平行式類神經網路電力負載預測系統模式化之研究

研究生：黃敬淳　　指導教授：張炳騰 博士
彭泉 博士

東海大學工業工程與經營資訊研究所

摘要

電能是人類生活中重要的能源之一，更是國家經濟發展之重要基石，而電力負載預測（Power Load Forecasting）不論是在電力輸送作業、儲存能源調配、線上調控或緊急事故處理中均扮演著極為重要的角色。因此，許多專家學者曾使用不同的方法，以致力於電力負載預測此一議題。一般而言，如時間序列法（Time Series Analysis），以及近年來常用的專家系統（Expert Systems）、灰色理論（Grey System Theory）和類神經網路（Artificial Neural Networks）。其中，類神經網路更是被廣泛應用。

然而，由於進行電力負載預測時，考慮的相關影響變數均非常之多，而傳統的類神經網路雖然能利用相關影響變數建立類神經網路訓練資料，卻不能對各個相關影響變數間的關係進行探討並調整之。因此，本研究的目的為建構一平行式類神經網路，並將其應用於電力負載預測，以求更精確的電力負載預測結果，並分別與傳統倒傳遞網路（Back-Propagation Network, BP）、徑向基底網路（Radial Basis Function Network, RBFN）、一般迴歸神經網路（General Regression Neural Network, GRNN）進行比較。由實驗結果得知，平行式類神經網路架構之絕對平均誤差較其他三者方法為低，因此利用平行式類神經網路架構進行電力負載預測系統較其他三者方法更為精確。

關鍵字：電力負載預測（Power Load Forecasting）、平行式類神經網路(Parallel Neural networks) 、倒傳遞網路（Back-Propagation Network, BP）

Study of Power Load Forecasting System Modeling With a Parallel Neural Network

Student：Ching-Chun Huang　　Advisor：Dr. Ping-Teng Chang

Dr. Chyuan Peng

Institute of Industrial Engineering & Enterprise Information

Tunghai University

ABSTRACT

The electric power is one of the important energy that is the foundation for the economic development of a country. For electric power business, accurate load forecasting plays an important role in economic scheduling of generating capacity, scheduling of fuel purchases, planning of energy transactions, and dispatching of generation units. Many methods have been developed for power load forecasting. Consequently, time series analysis, expert systems, grey system theory and artificial neural networks have been proposed for power load forecasting. Especially, artificial neural networks have been used widely.

Not only too many input variables of power load forecasting need to be considered, but the traditional back-propagation network can't adjust the input variables mutually with their relations, in this research, we developed a parallel neural network to forecast the power load. Then, we compared the actual power load with the results of load forecasting of the parallel neural network model, the back-propagation network model, the radial basis function network model and the general regression neural network model. According to the forecasting results, our parallel neural network models is more accurately than other methods, the mean absolute percentage error (MAPE) also reveals that our parallel neural network models perform better than other method.

Keywords：Power Load Forecasting、Parallel Neural networks、Back-Propagation Network

致謝

承蒙恩師 張炳騰教授這兩年來對學生的研究工作及生活悉心教誨與指導，學生不僅順利完成本文且對處事態度等各方面都有長足的進步，謹此致上最高之敬意與謝忱。

論文初稿承蒙 彭泉博士、 洪堯勳博士、 白炳豐教授與 陳琨太教授於口試期間不吝指正及提供寶貴意見，特此致謝。

研究期間得蒙博士班范宏武、洪國禎、林國平、林志昇學長特別撥允指導，同窗摯友俊嘉、小強、文偉、瑋珊於學業及生活上的切磋與勉勵，學弟龍廷、國丞、鼎翰、禎祥、香君於論文寫作期間的相助與生活上的相伴。

最後，謹以本文獻給最親愛的家人與女友，尤其是辛苦栽培我的父母親，感謝你們無怨無悔的照顧、支持與鼓勵。

黃敬淳 謹誌於

東海大學工業工程與經營資訊研究所

民國九十四年六月

目錄

圖目錄

表目錄

第一章　緒論

1.1 研究動機與背景

電能是人類生活中重要的能源之一，隨著國內經濟蓬勃發展，國民崇尚優質生活之際，不僅工業用電需求日益殷切，民生用電亦大幅增加。倘若電力供應不足，將造成民生及工業的停擺，嚴重影響國家經濟發展的腳步。因此，電業必須不斷地擴建電廠，開發電源以滿足用戶的需求。然而由於電力負載隨著用戶需求情形隨時變化，而電能係屬於無形的產品，具有生產與消費同時發生，又不能大量存儲之特性，因此欲有效達成電力系統之供需平衡，則必須仰賴於精確的電力負載預測。

因此對於電力事業而言，必須掌握用電負載的趨勢，以求電力系統的穩定與可靠，更經濟地開發電能提供高品質的電力，和有效的調度供電機組。至於如何掌握未來用電趨勢，以求有效達成電力系統之供需平衡，電力負載預測不論是在電力輸送作業、儲存能源調配、線上調控或緊急事故處理中均扮演著極為重要的角色。故精確的負載預測，能提供系統較正確的排程與規劃，並降低運轉成本及提高供電可靠度，避免造成限電危機或資源的閒置與浪費。

1.2 研究目的

電力系統之負載預測中若以預測週期長短區分，可概分為長期、中期及短期負載預測。一般而言，五到十年的長期負載預測主要提供電力公司發電量的長期規劃，以做為電廠增設及電力事業未來經營策略的依據；數月到五年間的中期負載預測則用於檢修排程及電力分配，以使既定之發電量能作充份利用；而數小時到數週之短期負載預測則提供機組協調、發電預定、經濟調度及負載管理等之重要依據。

由於電力負載預測是如此的重要，因此許多專家學者曾使用不同的方法，以致力於此一議題的研究。一般而言，可區分為傳統統計上的時間序列法(Time Series Analysis)，以及近年來常用的專家系統（Expert Systems）、灰色理論（Grey System Theory）和類神經網路（Artificial Neural Networks）。其中，由於類神經網路最大的優點在於它不需要複雜的數學運算，只需要適當的輸入與輸出樣本以供學習，即可做出精確的預測；此外，類神經網路對於變數間非線性關係的掌握亦較其他模式靈敏，故近年來類神經網路

被廣用於電力負載預測議題上。

自 90 年代開始即有大量以類神經網路預測電力負載之研究，而在這些研究當中，大多是以相關影響變因建立類神經網路訓練資料，以預測每天的尖峰用電負載或總用電負載量，以確保高品質的電力負載與快速調度供電機組的能力；其中，以倒傳遞網路（back-propagation network）更是被廣泛應用。

然而，傳統的類神經網路雖然能利用影響電力負載預測的相關變數建立訓練資料，卻不能針對各個相關變數間的關係進行探討並調整之。因此，本研究的目的為針對電力負載預測之議題，架構相對應之平行式類神經網路電力負載預測系統，以改善利用傳統倒傳遞網路進行電力負載預測之缺失，並求得更精確的電力負載預測結果，提供電力系統較正確的排程與規劃，並降低運轉成本及提高供電可靠度，避免造成限電危機或資源的閒置與浪費。此外，研究成果並可提供應用類神經網路系統於其他預測的研究人員進行參考。

1.3 研究方法與步驟

本研究主要是透過電腦實驗的方式進行研究，研究的問題為電力負載預測問題，並根據本研究所提的平行式類神經網路進行系統模式化，最後再依據實驗數據與結果作分析討論。本研究之研究步驟如下：

