第三章、神經網路

第一節、神經網路簡介

神經網路為人們在研究生物神經模型（Biological Neural Network）時，發現人腦中具有巨量平行計算及訊息分善處理的能力。人類的大腦大約由 10^{11} 個神經細胞（Nerve Cells）組成，而每個神經細胞又有 10^4 個突觸(Synapses)與其他細胞互相連結成一個非常複雜的神經網路。當人類的感官受到外界刺激經由神經細胞傳遞訊號到大腦，大腦便會下達命令傳遞至相關的受動器（Effector）做出反應（例如：手的皮膚接觸到燙的物體立即放開），這樣的過程往往需要經由反覆的訓練，才能做出適當的判斷，並且記憶於腦細胞中。如果大腦受到損害（例如中風患者），便需要藉由復健的方式，重新學習。神經網路的運作便源於此，藉由不同的演算法訓練神經網路，使得神經網路的輸出能達到我們所要求的結果。因此，可將神經網路視為一種數理迴歸方法，主要是針對輸入層與輸出層資料，以不同的演算方式，來建立兩者間映射關係的一門技術(類神經網路簡介:http://www.gct.ntou.edu.tw/Lab/aiwww/neural.html)。

神經網路主要是由許多神經元(Artificial Neurons)或節點(Nodes)所組成，神經元是生物神經元的簡單模擬，它從外界環境或其它神經元取得資訊，並以簡單的運算程序後輸出其結果到外界或其它神經元。神經網路的目的為模擬生物神經元間之運算訊息的能力，能達到生物神經系統所具有大量平行計算及分散儲存與處理的工作能力，並能透過自動學習，學習樣本之間的關係，以其非線性映射能力和無模型估計的特徵，有效解決問題。神經網路的優勢在於它是資料導向(Data-oriented)而非模式導向(Model-oriented)。所以它並不需事先在輸入與輸出參數之間假設一個固定的關係，也就是不需在輸入與輸出參數之間建構一數學模型；其次是它的資料誤差容忍度大和容易適應新的資料，所以當輸入與輸出參數之間的關係無法完全清楚時，使用神經網路比起傳統數值方法與統計迴歸方法為佳(蘇昭安，2003)。

第二節、神經網路的工作原理

由於實際上生物神經元間是交互連結成為一個複雜的網路，要利用科學的方式架構一個如此次複雜的網路連結，雖非不可能但會耗費相當多的時間與金錢在網路的架構上，因此不符合實際應用上的需求，所以需要對於實際的生物神經網路加以簡化，故神經網路的架構可被想像成包含多個節點或神經元的多層次架構。其中假設神經元為神經網路的基本處理單元，亦為組成隱藏層的主要元素。藉由隱藏層中的每一個神經元，連結輸入資料與輸出資料，而一個神經元可對應多個輸入與輸出資料，而同一隱藏層間神經元不互相連結。如圖 3-1 所示，在一個最簡單的神經網路架構中，僅包含一個輸入層、一個隱藏層以及一個輸出層，其個別定義簡介如下(蘇昭安，2003)：

(1).輸入層（Input Layer）：

「層」是由多個節點所組成，使網路內節點的連結可以簡化視為層與層之間的連結。輸入層在網路架構中為接受資料並輸入訊息之一方，通常以一層表示，其處理單元數目如圖 3-1 中之 $X_1 \cdots X_{Ni}$ 視輸入內容而定，用以表現網路的輸入變數。

(2).輸出層（Output Layer）：

在網路架構中為提供資料輸出之一方，通常以一層表示，其處理單元數目如圖 3-1 中之 $Y_1 \cdots Y_{N0}$ 視輸出的內容而定，用以表現網路的輸出變數。

(3).隱藏層（Hidden Layer）：

介於輸入層與輸出層之間，使用非線性轉換函數，其層內之節點數(圖 3-1 中之 $H_1 \cdots H_{NH}$)並無標準方法可以決定，通常需以試驗方式來決定其最佳數目，並提供神經網路處理單元間的交互作用與問題的內在架構的能力。此外，隱藏層層數多寡並無一定之標準方法，較多的隱藏層層數通常表示類神經網路所處理的問題複雜程度較高，但過多的隱藏層將導致網路在學習過程中難以收斂，一般以一至二層時具有最好的收斂性質(How many hidden layers should I use?:http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-9.html)。圖 3-1 中，所表示的為單一隱藏層的神經網路架構。

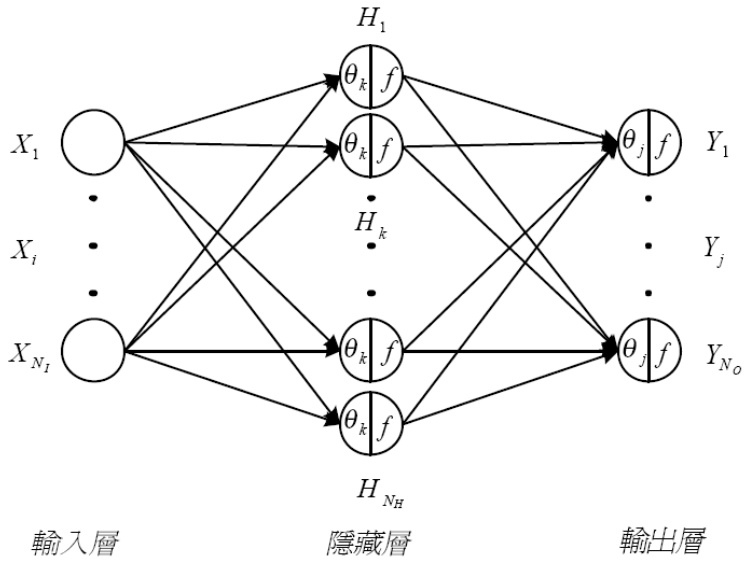


圖 3-1、神經網路架構圖

單一神經元模型原理如圖 3-2 所示，由輸入的 X 及連結權值 W，進行乘加的動作，此一步驟可藉由集成函數（Summation Function）來完成。集成函數的目的在於將前一層（輸入層）之輸出經由網路的連結權重值匯集至神經元中，通常是以函數的方式加以表示，如 $Y = f(x) = net = \sum_{i=1}^{N} W_i X_i - \theta$ 表示之，其中 W_i 為連結權值，網路學習的目的即為調整鍵結值，使其變得更大或是更小，通常由隨機的方式產生介於+1 到- 1 之間的初始值。鍵結值可視為一種加權效果，其值越大，則代表連結的神經元更容易被激發，對類神經網路的影響也更大；反之，則代表對類神經網路並無太大的影響，而太小的鍵結值通常可以移除以節省電腦計算的時間與空間，使得輸出向量達到目標，並使目標之誤差降至最小。X_i 代表輸入向量，Y_j 為輸入向量相對應的預期輸出值，而 θ 為該神經元之初始狀態，一般稱為該神經元的偏權值（Bias）或閥值（Threshold）。再經由轉換函數（Transfer Function）或作用函數（Activation Function）運算輸出，成為其他處理單元的輸入值。其功能是將神經元之輸入依各加權值作加總轉換至輸出的一種映射(Mapping)規則，亦是將非線性的影響導入網路中的一種設計。而使用不同的轉換函數（或作用函數）是不同神經網路模型間最大的差異，圖 3-3 所示為最常使用的非線性雙彎曲函數（Log-Sigmoid Function ，logsig(n)）、非線性雙彎曲正切函數（Tan-Sigmoid Function，tansig(n)）以及線性函數（Purelin Function ，purelin(n)）等。以非線性雙彎曲函數為例，可將輸入資料轉換為 0 與 1 之間的值運算輸出，而非線性雙彎曲正切函數則將輸入資料轉換為-1 與 1 之間。

