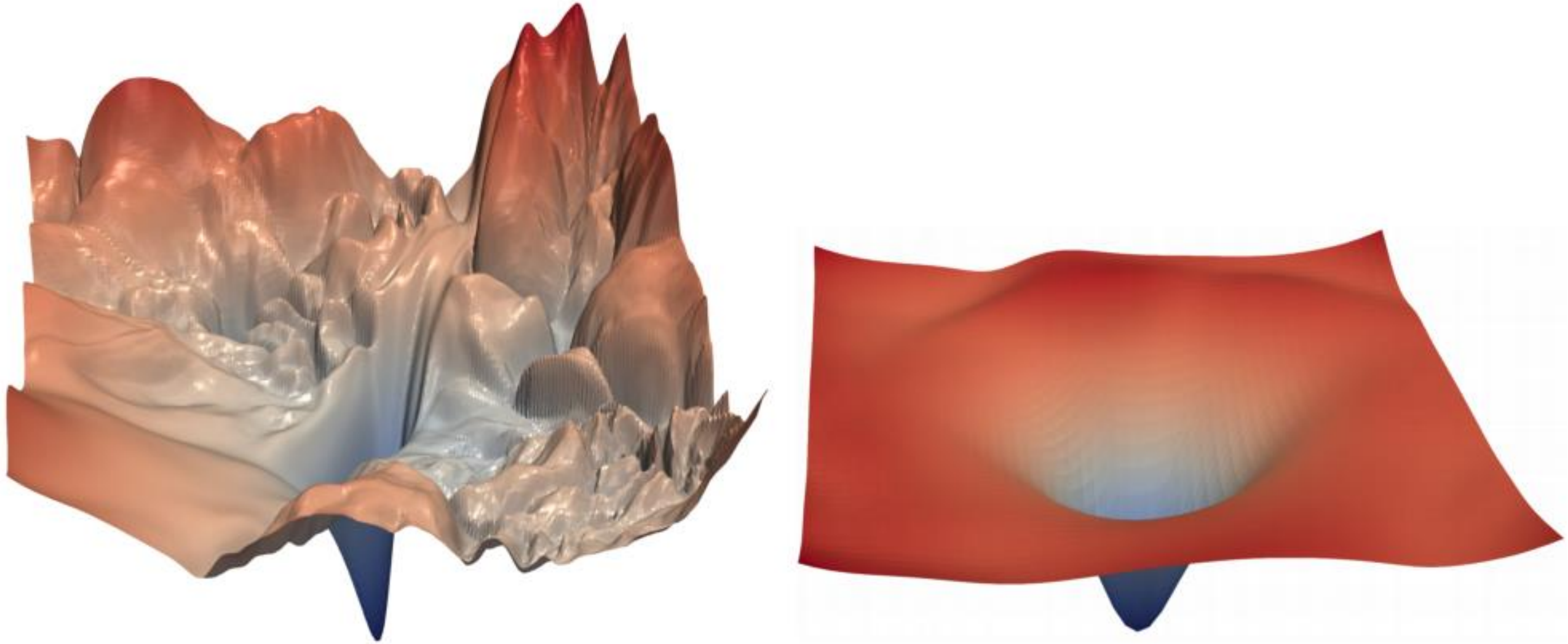
假設越後面層的Gradient很小，甚至Gradient vanish)。

$$\Delta \boldsymbol{W_1} = [(\boldsymbol{y} - \hat{\boldsymbol{y}}) f_4'(\boldsymbol{h_4})][\boldsymbol{W_3^T} f_3'(\boldsymbol{h_3})][\boldsymbol{W_2^T} f_2'(\boldsymbol{h_2})][\boldsymbol{W_1^T} f_1'(\boldsymbol{h_1})]\boldsymbol{x}$$

residual block



題外話: 有Residual block的loss space



(a) without skip connections   (b) with skip connections

Figure 1: The loss surfaces of ResNet-56 with/without skip connections. The proposed filter normalization scheme is used to enable comparisons of sharpness/flatness between the two figures.

Visualizing the Loss Landscape of Neural Nets

基礎機器學習與深度學習

- 神經網路如何利用導傳遞找解
- 神經網路太深層的問題:

梯度消失問題(Vanishing gradient)/梯度爆炸問題（exploding gradient problem）。

Activation Function為什麼要採用ReLU，而不是用Sigmoid。

Residual block克服神經網路不能太深層的問題。