1. 電力負載預測問題之分析

首先，收集相關電力系統資料，並針對電力負載預測，分析出真正影響預測結果之相關影響變數。

2. 平行式類神經網路系統之建立

接著，再將每一相關影響變數對應到每一個輸入神經元，並根據輸入神經元與輸入神經元的關係，將每個輸入神經元另外加以互相連結，已達到變數與變數之間相關性的調整。

3. 訓練資料的建立

類神經網路的性能與訓練資料的建立有極大的關係，必須先利用歷史數據建立網路的訓練資料。而訓練資料建立的優劣直接影響了預測精度。

4. 數值正規化

在由歷史數據獲得網路的訓練資料後，必須將樣本內的數據轉換成網

路的輸入與輸出訊號，以做為學習過程之輸入值。

5. 網路參數設計

在類神經網路之中有些參數，例如學習速率、隱藏層神經元數目以及隱藏層數目必須事先規劃。

6. 訓練

7. 預測

1.4 論文架構

本研究論文的內容共分為五章：第一章說明本研究之動機、目的、方法與步驟等相關內容；第二章則針對本論文內容所涉及之相關文獻加以探討，包括電力負載預測、類神經網路之相關文獻；第三章則根據本論文之目的與文獻所得之啟發，提出平行式類神經網路電力負載預測系統架構，詳細闡述系統架構中各項機制之功能設計與運作方式；第四章為論文中系統之實驗結果；第五章則根據本研究所得之結果，說明研究結論與未來發展方向。本論文的進行流程如圖 1.1。

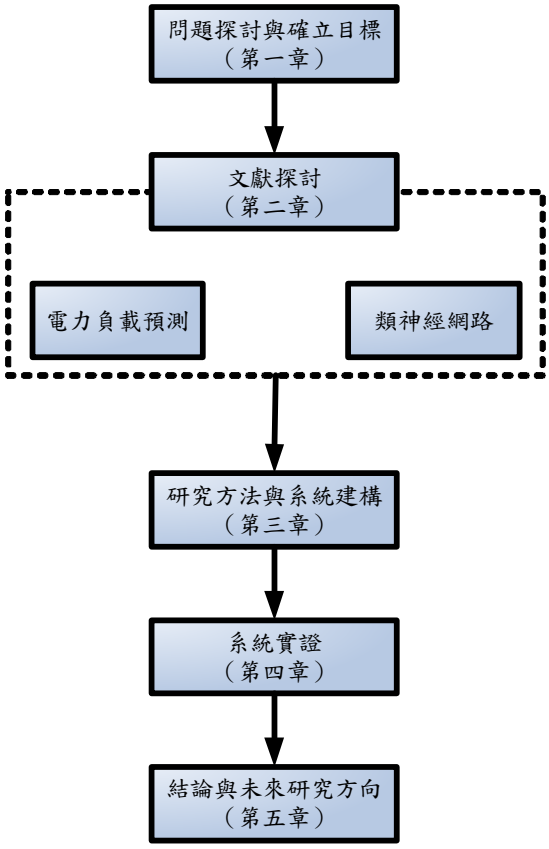

圖 1.1 研究流程圖

第二章　文獻探討

2.1 電力負載預測相關文獻探討

因應電能開發與調度所需，電力事業必須掌握用電負載的趨勢，以求電力系統的穩定與可靠。因此，如何預測未來負載的趨勢，使電能開發能與負載互相配合，已成為相關專家學者重要的課題。

預測是對某事物未來可能的趨勢，事先予與推測，而預測的方法必須考量到其準確性、即時性和實用性。在電力負載預測方面，依預測時間的長短可分為中長期預測、短期預測與即時預測。預測的方法不勝枚舉，一般而言，可區分為傳統統計上的時間序列法(Time Series Analysis)、以及近年來常用的專家系統（Expert Systems）、灰色理論（Grey System Theory）和類神經網路（Artificial Neural Networks）。各種方法都有其優缺點，現就針對上述的幾種預測方法與相關文獻作個簡單的敘述。

2.1.1 時間數列分析應用於電力負載預測文獻探討

所謂時間數列係指以時間順序型態出現之一連串觀測值，時間數列應用於預測之基礎乃是將過去資料彙整為函數型態。然而古典時間數列模式中並未包括其他外生變數，而僅以過去之資料為預測之依據。時間數列分析因僅需預測數列本身過去歷史資料即可建構相關模式，故在其他資料收集不易的情況下，此法是一個不錯的選擇。近來含外生變數之時間數列的發展亦開展時間數列模式的應用範圍。常見的時間數列模式包括有四種基本模式：AR 模式（autoregressive model）、MA 模式（moving average model）、ARMA 模式（autoregressive-moving average model）以及 ARIMA 模式（autoregressive integrated moving average model）。總體而言時間數列法對於週期性、季節性及循環性之趨勢易於掌握，且不需其他外生變數，而純粹以數列本身歷史數據做為預測之基礎，資料收集容易，成本花費低。然而，此模式較適宜用在短期預測，但此模式之操作需高度技巧與經驗，且因模式預測結果不易解釋，因而也限制了該模式之應用廣度。[1][24]

2.1.2 灰色理論應用於電力負載預測文獻探討

一般傳統預測分析，常需要大量數據或累積豐富經驗才可做出精確預估，然而 1982 年由大陸學者鄧聚龍教授所提出的灰色理論（Grey Theory），僅需由少數資料點，即可作出精確預測。灰色系統理論是一種針對在模型

不明確、訊息不完整之情況下，進行系統的關聯分析、模型建構、預測與決策的方法。其主要特色是能對系統之不確定性、多變量輸入、離散數據及數據不完整作有效的處理，並根據系統模型的發展趨勢作未來行為之預測。灰色理論利用原始序列經累加生成運算找出其規律性，即將離散不規則的原始序列，經生成處理使其呈嚴格遞增的指數規律，再據以建立灰微分方程來擬合並進行預測，因此所需數據少，也不需假設數列分佈情形。[32]

2.1.3 類神經網路應用於電力負載預測文獻探討

自 1987 年類神經網路的理論發展成熟之後，許多相關的應用與研究如雨後春筍般的紛紛出現。Lee et al.[21]於 1990 年，首先運用類神經網路於負載預測，直至今日，相關文獻也有系統的呈現出來。運用類神經網路最大的優點在於它不需要複雜的數學運算，只需要適當的輸入與輸出樣本以供學習，即可做出精確的預測。茲依預測時間的長短，區分成中長期預測、短期預測和即時預測，分述如下。

1. 中長期預測

Fung et al.[41]結合了經濟上之考量因素，如電費、國民生產毛額、與天氣變數，對香港的用電量做長期的負載預測，實驗的時間從 1970 年到 1992 年。這種建模的概念來自多重線性迴歸分析（Multiple Linear Regression）。但有別於一般的時間序列法只考慮歷史用電量，多重線性迴歸分析還得考慮經濟、天氣或電價的影響。因此他們將預測的對象細分為家庭用電、商業用電、工業用電與公共用電等方面，個別進行預測。在與多重線性迴歸分析的結果比較之後，雖然總用電預測的效果差不了多少，但在執行預測過程中類神經網路遠比多重線性迴歸分析來得方便。

Kermanshahi et al.[12]利用一個重複式的類神經網路架構從事日本九個電力單位未來 10 年的電力需求預測，其網路概念主要是認為長期負載預測中未來的電力負載與前一期的電力負載有相當密切的關係，故為在網路學習中提供這項即時訊息，便在傳統的倒傳遞網路中輸出神經元（代表未來負載預測值）與輸入層代表前期負載實績的神經元相連結，研究結果顯示平行式類神經網路預測結果較傳統類神經網路所得預測結果來得精確。