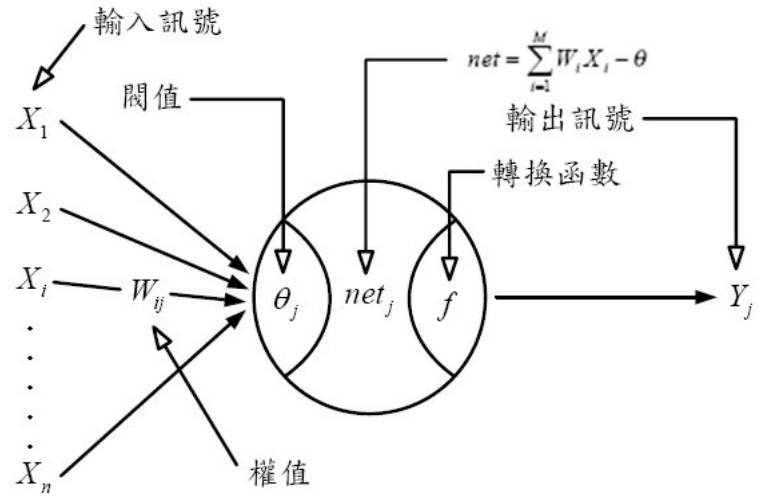


圖 3-2、單一神經元模型

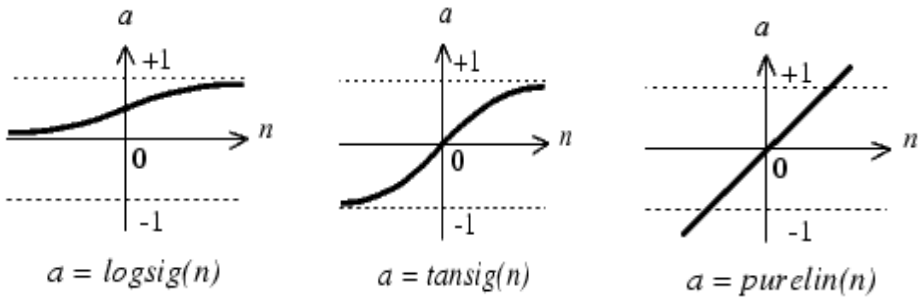


圖 3-3、常用的三種非線性轉換函數

神經網路的運算過程包含兩個運算模式，其一為學習(訓練)過程，另一個為測試(回想)過程，以下分別闡述其意義。

一、學習過程

所謂學習過程就是神經網路依事先給定的演算法，就所建立的已知訓練樣本中，學習及模擬其間輸入與輸出值之間內在的對應關係，調整連結權重值，建立兩者之間非線性映射關係，其過程如圖 3-4 所示。假設於 TWD67、TWD97 兩坐標系間存在有 n 個共同點，將 TWD67 坐標系 N、E 坐標放在輸入層，而將所對應的 TWD97 坐標系之 N 或 E 坐標分別放在輸出層，做為網路的學習目標。目的乃是希望藉由逐步修正連接神經元的加權值，使神經網路從學習的過程中讓所得輸出值與實測值之間誤差慢慢減小，並將兩者之間的對應關係儲存於隱藏層神經元中。此學習過程是以一次一個訓練樣本的方式，直到學習完所有的訓練樣本，稱之為一個循環週期（Epoch）。若輸出值與實測值之間誤差不盡理想，則反覆對於所給的已知樣本，訓練數個循環週期，直到收斂為止。

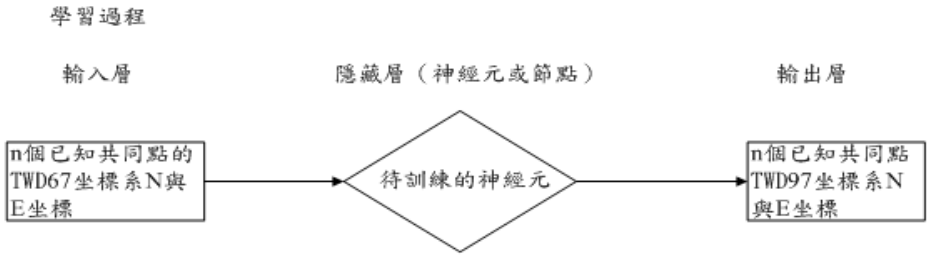


圖 3-4、學習過程示意圖

二、測試過程

神經網路經過學習之後，每個處理單元之間的連結權重，儲存著輸出值與輸入值之間的關係，因此當一個新的測試樣本（訓練樣本之外的資料，假設有 m 個兩坐標系間的共同點）輸入後，則網路會將先前所學習到之連結權重直接套用，而輸出一個與先前學習過程中所提供訓練樣本相近的輸出值如圖 3-5 所示。再將預測輸出的坐標與實際測量所得的坐標加以比較，即可評估訓練結果是否合乎預期的精度。換言之，測試過程的目的是在就所得到的預測輸出值與實際值之間接近程度來判斷神經網路學習的好壞，若無法達到既定目標，則需重新進行學習的過程。

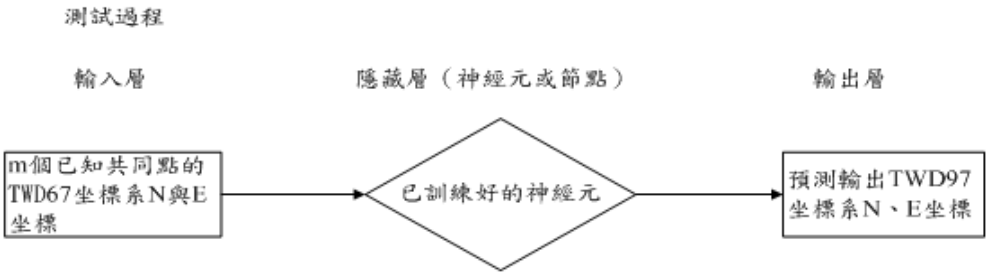


圖 3-5、測試過程示意圖

三、影響神經網路效能的因素

訓練樣本的大小、樣本所包含資料之正確性等因素，對於最後所得的結果影響相當大。如果能夠獲得的資料在控制區分佈越廣、越均勻，資料間差異性越大，所能得到的結果也就越好。除此之外，隱藏層所包含的神經元數目，也會對於最後神經網路訓練的結果有所影響。如果網路中隱藏層包含的神經元數目過多或訓練次數太多次則網路將會產生過度配適(Overfitting)的現象，如圖 3-6，網路只是去記憶所學習的資料並沒有建立非線性映射關係，所以在所有訓練點上雖配適(Curvefitting)的很好，但配適曲線在這些訓練點之間產生震盪，使得訓練點之外

的點無法分佈在良好的配適曲線上。有過度配適現象，當然也會產生未過度配適現象（Underfitting），主要產生原因為隱藏層包含的神經元數目過少或訓練次數太少，而使得訓練後神經網路之間的連結權值未調整到最佳狀態，而無法明確的描述輸入層與輸出層之間非線性映射關係，如圖 3-7。換句話說，訓練好的神經網路，不僅能夠適用於訓練點上，亦能適用於訓練點之間的區域，如圖 3-8。因為網路中採用連續值的轉換函數，所以可以得到連續值的推算輸出值。所以只要輸入的測試樣本與訓練樣本之間關係相近，則測試樣本的輸入值透過已訓練好的神經網路將會得到令人滿意的合理解。