Doveh et al.[18]以星期指標、季節指標、趨勢變數、前兩天的溫度值與負載量，預測日之溫度，進行每月總用電預測；其中，季節指標先經過模

糊化的處理，再帶入類神經網路的輸入層。先收集了以色列地區 1992 年 4 月至 1995 年 3 月的資料建立訓練模式，再以 1995 年 4 月至 1997 年 3 月進行驗證。在實驗當中，由於不同的網路加權值會導致不同的預測結果，他們便建立了 10 組訓練網路，經由反覆的測試和驗證，刪除了收斂效果最差的幾組網路，以解決局部最小值（Local Minima）對預測效果的影響。

Chandrashekara et al.[11]利用倒傳遞類神經網路從事 10 年期的配電系統負載預測，並結合專家系統對配電系統進行最適規劃。在系統負載預測部份的網路架構為 5 個輸入神經元（前兩年、前五年、前十年及前十五年之負載實績），8 個隱藏層神經元及一個輸出神經元（預測年之預測負載量）。其網路的特色為將類神經網路與資料庫系統加以結合，將網入輸出之負載預測值傳送至資料庫以供下一期網路輸入值之用，這項修正可有效降低網路學習時間。

Al -Saba et al.[35]比較倒傳遞類神經網路與線性模式（AR 與 ARMA）在長期需電預測的表現，研究結果顯示類神經網路在長期需電預測的精確性較其他兩類線性模式來得佳。Darbellay et al.也獲致相同之研究結果。

2. 短期預測

Peng et al.[36]提出以類神經網路來預測一天之總用電量。其網路輸入值包括前一天之總用電量、前一天之最高與最低溫度、預測日之最高與最低溫度。他們以美國西北地區為研究對象，以冬季的第一週與第二週的星期二至星期四為訓練資料，預測第三週的星期二至星期四的用電負載，而後再以第二週與第三週的星期二至星期四的實際用電量以預測第四週的用電負載，依此預測整個冬季。雖然整體預測誤差只在 4 到 5%，但是在第五週以及第十週的星期四誤差高達 10%以上。經由分析，那四天正逢假期，因此他們提出對於假日的狀況應該另外建立預測的機制，以求更精確的結果。

Peng et al.[37]提出一改良式的倒傳遞類神經網路並將其應用在短期負載預測上，網路主要特色除了傳統倒傳遞類神經的架構外，在原輸入神經元與輸出神經元間另外加以連結，以原有的網路架構掌握輸入神經元與輸出神經元間非線性的關係，而以新加入的連結掌握輸入神經元與輸出神經元間線性的關係。研究結果發現改良型的網路在預測的表現上較傳統倒傳遞類神經網路精確。

Srinivasan et al.[16]以三層式倒傳遞網路對新加坡地區進行一天尖峰用電預測。其網路輸入值包括歷史尖峰用電量、前一天之最高與最低溫度、預測日之最高與最低溫度與百分濕度。並將預測日分成週二至週四、週一、週五、週六、以及週日和假日五種不同的時段以進行一整年的當天尖峰用電預測。經過實驗的測試，發現隱藏層到輸出層的加權值以較小的數值能得到較精確的預測結果，誤差只有 2.39%。在跟 Box-Jenkins 法與雙指數迴歸分析法比較之後，倒傳遞網路的確能得到較為精確的預測結果。

Caire et al.[27]建立三種規模的倒傳遞網路進行 1986 至 1990 年法國六個城市的當天總用電量預測，分別為最大規模、最小規模與精簡最大規模三種型態。最大規模網路的輸入值包括前 35 天之總用電量、七個星期指標、前 35 天之最高溫與最低溫、以及當天的最高溫與最低溫預測值，總輸入神經元高達 134 個之多。而最小規模網路的輸入神經元只有 18 個，包括前五天的總用電量與平均溫度、當天的平均溫度預測值以及七個星期指標。精簡最大規模網路的輸入神經元雖然跟最大規模網路數目一樣多，但針對了各輸入神經元調整適當的比重。這三種網路模型實驗結果與 ARIMA 模型比較，以精簡最大規模網路的誤差最小，最大規模網路次之，而最小規模網路與 ARIMA 模型結果最差。所以他們提出只要輸入值考慮周全，類神經網路的預測效果比 ARIMA 來的精確。

Lu et al[13]運用倒傳遞網路來預測每小時用電負載、24 小時用電負載與每日的尖峰負載。在小時預測模式中，網路輸入變數含前兩天平均溫度、前 24 小時溫度、下一小時溫度預測、小時指標、星期指標與過去數小時的用電量。24 小時用電負載的網路輸入變數含過去 24 小時的用電量、前一天平均溫、未來 24 小時溫度預測與星期指標以預測未來 24 小時之用電量。尖峰負載預測的網路輸入變數包含前一天之尖峰用電量、前一天之平均溫度、預測日當天之平均溫與最高、最低溫度。此三種用電預測模式的結果顯示了工作日的預測誤差遠比週末或假日低，原因應該是週末或假日的歷史數據較少，無法充分提供網路訓練之用。這種建模方式是 24 小時負載預測最常見之方式， 相關的研究見 Bakirtzis[7]、Khotanzad[8][10]、Kiartzis[33]、Chow[39]等文獻。

Gooi et al.[20]提出另一種 24 小時負載預測之機制。他們先以倒傳遞網路預測當天之尖峰負載與低谷負載，網路的輸入值包括了歷史尖峰負載與

低谷負載、最高溫度與最低溫度、預測日當天之最高溫度與最低溫度預測，依照週二至週四、週一、週五、週六、週日和假日六種不同時段分別預測。得到當天尖峰負載與低谷負載預測值之後，再依據一天用電分佈狀況，分析每小時常態負載與尖峰負載和低谷負載之間的關係，進行 24 小時的預測。實驗結果顯示週二至週四時段因為歷史資料較為充分，預測效果最好；假日型態由於歷史資料較少，預測效果也最差。

Voss et al.[25]針對倒傳遞網路負載預測提出改善精度的方法。在倒傳遞網路負載預測中，由於不能確定訓練出來的模式能夠符合各種情況，因此必須隨時間的增加而需重新訓練，使得訓練出來的結果才能符合最新的負載趨勢。為了減少重新訓練的次數，他們提出移動平均濾波器的觀念做為倒傳遞網路的補償。利用前幾次預測誤差的數值，補償至倒傳遞網路的輸出，得到最新的預測值。在對埃及地區進行每月尖峰預測後，結果顯示加了移動平均濾波器為補償的模式誤差值只有 1.26%，大幅改善只有倒傳遞網路預測時的誤差 3.24%。不但能夠提升預測時的精度，也能減少訓練網路的次數。

Islam et al.[34]利用倒傳遞類神經網路從事短期每月電力負載及能源消費之預測，兩類神經網路的輸入變數皆採用氣溫、降雨量輻射量等氣象資料，利用 1986 年至 1990 年的資作為網路訓練用，1991-1992 年的預測結果與實際值比較的結果發現每月電力負載結果之平均誤差為 6%；而每月能源消費結果之平均誤差為 10% 。

Chow et al.[38]時間數列預測的概念將前幾期負載實績當成網路輸入變數，另外並考慮前一期的最高、最低與平均氣溫及當期預測最高與最低氣溫等亦當成網路輸入變數，以預測未來每小時負載量。研究以香港地區歷史負載為例，研究結果發現以此方法所得預測結果誤差較以往的預測結果降低 0.9% 。