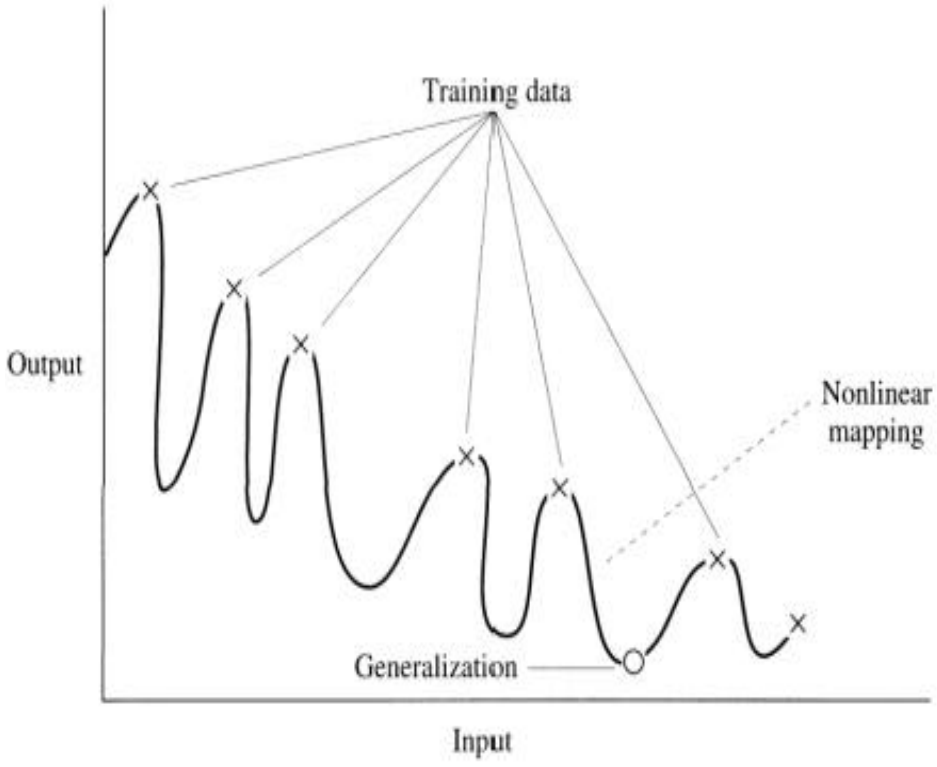


圖 3-6、產生過度配適示意圖

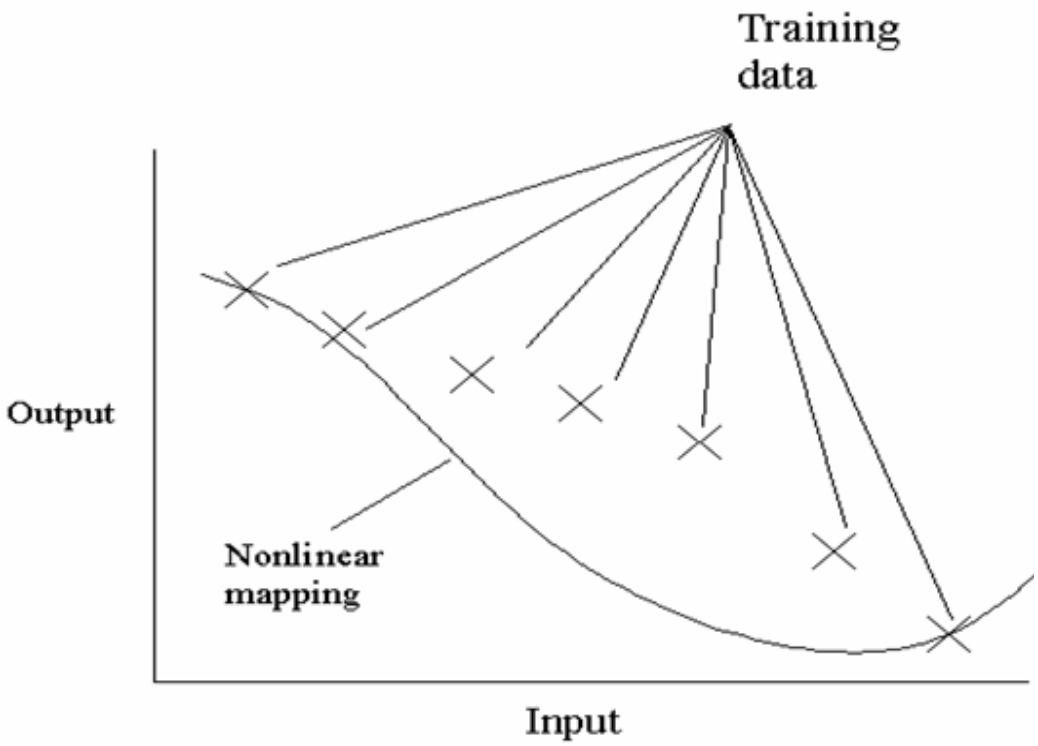


圖 3-7、Underfitting 示意圖

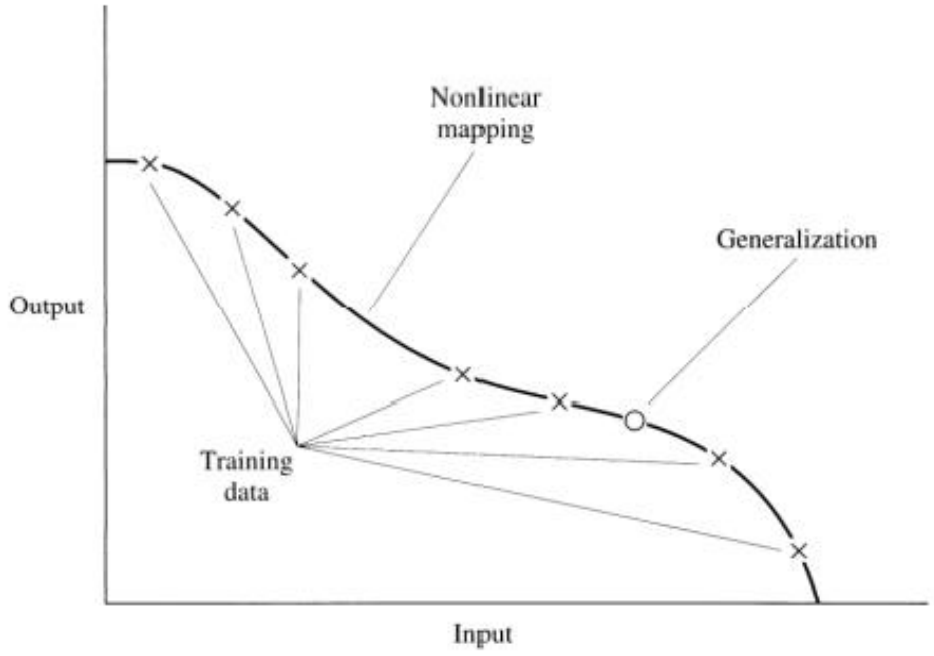


圖 3-8、良好的配適曲線示意圖

第三節、倒傳遞神經網路

倒傳遞神經網路(Back-Propagation Neural Network)是目前網路學習模式中最具代表性，且應用最廣泛的模式，最早期的應用是在經濟預測方面的問題，不過在當時並未受到重視。直到 Rumelhart 及 McCleland 發表「PARALLEL DISTRIBUTED PROCESSING: Explorations in the Microstructure of Cognition，1986」的書中詳細介紹，才廣為人知。

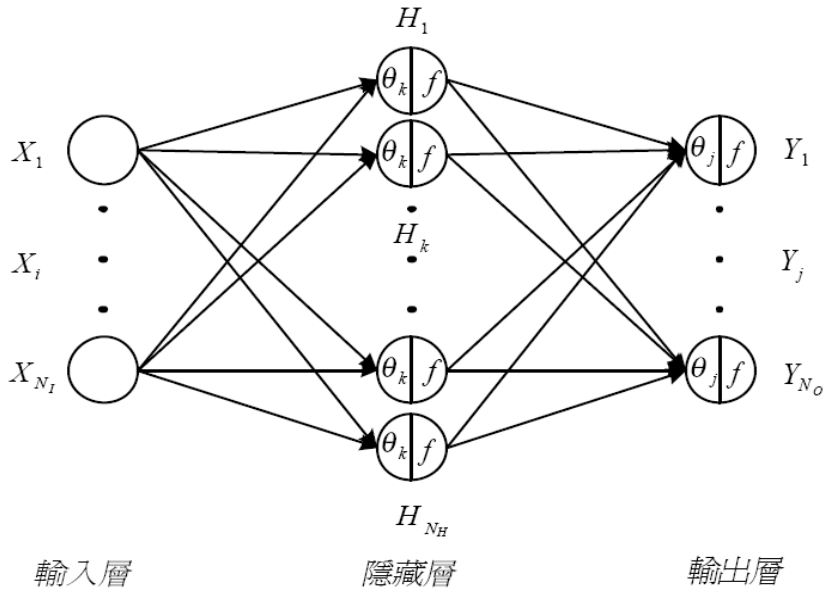
而為什麼會被稱為倒傳遞神經網路，乃是因為其採用的演算法為誤差反向傳播算法（Error Back-Propagation，簡稱 BP 算法）而得名。BP 算法結構簡單，容易應用，於人工神經網路的實際應用中，約有 80%-90%的神經網路模型是採用 BP 算法或它的其他變化形式，目前主要運用於模式辨別與分類、函數逼近、數據壓縮、影像處理及預測等領域(蘭雪梅等，2003)。

倒傳遞神經網路是將一組樣本的 I/O 問題變為一個非線性最佳化的問題，其基本原理是利用微積分中的梯度陡降法（Gradient Steepest Descent Method）計算並逐次調整網路權值，使輸出預測值與實際目標值之間的誤差最小化，以得到精確學習的一種演算方式。因此，倒傳遞神經網路常用於預測分類、診斷等領域。若將此種模式視為輸入值與輸出值之間的映射關係，則倒傳遞神經網路演算法可視為一種輸入與輸出之間的映射過程。本研究將利用此法來進行坐標轉換之間的研究探討，而之後章節中倒傳遞神經網路均簡稱為神經網路。

一、倒傳遞神經網路演算法

倒傳遞神經網路演算法的網路學習過程，包括了正向傳遞與反向傳遞兩個方向傳遞之演算，不過在同一時間，網路上僅會有一個方向傳遞之演算發生。其中正向傳遞，就是資料由輸入層經過相關權重處理後傳遞至隱藏層，透過轉換函數可以計算每一神經元所對應的輸出值及其誤差函數的過程。當正向傳遞輸出層不能得到期望的輸出值時，將推算所得之輸出值與目標輸出值，代入能量函數透過梯度陡降法得到加權值修正量，此一過程稱之為反向傳遞。透過正向及反向的來回運算，如此不斷的重複正向與反向傳遞運算藉以產生一組最佳之權重值，而網路的測試過程僅透過正向傳遞利用最佳權重值產生輸出值。以圖 3-9 來說明，正向傳遞是從輸入層開始，一層一層向前傳遞並計算各層神經元的輸出值。反向傳遞則是由輸出層開始，以輸出目標值為依據，將前一層的誤差向後傳遞，並以此為基礎修改鍵結值，接著計算該層的誤差再將其逐層往後傳遞，這就是為什麼此網路演算方式被稱為倒傳遞神經網路的原因了。