Mohamad et al.[17]結合倒傳遞類神經網路與專家系統在針對埃及 1993 年短期小時負載預測的問題上，並將預測結果與傳統多元迴歸所得結果加以比較，研究結果顯示利用此種方法之預測結果平均絕對值誤差百分比約為 2.63%較多元迴歸之 4.69%佳。

Liu et al.[23]比較模糊邏輯（Fuzzy Logic, FL）、類神經網路與自迴歸模式（Auto Regressive model, AR）這三類預測技術在短期負載預測議題上的

適用性，研究結果發現以模糊邏輯與類神經網路這兩個方法在動態負載預測應用上表現較佳。

Khotanzad et al.[9]利用倒傳遞類神經網路從事美國及加拿大 32 個供電單位的未來一週每小時負載預測，其輸入變數為預測日與過去一天實際最高及最低氣溫、未來 7 日每日的預測最高與最低氣溫、預測日當天小時負載量預測值，研究結果顯示不同的供電單位之類神經網路架構僅需修正隱藏層神經元的數目即可適用。

Chin et al.[14]利用倒傳遞類神經網路結合 rule-base 專家系統，進行台灣地區短期負載預測。由專家系統提供 t 期每小時負載預測值，並結合前 t-1 期的每小時負載實績作為倒傳遞類神經網路的網路訓練範例輸入，以此兩個輸入變數預測第 t 期的系統每小時負載量。研究結果顯示結合專家系統與倒傳遞類神經網絡的預測結果較僅由專家系統或類神經網路所得預測結果來得精確。

Tamimi et al.[26]結合了模糊專家系統與倒傳遞網路，對 1997 年堪薩斯州進行 24 小時負載預測。他們先利用歷史溫度資料建立模糊系統的兩個輸入：預測日前一天的實際溫度與預測日當天之溫度預測，每個輸入分別有五個歸屬函數。系統的輸出是預測日當天與前一天所預期的電力消耗差距。此模糊專家系統的輸出結合了星期指標、小時指標、預測日前三天和一星期前同時段附近三小時的實際負載、溫度、風力等氣象資料，做為倒傳遞網路的輸入值。實驗結果與只用倒傳遞網路或 ARMA 模型比較，顯示這種結合模糊專家系統與倒傳遞網路的模式預測效果較為精確。

3. 即時預測

雖然自 1990 年開始就有大量運用類神經網路於負載預測的文獻，但運用類神經網路於即時預測卻在 1996 年才有這方面的文獻出現。Liu et al.[22]比較了模糊理論、類神經網路與自迴歸分析三種方法在即時用電預測的結果。在類神經網路方面，以前 30 組負載數據、四個時間成分和四個負載參數為輸入值，以預測後半小時內每一分鐘的用電負載。由於欲預測的負載趨勢呈現穩定的狀態，所以誤差收斂在一個極小的範圍之內。

Charytoniuk et al.[40]提出另一種即時用電的機制。在他們的研究中，在把一天的負載趨勢常態化之後，認為雖然每天大體的負載量不盡相同，但是負載相對增加量（Relative Increment in Load）卻非常類似；而且由於

溫度因素在短時間之內變化甚微，不至於影響用電負載。因此，他們捨去傳統上常考慮的溫度因數，並利用類神經網路預測下一時間的負載相對增加量，即可求出下一時間之負載預測。

Chen et al.[28][29]在進行即時用電預測之前先對歷史用電資料分析整理。首先將不合理的負載狀況、突波與遺失的數據以零值取代，並刪除多餘的數據。在進行對數轉換之後，以前 18 組的數據進行下五分鐘的預測。在此模型中，一樣省略氣候因數的影響，只用負載量為輸入值。

由以上的文獻，可清楚的瞭解當用類神經網路預測用電負載時，會隨著預測的時間長短，而有不同的影響因數。以中長期為例，所考慮的變數通常包括電價、國民生產毛額、人口變動與氣候因數等；而短期負載預測所考慮的就有氣溫、日子型態（工作日、週末假日等）或小時指數等因素。即時用電預測，至目前為止文獻雖然不多，但已經肯定氣候因數的影響不大。在現今的能源管理系統之中，負載頻率控制與經濟調度需要更快的預測機制，包括了傳輸性能、機組運轉計畫、與執行效能等需考量的因素，以訂定最佳管理決策。所以即時用電預測所扮演的角色也越來越顯得重要。

表 2. 1 列出幾種預測方法的特性，以供比較。

表 2. 1 預測方法與限制

數學方法	所需最少數據	數據型態	數據間隔	準備時間
迴歸分析法	10 或 20 個	同趨勢且具規律性	短或中	短
時間序列法	20 個以上	同趨勢且具規律性且可自我調整	短或中	短（稍長）
灰色理論	4 個	不定	短、中、長	短
類神經網路	不定	不定	短、中、長	長

2.2 類神經網路

2.2.1 類神經網路簡介

簡單來說，類神經網路是一種包含軟體和硬體的計算系統，使用大量的相連人工神經元來模仿生物神經網路的能力。它模仿真人大腦的思考方式，以平行處理大量資料及容錯快速等特點，開創了另一新計算器的領域。目前已經廣泛的運用在工業的控制或商業決策等方面。模仿生物神經網路的原因，是因爲人腦在圖形與語音辨識能力比現在電腦強，推論原因，認爲人腦中必存在某種計算原理，可完成如此複雜的工作。所以類神經網路的主要目的，就是要瞭解這種計算原理來設計一個功能強大的計算系統，將複雜的問題以模仿人腦處理大量資訊的方式將其簡單化，以利解決。在控制領域中，若要精確的分析輸入與輸出之間的關係，則必須將系統藉由數學的方式做成模型。但是，實際的系統往往是複雜且非線性的，因此如何以數學的方式簡化系統及線性化，就成為控制學門中一個重要的課題。而類神經網路的優點在於並不需要瞭解系統的數學模型為何，而直接以神經網路取代系統的模型，一樣可以得到輸入與輸出之間的關係。

人類的大腦大約由 1011 個神經細胞組成，而每個神經細胞又約有 1000 個神經節（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做出反應，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。類神經網路的運作便源於此，藉由不同的演算法訓練類神經網路使得神經網路的輸出能達到我們所要求的結果。

總而言之，我們針對要解決的問題中，收集它輸出與輸入資料；再利用類神經網路求得輸出入間原本無從得知的對映關係，即兩者之間的轉移系統。以後面對同性質的新問題時，即可利用輸入資料和所求得的轉移系統得到輸出。這就是類神經網路的目的。

2.2.2 生物神經元與人工神經元模型

1. 生物神經元模型

要瞭解類神經網路的模式，當然得從瞭解生物神經模式開始。1949 年 onaldHebb 發表 Hebbian 學習法則：「當人腦在學習不同的事物時，每個腦

細胞的連結都隨時在改變。如果一個腦細胞受到另外一個腦細胞連續的作用時，它們之間連結的力量就會增強。」。此學習法則引導了數十年在類神經網路上的研究。而一個生物神經元模型如圖 2.1 所示，其中：

（1）神經細胞核（Nucleus）：神經細胞的中心體，其作用大概是將神經樹收集的訊號作加成後再作非線性轉換，然後由神經軸傳送到其他的神經細胞。

（2）神經軸（Axon）：連接在神經細胞核上，以傳送由神經細胞核產生的訊號至其他的神經細胞。

（3）神經樹（Dendrites）：是神經細胞呈樹枝狀的輸出入結構。

（4）神經節（Synapse）：輸入與輸出神經樹相連接的點，是神經網路上的記憶體，它表示兩個神經細胞間的聯結強度，此聯結強度以一個數值表示，稱爲加權值（Weight）。