---------→正向傳遞



←---------反向傳遞

圖 3-9、正向、反向傳遞示意圖

以下就由圖 3-9 說明僅含有單一隱藏層的簡單倒傳遞神經網路，如何進行正向與反向傳遞的數學演算的法則。最簡單的神經網路是由一個輸入層、一個隱藏層與一個輸出層所架構。訊號經由各層間單一神經元相互連接(可參考圖 3-2 所示)，在每一個神經元中他們輸入與輸出的數學關係如下（蔡慶賢，2002、蘇昭安，2003）:

$$Y_j = f(x) = net_j = \sum_i^N W_{ij} X_i - \theta_j,\ i = 1...N，j = 1...M \qquad (3\text{-}1)$$

式中：

X_i =第 i 個輸入變數。

Y_j=第 j 個處理單元的輸出值。

$f(x)$=轉換函數，功能是將神經元之輸入依各加權值作加總轉換至輸出的一種映射(Mapping)規則，亦是將非線性的影響導入網路中的一種設計。

W_{ij}=第 i 個輸入與第 j 個神經元之連接權重值。

N =輸入變數之個數， M =神經元之個數。

θ_j =第 j 個神經元的偏權值。

假設網路輸入集合為{X}，隱藏層處理單元之輸出集合為{H}，輸出層之輸出集合為{Y}，正向傳遞演算流程如下：

(一)、利用方程式（3-1）計算隱藏層神經元的輸出值。

$$Y_k = f_1(net_k) = \sum_i^N W_{ik} X_i - \theta_k,\ i = 1...N_{INPUT},\ j = 1...N_{HIDDEN} \quad (3\text{-}2)$$

式中：

Y_k=第 k 個神經元的輸出值。

X_i=第 i 個輸入變數。

net_k = 第 k 個隱藏層神經元的加權乘積和。

W_{ik}=第 i 個輸入與第 k 個神經元之加權乘積和。

θ_k=第 k 個神經元的偏權值。

$f_1(\ )$=輸入層採用之轉換函數。

N_{INPUT} :輸入單元之個數， N_{HIDDEN} :隱藏層神經元之個數。

(二)、接著將隱藏層之輸出值當成輸出層之輸入值，再計算輸出層的輸出值。

$$Y_j = f_2(net_j) = \sum_i^N W_{ij} H_k - \theta_j,\ k = 1...N_{HIDDEN},\ j = 1...N_{OUTPUT} \quad (3\text{-}3)$$

式中：

Y_j=第 j 個神經元的推估輸出值。

net_j = 第 j 個隱藏層神經元的加權乘積和。

W_{ij}=第 i 個輸入與第 j 個神經元之加權乘積和。

θ_j=第 j 個神經元的偏權值。

$f_2(\ )$=輸出層採用之轉換函數。

N_{HIDDEN} ：隱藏層神經元之個數， N_{OUTPUT} ：輸出單元之個數。

由於倒傳遞類神經網路屬於監督式學習(Supervised Learning)網路，監督式學習主旨乃在於降低網路輸出值與實際值之間的差異量，因此當由正向傳遞網路計算所得之推算輸出值與學習之目標輸出值相差太大時，此時網路會自動反向傳遞以計算修正連接權重值及偏權值。如何比較推算輸出值及目標輸出值之間差異，通常採用一個能量函數(簡稱誤差平方函數)來表示資料間的差異量，如下式：

$$E = \frac{1}{2}\sum_j^M (T_j - Y_j)^2 \quad (3\text{-}4)$$

式中：

T_j=第 j 個輸出層之目標輸出值。

Y_j=第 j 個輸出層之推算輸出值。

M =神經元之個數。

(3-4)式中常數二分之一，單純只是為了能量函數微分後推導的方便性所假設。當 E 值大於某指定之誤差容忍值(Tolerance)時，反向倒傳遞神經網路演算則啟動。以梯度陡降法方式來計算權重修正量，使得誤差函數往最大梯度方向下降，其演算方式如下:

$$\Delta W = -\eta \cdot \frac{\partial E}{\partial W} \tag{3-5}$$

式中：

W =各層神經元間的連結加權值。

∂E =為各層神經元間的連結權重值之修正量。

η =學習速率(Learning Rate)，控制連結權重值修正量的幅度，修正誤差函數使其最小化。一般而言太大或太小的學習速率均會導致網路收斂不易。

而網路反向傳遞演算流程如下：

1. 計算輸出層差距量與隱藏層的差距量

計算輸出層差距量的公式推導如下：

我們可以利用微積分中的鏈鎖律(Chain Rule)，將(3-5)式中 $\frac{\partial E}{\partial W}$ 項(誤差函數對網路各層之間連結權重值的偏微分)化解為：

$$\frac{\partial E}{\partial W_{jk}} = \frac{\partial E}{\partial Y_j} \cdot \frac{\partial Y_j}{\partial net_j} \cdot \frac{\partial net_j}{\partial W_{jk}} \tag{3-6}$$

$$\frac{\partial E}{\partial Y_j} = \frac{\partial}{\partial Y_j}\left[\frac{1}{2}\sum_j^M \left(T_j - Y_j\right)^2\right] = -\left(T_j - Y_j\right) \tag{3-7}$$

$$\frac{\partial Y_j}{\partial net_j} = \frac{\partial}{\partial net_j} f\left(net_j\right) = f\left(net_j\right) \cdot \left[1 - f\left(net_j\right)\right] = Y_j \cdot \left(1 - Y_j\right) \tag{3-8}$$

$$\frac{\partial net_j}{\partial W_{jk}} = \frac{\partial}{\partial W_{jk}}\left(\sum_k^{N_{HIDDEN}} W_{jk} H_k - \theta_j\right) = H_k \tag{3-9}$$

將(3-7)、(3-8)、(3-9)式，代入(3-6)式後，可得：

$$\frac{\partial E}{\partial W_{jk}} = \left(T_j - Y_j\right)\cdot Y_j\left(1-Y_j\right)\cdot H_k \tag{3-10}$$

其次，定義 δ_j 輸出層第 j 個輸出單元的差距量：

$$\delta_j = \left(Y_j - T_j\right)\cdot Y_j\left(1-Y_j\right) \tag{3-11}$$

同理，計算隱藏層差距量的公式推導如下：

誤差函數對第 i 個輸入單元與第 k 個隱藏層神經元間的連結權重值的偏微分可改寫為：

$$\frac{\partial E}{\partial W_{ki}} = \frac{\partial E}{\partial H_k}\cdot\frac{\partial H_k}{\partial net_k}\cdot\frac{\partial net_k}{\partial W_{ki}} \tag{3-12}$$

再做一次鏈鎖律可得：

$$\frac{\partial E}{\partial W_{ki}} = \left(\sum_j^M \frac{\partial E}{\partial Y_j}\cdot\frac{\partial Y_j}{\partial net_j}\cdot\frac{\partial net_j}{\partial H_k}\right)\cdot\frac{\partial H_k}{\partial net_k}\cdot\frac{\partial net_k}{\partial W_{ki}} \tag{3-13}$$