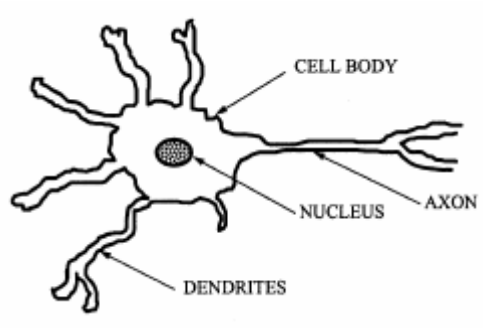

圖 2.1 生物神經元模型

當神經細胞透過輸入神經樹接受到其他神經細胞的訊號後，透過神經細胞核將訊號加成後，再作一次的非線性轉換，產生新訊號；假如這訊號夠強（需大於細胞核的偏權值），就會經由神經軸傳到輸出神經樹，傳給其他神經細胞。一般而言當神經網路在進行學習時，外界刺激神經細胞會改變神經節上的加權值，當加權值最後趨向穩定時，表示學習已告完成。

2. 人工神經元模型

如圖 2.2 所示，人工神經元是模仿生物神經元的原理。每一個人工神經元都有多個輸入值 $x_1, x_2, \cdots x_n$ 及一個輸出值 y，輸入值與輸出值的關係式，一般可用輸入值的加權乘積和的函數來表示，即

$$y_j = f(\sum_{i=1}^{n} w_{ij} x_i - \theta) \qquad (1)$$

其中，

w_{ij}：模仿生物神經細胞的神經節加權值。

θ：模仿生物神經細胞的細胞核偏權值（Bias），即輸入訊號的加權乘積和必須要大於偏權值後，才能被傳輸到其他人工神經元中。

$f(\theta)$：模仿生物神經細胞的細胞核非線性轉移函數。

n：人工神經元輸入數目。

類神經網路就是模仿生物腦神經的學習功能。所謂的學習模仿，就是調整適當權重的過程，學習的目的就是為了求得適當的權重以便將輸入函數對映到適當的輸出函數上。[4]

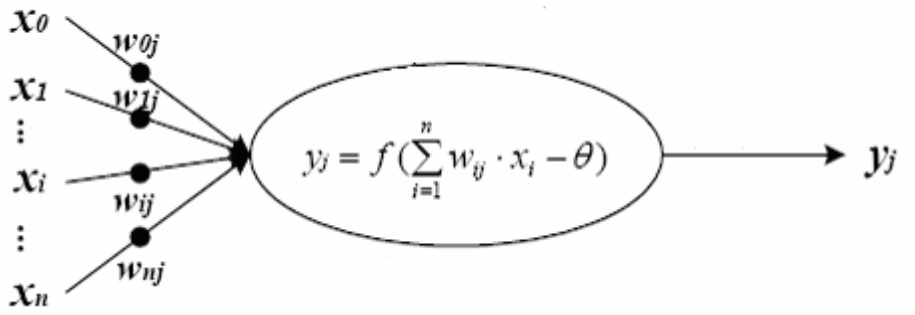

圖 2.2 人工神經元模型

2.3 倒傳遞網路（Back-Propagation Network）

2.3.1 倒傳遞網路原理

倒傳遞網路是種階層式網路，包含輸入層、隱藏層、輸出層，隱藏層可能不只一層，而每一層皆由神經元所構成。同層的神經元彼此不相連，而不同層的神經元則彼此相連。信號的傳輸方向是單方向的，由輸入層傳到輸出層，如圖 2.3 所示[1]

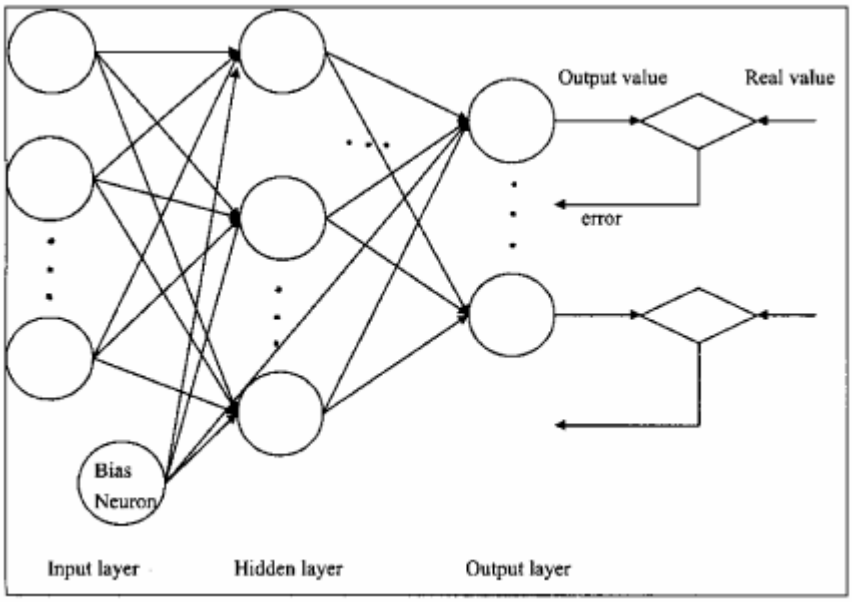

圖 2.3 倒傳遞網路圖

倒傳遞類神經網路的運作過程分為學習過程與回想過程。學習過程是一種監督式學習，它從問題中取得訓練用的輸入值和目標輸出值。輸入值經過學習過程計算出一輸出值，和目標值比較產生誤差值，此誤差值經過計算，再調整加權值與偏權值，如此反覆調整直到輸出和目標值差距在合理範圍內，則稱為網路收斂。

回想過程是在學習過程得到的加權值與偏權值後，將待推的輸入代入，預測最可能的輸出。所以類神經網路的應用，常在於分類、預測、過濾雜訊等各方面。

2.3.2 倒傳遞網路演算法則

倒傳遞演算法是一種監督式學習法則，利用梯度下降（gradient descent）法使性能指標（或稱誤差函數、能量函數）最小化。如圖 2.4 所示，其中

p_r：第 r 個輸入點，$r=1,2,\cdots,R$，R 是輸入點的數目。

m：網路層別，$m=0,1,\cdots,M$；當 $m=0$ 代表輸入層，當 $m=1\sim M-1$ 代表隱藏層，當 $m=M$ 代表輸出層，$m=0,1,\cdots,M$。

M：網路總層數

S^m：第 m 層的節點數，$m=0,1,\cdots,M$。

$w_{i,j}^m$：第 m 層第 i 個節點連結到第 $m-1$ 層第 j 個節點的權重值，$m=0,1,\cdots,M$，

$i=1,2,\cdots,S^m$ ， $j=1,2,\cdots,S^{m-1}$ 。

b_i^m ：第 m 層的的第 i 個節點的偏權值， $m=1,2,\cdots,M$ ， ， $i=1,2,\cdots,S^m$ 。

f^m ：第 m 層的轉移函數， $m=0,1,\cdots,M$ 。

n_i^m ：第 m 層的第 i 個節點中集成函數的輸出值， $m=1,2,\cdots,M$ ， $i=1,2,\cdots,S^m$ 。

a_i^m ：第 m 層的第 i 個節點的輸出值， $m=0,1,\cdots,M$ ， $i=1,2,\cdots,S^m$ 。

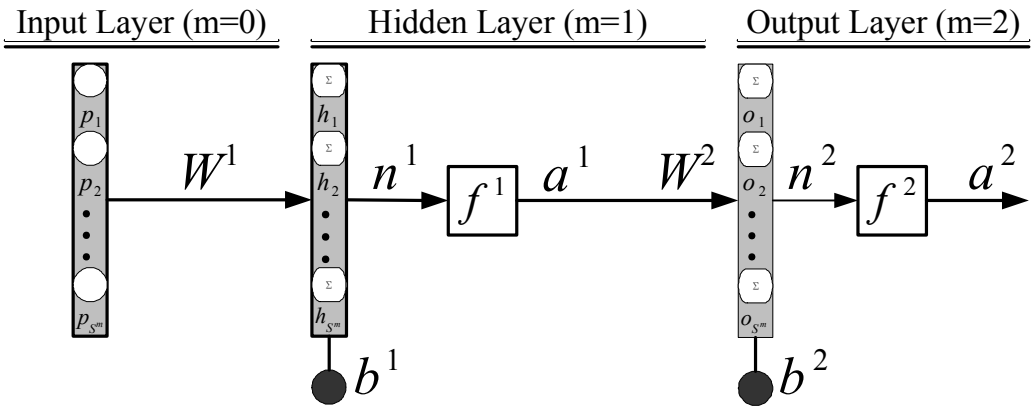

圖 2.4 倒傳遞網路架構圖

對於多層網路，其每一層的輸出為下一層的輸入，

$$\mathrm{a}^{m+1}=f^{m+1}(\mathrm{W}^{m+1}\mathrm{a}^m+\mathrm{b}^{m+1})，m=0,1,2,\cdots,M-1 \quad (2)$$

其中

$$\mathrm{a}^{m+1}=\begin{bmatrix}a_1^{m+1} & a_2^{m+1} & \cdots & a_{S^{m+1}}^{m+1}\end{bmatrix}^T \quad (3)$$

$$\mathrm{W}^{m+1}=\begin{bmatrix} w_{1,1}^{m+1} & w_{1,2}^{m+1} & \cdots & w_{1,S^m}^{m+1} \\ w_{2,1}^{m+1,v} & w_{2,2}^{m+1} & \cdots & w_{2,S^m}^{m+1} \\ \vdots & \vdots & & \vdots \\ w_{S^{m+1},1}^{m+1} & w_{S^{m+1},2}^{m+1} & \cdots & w_{S^{m+1},S^m}^{m+1} \end{bmatrix} \quad (4)$$

$$\mathrm{b}^{m+1}=\begin{bmatrix}b_1^{m+1} & b_2^{m+1} & \cdots & b_{S^{m+1}}^{m+1}\end{bmatrix}^T \quad (5)$$

輸入層節點所接受到外部輸入為

$$\mathrm{a}^0=\mathrm{p} \quad (6)$$

其中

$$\mathrm{p}=\begin{bmatrix} p_1 & p_2 & \cdots & p_R \end{bmatrix}^T \tag{7}$$

輸出層節點的輸出為此網路的輸出

$$\mathrm{a}^M = \mathrm{y} \tag{8}$$

令其性能指標如下：

$$F(\mathrm{x}) = (\mathrm{t}-\mathrm{a}^{\mathrm{M}})^T(\mathrm{t}-\mathrm{a}^{\mathrm{M}}) = \mathrm{e}^T\mathrm{e} = \sum_{j=1}^{S^M}(t_j - a_j^M)^2 \tag{9}$$

其中

x：網路之加權值及偏權值向量（vector）

t：目標值向量（vector）

a^M：輸出層之輸出值向量（vector）

$\mathrm{e} = \mathrm{t} - \mathrm{a}^M$：誤差值向量（vector）

更新網路加權值及偏權值的方法如下：

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha\frac{\partial F}{\partial w_{i,j}^m} \tag{10}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha\frac{\partial F}{\partial b_i^m} \tag{11}$$

其中，α 為學習速率（learning rate）。

因為

$$n_i^m = \sum_{i=1}^{S^{m-1}} w_{i,j}^m a_j^{m-1} + b_i^m \tag{12}$$

所以第（10）（11）式可以簡化成第（13）（14）式

$$\begin{aligned} w_{i,j}^m(k+1) &= w_{i,j}^m(k) - \alpha\frac{\partial F}{\partial n_i^m}\cdot\frac{\partial n_i^m}{\partial w_{i,j}^m} = w_{i,j}^m(k) - \alpha\cdot\frac{\partial F}{\partial n_i^m}\cdot a_j^{m-1} \\ &= w_{i,j}^m(k) - \alpha\cdot\delta_i^m\cdot a_j^{m-1} \end{aligned} \tag{13}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha\frac{\partial F}{\partial n_i^m}\cdot\frac{\partial n_i^m}{\partial b_i^m} = b_i^m(k) - \alpha\cdot\frac{\partial F}{\partial n_i^m} = b_i^m(k) - \alpha\cdot\delta_i^m \tag{14}$$

其中，定義靈敏度 $\delta_i^m \equiv \frac{\partial F}{\partial n_i^m}$ 為性能指標 F 在第 m 層的淨輸入的第 i 個單元的變化量，上兩式若以矩陣形式表示，則可寫成：

$$\mathrm{W}^m(k+1)=\mathrm{W}^m(k)-\alpha\delta^m\left(\mathrm{a}^{m-1}\right)^T \tag{15}$$

$$\mathrm{b}^m(k+1)=\mathrm{b}^m(k)-\alpha\delta^m \tag{16}$$

其中

$$\delta^m\equiv\frac{\partial F}{\partial \mathrm{n}^m}=\left[\begin{array}{cccc}\frac{\partial F}{\partial \mathrm{n}_1^m} & \frac{\partial F}{\partial \mathrm{n}_2^m} & \cdots & \frac{\partial F}{\partial \mathrm{n}_{S^m}^m}\end{array}\right]^T \tag{17}$$

以下介紹靈敏度的計算，利用下列 Jacobian 矩陣

$$\frac{\partial \mathrm{n}^{m+1}}{\partial \mathrm{n}^m}=\left[\begin{array}{cccc}\frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{S^m}^m}\\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_2^{m+1}}{\partial n_{S^m}^m}\\ \vdots & \vdots & & \vdots\\ \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_1^m} & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_{S^m}^m}\end{array}\right] \tag{18}$$

其第 *i,j* 個單元如下：

$$\frac{\partial n_i^{m+1}}{\partial n_j^m}=\frac{\partial\left(\sum_{j=1}^{S^m}w_{i,j}^{m+1}a_j^m+b_i^{m+1}\right)}{\partial n_j^{m,v}}=w_{i,j}^{m+1}\frac{\partial a_j^m}{\partial n_j^m}=w_{i,j}^{m+1}\frac{\partial f^m(n_j^m)}{\partial n_j^m}=w_{i,j}^{m+1}\dot{f}^m(n_j^m) \tag{19}$$

其中

$$\dot{f}^m(n_j^m)=\frac{\partial f^m(n_j^m)}{\partial n_j^m} \tag{20}$$

因此，Jacobian 矩陣可寫成：

$$\frac{\partial \mathrm{n}^{m+1}}{\partial \mathrm{n}^m}=\mathrm{W}^{m+1}\dot{\mathrm{F}}^m(\mathrm{n}^m) \tag{21}$$