式中：

$$\frac{\partial net_j}{\partial H_k} = \frac{\partial}{\partial H_k}\left(\sum_k^{N_{HIDDEN}} W_{jk}H_k - \theta_j\right) = W_{jk} \tag{3-14}$$

$$\frac{\partial H_k}{\partial net_k} = \frac{\partial}{\partial net_k} f\left(net_k\right) = f\left(net_k\right)\cdot\left[1 - f\left(net_k\right)\right] = H_k\cdot\left(1-H_k\right) \tag{3-15}$$

$$\frac{\partial net_k}{\partial W_{ki}} = \frac{\partial}{\partial W_{ki}}\left(\sum_i^N W_{ki}X_i - \theta_k\right) = X_i \tag{3-16}$$

將(3-2)、(3-3) 式代入(3-16)後式，可得：

$$\begin{aligned}\frac{\partial E}{\partial W_{ki}} &= \left[\sum_j^{N_{OUTPUT}}\left(T_j - Y_j\right)\cdot Y_j\left(1-Y_j\right)\cdot W_{jk}\right]\cdot H_k\cdot\left(1-H_k\right)\cdot X_i \\ &= \left[\sum_j^{N_{OUTPUT}}\delta_j\cdot W_{jk}\right]\cdot H_k\cdot\left(1-H_k\right)\cdot X_i\end{aligned} \tag{3-17}$$

定義 Δk 為隱藏層第 k 個神經元的差距量:

$$\Delta k = \left[\sum_j^{N_{OUTPUT}}\delta_j\cdot W_{jk}\right]\cdot H_k\cdot\left(1-H_k\right) \tag{3-18}$$

其中 $\sum_j^{N_{OUTPUT}}\delta_j\cdot W_{jk}$ 項，代表輸出層差距量的加權乘積和。

所以隱藏層差距量的計算與輸出層的差距量有關，這種輸出層的誤差倒傳遞到隱藏層來計算其誤差量的現象，就是此網路之所以有倒傳遞此一名稱的由來。

2. 計算各層間的加權值修正量與偏權值修正量

由(3-5)、(3-10)、(3-11)式可以得到隱藏層與輸出層間的權重值修正量如下：

$$\Delta W_{jk} = -\eta \cdot \frac{\partial E}{\partial W_{jk}} = \eta \cdot \delta_j \cdot H_k \tag{3-19}$$

同理，可知輸出單元的偏權值修正量如下：

$$\Delta \theta_j = -\eta \frac{\partial E}{\partial \theta_j} = -\eta \cdot \delta_j \tag{3-20}$$

由式(3-5)、(3-17)、(3-18)可以得到輸入層與隱藏層間的權重值修正量如下：

$$\Delta W_{ki} = -\eta \cdot \frac{\partial E}{\partial W_{ki}} = \eta \cdot \Delta k \cdot X_i \tag{3-21}$$

同理，可知隱藏層神經元的偏權值修正量如下：

$$\Delta \theta_k = -\eta \frac{\partial E}{\partial \theta_k} = -\eta \cdot \Delta k \tag{3-22}$$

3. 更新各層間的權重值與偏權值

(1) 更新隱藏層與輸出層間的權重值及偏權值之步驟如下：

$$W_{jk(n)} = W_{jk(n-1)} + \Delta W_{jk(n)} \tag{3-23}$$

$$\theta_{j(n)} = \theta_{j(n-1)} + \Delta \theta_{j(n)} \tag{3-24}$$

(2) 更新輸入層與隱藏層間的權重值及偏權值如下：

$$W_{ki(n)} = W_{ki(n-1)} + \Delta W_{ki(n)} \tag{3-25}$$

$$\theta_{k(n)} = \theta_{k(n-1)} + \Delta \theta_{k(n)} \tag{3-26}$$

其中$W_{(n)}$與$W_{(n-1)}$分別表示權重值第n次及第$n-1$次的值。而$\theta_{(n)}$和$\theta_{(n-1)}$分別表示偏權值第n次及$n-1$第次的值。之後重覆運算上述網路正反向傳遞運算，直到E值小於設定之誤差容忍值為止，則訓練完成。

二、倒傳遞神經網路參數的決定

不管是倒傳遞神經網路或其他類型的神經網路演算模式，事先需決定許多重要的參數，網路才能夠有效運作及演算，以下分別說明各項網路參數的意義：

（一）、隱藏層層數

先前說明過，最簡單的神經網路架構包含三層，也就是一層輸入層、一層隱藏層與一層輸出層。而其中隱藏層的層數可以有兩層或三層，一般來說隱藏層層數將會影響最後預測的效果。一般來說一或二層的隱藏層在各方面的表現均相當類似，並具有較佳的收斂效果，而多於兩層的話將不利於收斂且誤差較大(Villiers and Barnard，2002)。通常隱藏層數目過多將會導致計算時間過長及收斂效果變

差。惟若選擇使用兩層隱藏層則容易落入局部最小值(Local Minimum)，使得收斂時間加長。所謂局部最小值乃是網路學習過程由於先前所定義能量函數(誤差函數)的空間分佈特性，有時導致最佳之連結權重值，產生於網路誤差收斂在一局部最小值而非整體最小值(Global Minimum)。

假設能量函數的空間分佈是理想拋物碗面(Jacek，1992)，如圖 3-10，網路誤差將收斂至一整體最小值，不過一般實際上的非線性問題其能量函數的空間分佈皆會有局部最小值的問題，取圖 3-10 之剖面圖，實際上圖形是類似於圖 3-11，為凹凸不平的不規則曲面。假設神經網路從 P 點開始以梯度陡降法方式尋求最佳的權重值時，由圖形分佈的情況可知，將收斂於點 Ml 的位置，而不會繼續下降收斂至 Mg 的位置。而實際上，以圖形分佈可知整體的最小值乃是位於 Mg 而非 Ml。一般而言，網路最後即使產生可以接受的合理解，也不保證網路誤差就是收斂於整體最小值，不過因為產生的是合理解，所以雖非整體最小值或甚至根本還不到局部最小值也都無所謂；然而如果網路誤差收斂因為落入局部最小值，導致網路不易產生可以接受的合理解，則改善的措施通常是改變下面將介紹的學習速率或隱藏層神經元數目，有時甚至會調整初始連接權重值，來避免此一現象的產生。