其中

$$\dot{\mathrm{F}}^m(\mathrm{n}^m)=\left[\begin{array}{cccc}\dot{f}^m(\mathrm{n}_1^m) & 0 & \cdots & 0\\ 0 & \dot{f}^m(\mathrm{n}_2^m) & \cdots & 0\\ \vdots & \vdots & & \vdots\\ 0 & 0 & \cdots & \dot{f}^m(\mathrm{n}_{S^m}^m)\end{array}\right] \tag{22}$$

藉由使用在矩陣形式中之連鎖律，寫出靈敏度的遞迴關係：

$$\delta^m = \frac{\partial F}{\partial \mathrm{n}^m} = \left(\frac{\partial \mathrm{n}^{m+1}}{\partial \mathrm{n}^m}\right)^T \frac{\partial F}{\partial \mathrm{n}^{m+1}} = \tilde{\mathrm{F}}^m(\mathrm{n}^m)(\mathrm{W}^{m+1})^T \frac{\partial F}{\partial \mathrm{n}^{m+1}} \tag{23}$$
$$= \tilde{\mathrm{F}}^m(\mathrm{n}^m)(\mathrm{W}^{m+1})^T \delta^{m+1}$$

由上式可知，靈敏度從最後一層到第一層被倒向傳遞傳回網路，如下所示：

$$\delta^M \to \delta^{M-1} \to \cdots \to \delta^2 \to \delta^1$$

最後，介紹靈敏度的起始點 δ^M

$$\delta_i^M = \frac{\partial F}{\partial n_i^M} = \frac{\partial \sum_{j=1}^{S^M} \left(t_j - a_j^M\right)^2}{\partial n_i^M} = -2\left(t_i - a_i^M\right)\frac{\partial a_i^M}{\partial n_i^M} \tag{24}$$
$$= -2\left(t_i - a_i^M\right)\frac{\partial f^M(n_i^M)}{\partial n_i^M} = -2\left(t_i - a_i^M\right)\tilde{f}^M(n_i^M)$$

上式若以矩陣形式表示，則可寫成：

$$\delta^M = -2\tilde{\mathrm{F}}^M(\mathrm{n}^M)\left(\mathrm{t} - \mathrm{a}^M\right) \tag{25}$$

2.3.3 倒傳遞網路運算步驟

綜合上述，倒傳遞網路演算摘要如下：

Step1. 隨機選用倒傳遞神經網路的參數初始值，適當的學習速率值。

Step2. 將輸入資料經由網路順向傳遞至輸出層。

$$\mathrm{a}^0 = \mathrm{p} \tag{26}$$

$$\mathrm{a}^{m+1} = f^{m+1}(\mathrm{W}^{m+1}\mathrm{a}^m + \mathrm{b}^{m+1}),\quad m = 0,1,\cdots M-1 \tag{27}$$

$$\mathrm{a}^M = \mathrm{y} \tag{28}$$

Step3. 將靈敏度經由網路倒傳遞至各層。

$$\delta^M = -2\tilde{\mathrm{F}}^M(\mathrm{n}^M)\left(\mathrm{t} - \mathrm{a}^M\right) \tag{29}$$

$$\delta^m = \tilde{\mathrm{F}}^m(\mathrm{n}^m)(\mathrm{W}^{m+1})^T\delta^{m+1},\quad m = 1,2,\cdots M-1 \tag{30}$$

Step4. 更新權值（weights）和偏權值（biases）。

$$\mathrm{W}^m(k+1) = \mathrm{W}^m(k) - \alpha \cdot \delta^m\left(\mathrm{a}^{m-1}\right)^{\mathrm{T}} \tag{31}$$

$$b^m(k+1) = b^m(k) - \alpha\delta^m \quad (32)$$

Step5. 重覆 *Step2* 至 *Step4*，直到系統的性能指標之值達到要求，或訓練次數達到預設值才停止。

2.4 L-M（Levenberg-Marquardt）演算法應用於倒傳遞網路

2.4.1 牛頓演算法（Newton Algorithm）

牛頓演算法是相似於梯度下降（gradient descent）演算法的觀念，其主要精神在對性能指標取二次泰勒展開式：

$$\begin{aligned} &F(x(k+1)) = F(x(k) + \Delta x(k)) \\ &\approx F(x(k)) + g^T(k)\Delta x(k) + \frac{1}{2}\Delta x^T(k)A(k)\Delta x(k) \end{aligned} \quad (33)$$

其中

$x(k)$表示在第 k 次迭代中加權值及偏權值的向量

$g(k)$為第 k 次迭代中 $F(x(k))$的梯度值$\nabla F(x(k))$

$A(k)$為第 k 次迭代中 $F(x(k))$的二次微分值$\nabla^2 F(x(k)$

第（33）式兩邊取其梯度，並令其等於 0，可得

$$g(k) + A(k)\Delta x(k) = 0 \quad (34)$$

因此

$$\Delta x(k) = A^{-1}(k)g(k) \quad (35)$$

$$x(k+1) = x(k) - A^{-1}(k)g(k) \quad (36)$$

2.4.2 L-M 演算法（Levenberg-Marquardt Algorithm）

L-M 演算法是由 Levenberg 及 Marquardt 所發展出來的快速演算法，其特點在針對非線性函數的誤差平方和求其最佳化參數值。茲介紹其觀念如下：

由前節所示，性能指標（performance index）$F(x(k))$，利用牛頓法得其演算法如下：

$$x(k+1) = x(k) - A^{-1}(k)g(k) \quad (37)$$

其中

$x(k)$第 k 次迭代後之加權值、偏權值向量

$$A(k) \equiv \nabla^2 F(x)\big|x = x(k) \tag{38}$$

$$g(k) \equiv \nabla F(x)\big|x = x(k) \tag{39}$$

假設性能指標 $F(x)$為具有平方和型式的函數，如下式所示：

$$F(x) = \sum_{h=1}^{N} v_h^2(x) = v^T(x)v(x) \tag{40}$$

則

$$\nabla F(x) = \begin{bmatrix} \dfrac{\partial F(x)}{\partial x_1} & \dfrac{\partial F(x)}{\partial x_2} & \cdots & \dfrac{\partial F(x)}{\partial x_n} \end{bmatrix}^T \tag{41}$$

其中

$$\left[\nabla F(x)\right]_j = \frac{\partial F(x)}{\partial x_j} = 2\sum_{h=1}^{N} v_h(x)\frac{\partial v_h(x)}{\partial x_j} \tag{42}$$

故$\nabla F(x)$可表示如下

$$\nabla F(x) = 2J^T(x)v(x) \tag{43}$$

其中 $J(x)$為 Jacobian matrix

$$J(x) = \begin{bmatrix} \dfrac{\partial v_1(x)}{\partial x_1} & \dfrac{\partial v_1(x)}{\partial x_2} & \cdots & \dfrac{\partial v_1(x)}{\partial x_n} \\ \dfrac{\partial v_2(x)}{\partial x_1} & \dfrac{\partial v_2(x)}{\partial x_2} & \cdots & \dfrac{\partial v_2(x)}{\partial x_n} \\ \vdots & \vdots & & \\ \dfrac{\partial v_N(x)}{\partial x_1} & \dfrac{\partial v_N(x)}{\partial x_2} & \cdots & \dfrac{\partial v_N(x)}{\partial x_n} \end{bmatrix} \tag{44}$$