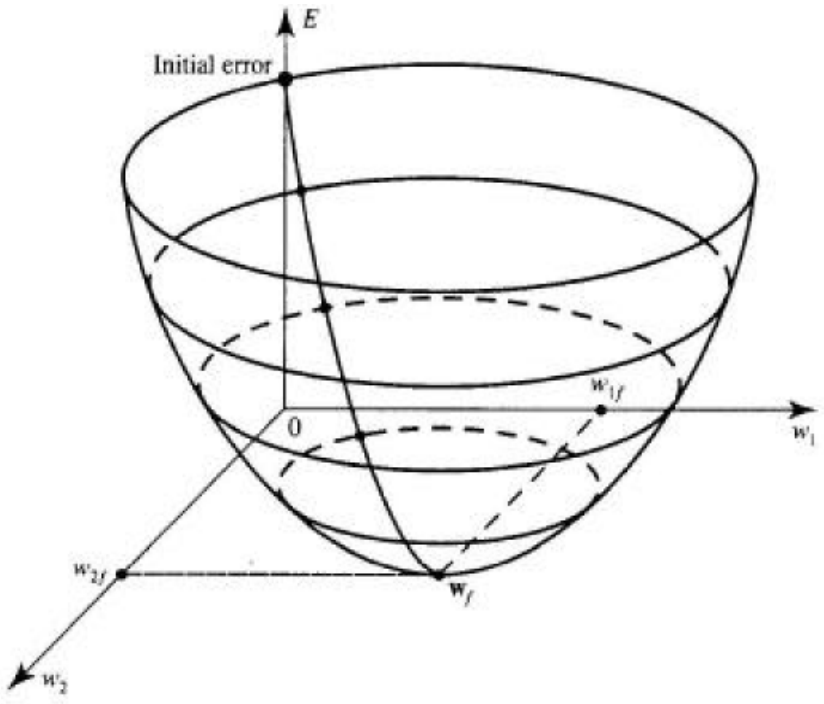


圖 3-10、理想能量函數空間分佈示意圖

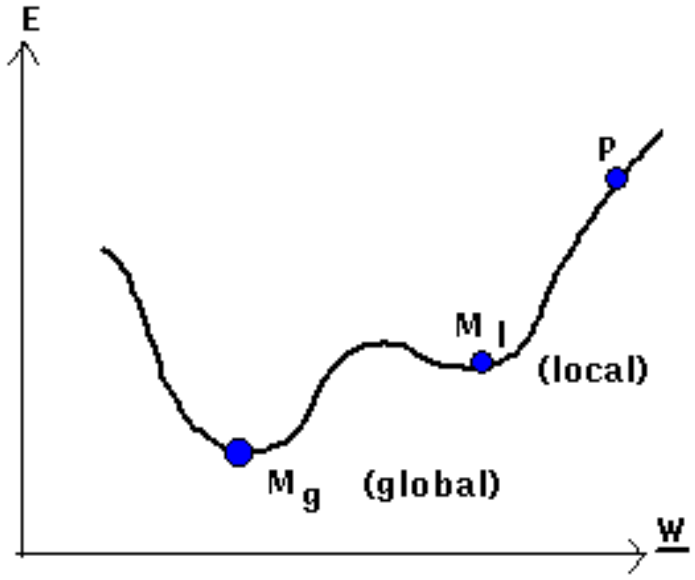


圖 3-11、實際能量函數空間分佈示意圖

（二）、訓練函數

不同的訓練函數有其基本上的定義與應用範圍，最主要的目的是利用不同的演算方式加快神經網路收斂速度以及提升精度，使得誤差函數能夠達到最小化；不同的訓練函數在使用上，僅有採用演算法與使用時機上的差異，而無彼此間好壞、優劣的區別，主要取決的因素為使用者考量本身所擁有的資料特性、本身的喜好以及研究方向而加以選擇不同的訓練函數。MATLAB7.0 中的神經網路工具箱，提供相當多種類的訓練函數可供選擇，詳細資料可參考 Demuth, and Beale(2002)所撰寫之 User's Guide of Neural Network Toolbox For Use with MATLAB。

（三）、隱藏層神經元數目

一般來說，要選取多少個隱藏層所包含的神經元數目並無一定標準。在對於樣本數目固定的情況下，如果隱藏層神經元數目過少將無法有效模擬輸入與輸出間的正確關係，過多則會產生之前討論過的過度配適現象而無法適用於所感興趣的問題領域。根據參考國內外文獻討論的結果，於樣本數比較多的情況下隱藏層神經元數目選擇 35 即可有效的描述輸入與輸出間的正確關係(Piroska Zaletnyik，1999)，但確切數目多少仍須配合實際樣本數目、實驗區域的大小、預估訓練所需耗費的時間，始能決定。因此本研究對於選取確切的隱藏層神經元數目，將利用試誤法來探討最佳的數目為何，以求出最佳的模擬值。

（四）、訓練次數

指的是終止循環次數，為神經網路達到預先設定的停止訓練目標時之循環次數，以 Epoch 為單位。若不固定終止循環次數，而讓程式自行收斂，其結果標準偏差不僅較趨於一致，且可有效避免訓練失敗的情形發生，但是要達到收斂條件所需時間較長，因此確切的訓練次數，需要依據不同的訓練樣本數、資料間實際

差異情形以及不同的實驗區，以試誤法的方式配合監控曲線圖實際跳動情形，以決定最佳的訓練次數或終止循環次數。

（五）、訓練比例

訓練比例是指從原始訓練樣本中取樣的間距。這是因為我們所得到的訓練樣本，不一定都是小樣本，除了為節省資料處理上所需時間，要對於原始資料進行取樣。當訓練樣本數夠大的話，我們也需希望將訓練樣本之中一部份選出來當測試樣本，以檢驗訓練完畢的神經網路是否能夠達到我們所要求的精度。理論上來說，取樣的間距越密，所獲得之結果，精度也就越高。

（六）、學習因子（η）為與慣性項（α）：

學習因子（η），或稱之為神經網路的學習速率（Learning Rate），於神經網路的訓練過程中是一個非常重要的參數。過大或過小的學習因子將直接影響網路的收斂速度，選擇較大的學習因子，則可以快速的逼近能量函數（誤差函數）的最小值，但過大的η將導致網路權重值修正過量而產生震盪的現象，如圖 3-12 所示；太小的η值則會使得收斂速度變慢，增加計算上所需時間，且容易掉入局部最小值而難以跳出如圖 3-13 所示。而慣性項（α）為學習因子的重要輔助工具，其作用為改善收斂過程中的震盪情形。然而這兩個參數的決定往往需要利用不斷的嘗試，並配合不同大小的樣本數，來加以取決。

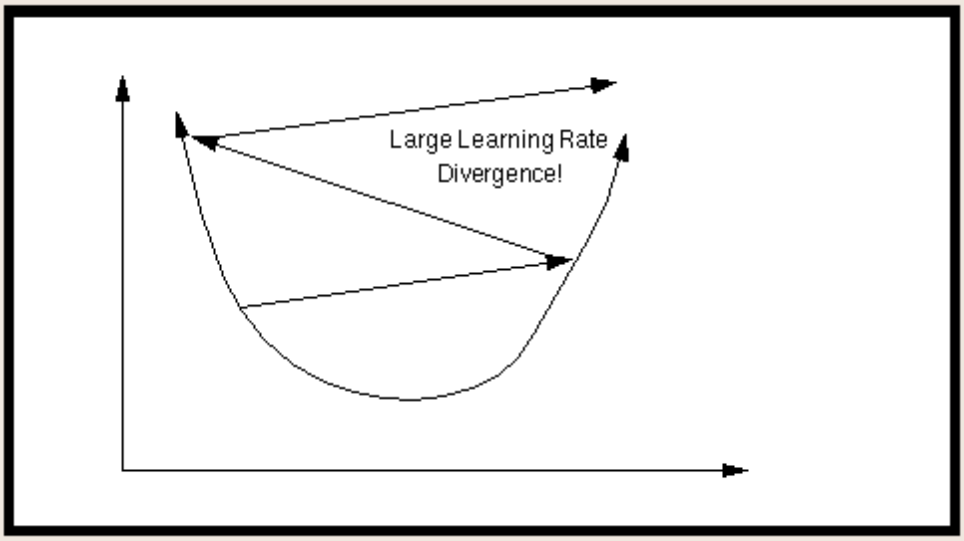


圖 3-12、快的學習速率示意圖

（摘錄自 http://www.willamette.edu/~gorr/classes/cs449/precond.html）

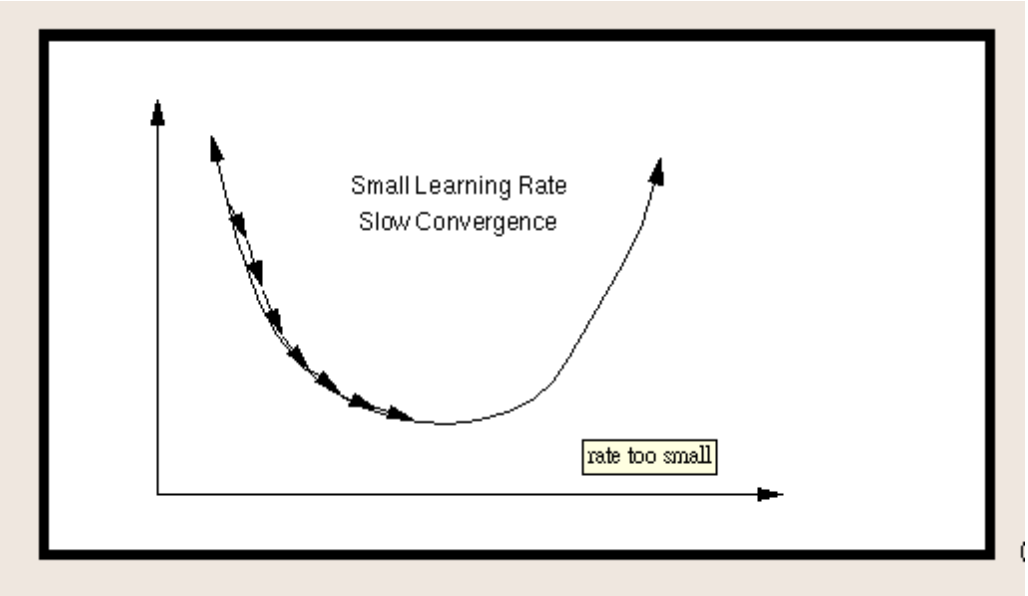


圖 3-13、慢的學習速率示意圖

（摘錄自 http://www.willamette.edu/~gorr/classes/cs449/precond.html）

三、神經網路使用上之限制

任何一種方法，都有其基本上的限制，神經網路也不例外。其主要的限制有二，分述如下：

（一）、參數的決定：

利用神經網路方式進行坐標轉換時，事先需決定許多重要的參數，網路才能夠有效運作及演算。而如何針對不同的輸入層與輸出層資料，進行最佳的擬合，是以不斷重複試驗的方式來進行，以兼顧精度與訓練所需的時間。譬如說改變隱藏層神經元數目、選擇不同訓練函數、訓練比例與次數，對於最後的結果與訓練所需時間都有很大的影響，而如何決定每一個項目的最佳參數，是以試誤法的方式一一進行，逐步改正不同參數對於結果的影響。因此，對於每項參數進行最佳化是相當耗費時間且複雜的過程。且仍有平移、旋轉及尺度等未知參數無法個別表示之缺點，也因此無法較為明確的對坐標轉換的幾何意義加以說明。

（二）、黑盒子（Black Box）

在利用神經網路方式進行坐標轉換時，所使用軟體為 Math Works 公司開發的一種科技應用軟體，MATLAB7.0 中的神經網路工具箱來訓練 BP 神經網路，建立輸入層與輸出層資料的映射關係。然而，此一映射關係建立之後，由於所使用軟體的限制，僅存在於訓練完畢當時，軟體的暫存記憶體內，而無法利用數學函數（Function）關係來描述輸入層與輸出層資料之間的映射關係，此問題稱之為黑盒子，如圖 3-14 所示。換言之，當我們重新啟動電腦，以新的輸入層與輸出層資料來訓練神經網路，每次訓練的結果都會有和前一次的訓練結果稍有不同。解決的方式通常為重複執行同一組資料，並比較結果後，經評估比較結果後

選擇一組精度最高的資料，再將這一組資料內插成為規則網格。當此網格轉換模式建立之後，後續外業測量時，可即時的將觀測資料輸入至此模式當中，即時的進行坐標轉換，並實地進行檢核，以確保外業成果合乎規範，以確保土地所有權人權益，並避免發生測量錯誤而衍生界址糾紛之情事。

本研究主要目標之一，即是建構如上述所說的網格式即時地籍坐標轉換的模式，除了能夠有效解決所使用軟體先天上會產生黑盒子的限制，亦能夠提供地政機關執行戶地測量、鑑界等外業測量業務時可直接作地籍坐標轉換。

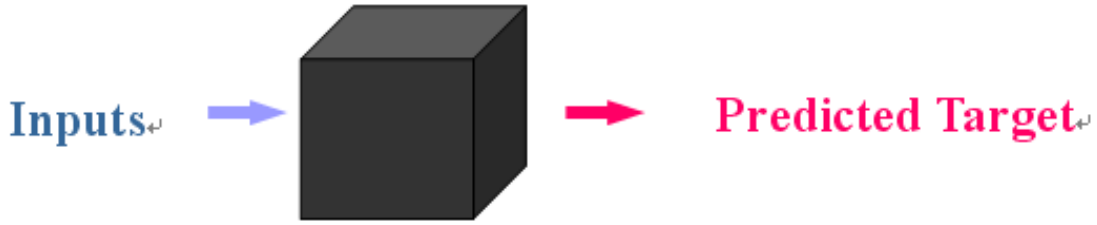


圖 3-14、黑盒子示意圖