亦可表示成

$$J(x) = [J_{h,l}]_{N\times n} \tag{45}$$

其中

$$J_{h,l} = \frac{\partial v_h}{\partial x_l} \tag{46}$$

$$v(x) = [v_1(x) \quad v_2(x) \quad \cdots \quad v_N(x)]^T \tag{47}$$

$$x^T = [x_1 \quad x_2 \quad \cdots \quad x_n] \tag{48}$$

另外

$$\nabla^2 F(x)=\nabla(\nabla F(x))=\begin{bmatrix}\frac{\partial^2 F(x)}{\partial x_1\partial x_1} & \frac{\partial^2 F(x)}{\partial x_1\partial x_2} & \cdots & \frac{\partial^2 F(x)}{\partial x_1\partial x_n}\\ \frac{\partial^2 F(x)}{\partial x_2\partial x_1} & \frac{\partial^2 F(x)}{\partial x_2\partial x_2} & \cdots & \frac{\partial^2 F(x)}{\partial x_2\partial x_n}\\ \vdots & \vdots & & \vdots\\ \frac{\partial^2 F(x)}{\partial x_n\partial x_1} & \frac{\partial^2 F(x)}{\partial x_n\partial x_2} & \cdots & \frac{\partial^2 F(x)}{\partial x_n\partial x_n}\end{bmatrix}=\left[A_{i,j}\right]_{n\times n} \quad (49)$$

其中

$$\left[\nabla^2 F(x)\right]_{i,j}=\frac{\partial^2 F(x)}{\partial x_i\partial x_j}=2\sum_{h=1}^{N}\left\{\frac{\partial v_h(x)\partial v_h(x)}{\partial x_i\partial x_j}+v_h(x)\frac{\partial^2 v_h(x)}{\partial x_i\partial x_j}\right\} \quad (50)$$

故$\nabla^2 F(x)$可表示如下

$$\nabla^2 F(x)=2J^T(x)J(x)+2S(x) \quad (51)$$

此處

$$S(x)=2\sum_{h=1}^{N}v_h(x)\nabla^2 v_h(x) \quad (52)$$

假設二次微分項 $S(x)$非常小，則第（51）式可近似表示成下式：

$$\nabla^2 F(x)\approx 2J^T(x)J(x) \quad (53)$$

再將$\nabla F(x)$（（43）式）及$\nabla^2 F(x)$（53）式）代回原牛頓演算法（式）得

$$\begin{aligned}x(k+1)&=x(k)-[2J^T(x(k))J(x(k))]^{-1}2J^T(x(k))v(x(k))\\&=x(k)-[J^T(x(k))J(x(k))]^{-1}J^T(x(k))v(x(k))\end{aligned} \quad (54)$$

此為 Gauss-Newton Method。

因$[2J^T(x(k))J(x(k))]^{-1}$不一定存在，為克服此問題，所以在 Gauss-Newton Method 中增加一修正項$\mu(k)$如下列（55）、（56）式所示。

$$x(k+1)=x(k)-[J^T(x(k))J(x(k))+\mu(k)I]^{-1}J^T(x(k))v(x(k)) \quad (55)$$

$$\Delta x(k)=-[J^T(x(k))J(x(k))+\mu(k)I]^{-1}J^T(x(k))v(x(k)) \quad (56)$$

其中 I 為單位矩陣。

此即為 L-M 演算法。

當 $\mu(k)$=0 時，演算法變成 Gauss-Newton method。當 $\mu(k)$很大時，L-M 演算法變成具有小的步階大小的梯度下降法。

2.4.3 L-M 演算法應用於倒傳遞演算法

以下說明如何以 L-M 演算法訓練多層網路。我們所用的性能指標是網路輸出層輸出之誤差平方和，如下式（57）所示。

$$F(x)=\sum_{q=1}^{Q}(t_q-a_q^M)^T(t_q-a_q^M)=\sum_{q=1}^{Q}(e_q^Te_q)^2=\sum_{q=1}^{Q}\sum_{k=1}^{S^M}(e_{k,q})^2=\sum_{h=1}^{N}v_h^2(x) \quad (57)$$

其中

t_q 是第 q 個 input／target pair，輸出層輸出之目標值向量。

a_q^M 是對第 q 個 input／target pair，輸出層輸出之預測值向量。

e_q 是第 q 個 input／target pair，輸出層輸出之目標值與預測值之誤差向量。

$e_{k,q}$ 是對第 q 個 input／target pair，輸出層第 k 個神經元目標值與預測值之誤差。

$$v^T=[v_1(x)\quad v_2(x)\quad\cdots\quad v_N(x)]=\left[e_{1,1}\quad e_{2,1}\quad\cdots\quad e_{S^M,1}\quad e_{1,2}\quad\cdots\quad e_{S^M,Q}\right] \quad (58)$$

其中

$$v_h(x)=e_{k,q} \quad (59)$$

$$h(q-1)S^M+K \quad (60)$$

$$N=Q\times S^M \quad (61)$$

令

$$x^T=[x_1\quad x_2\quad\cdots\quad x_n] \quad (62)$$

其中

$$n=S^1(R+1)+S^2(S^1+1)+\cdots S^M(S^{M-1}+1) \quad (63)$$

第（44）式之 Jacobian matrix 可改寫成

$$J(x)=\begin{bmatrix} \frac{\partial e_{1,1}}{\partial w^1_{1,1}} & \frac{\partial e_{1,1}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{1,1}}{\partial w^1_{1,R}} & \frac{\partial e_{1,1}}{\partial w^1_{2,1}} & \cdots & \frac{\partial e_{1,1}}{\partial w^1_{S^1,R}} & \frac{\partial e_{1,1}}{\partial b^1_1} & \cdots & \frac{\partial e_{1,1}}{\partial b^1_{S^1}} & \frac{\partial e_{1,1}}{\partial w^2_{1,1}} & \cdots & \frac{\partial e_{1,1}}{\partial b^1_{S^M}} \\ \frac{\partial e_{2,1}}{\partial w^1_{1,1}} & \frac{\partial e_{2,1}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{2,1}}{\partial w^1_{1,R}} & \frac{\partial e_{2,1}}{\partial w^1_{2,1}} & \cdots & \frac{\partial e_{2,1}}{\partial w^1_{S^1,R}} & \frac{\partial e_{2,1}}{\partial b^1_1} & \cdots & \frac{\partial e_{2,1}}{\partial b^1_{S^1}} & \frac{\partial e_{2,1}}{\partial w^2_{1,1}} & \cdots & \frac{\partial e_{2,1}}{\partial b^1_{S^M}} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots & & & \vdots & \vdots & & \vdots \\ \frac{\partial e_{S^M,1}}{\partial w^1_{1,1}} & \frac{\partial e_{S^M,1}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{S^M,1}}{\partial w^1_{1,R}} & \frac{\partial e_{S^M,1}}{\partial w^1_{2,1}} & \cdots & \frac{\partial e_{S^M,1}}{\partial w^1_{S^1,R}} & \frac{\partial e_{S^M,1}}{\partial b^1_1} & \cdots & \frac{\partial e_{S^M,1}}{\partial b^1_{S^1}} & \frac{\partial e_{S^M,1}}{\partial w^2_{1,1}} & \cdots & \frac{\partial e_{S^M,1}}{\partial b^1_{S^M}} \\ \frac{\partial e_{1,2}}{\partial w^1_{1,1}} & \frac{\partial e_{1,2}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{1,2}}{\partial w^1_{1,R}} & \frac{\partial e_{1,2}}{\partial w^1_{2,1}} & \cdots & \frac{\partial e_{1,2}}{\partial w^1_{S^1,R}} & \frac{\partial e_{1,2}}{\partial b^1_1} & \cdots & \frac{\partial e_{1,2}}{\partial b^1_{S^1}} & \frac{\partial e_{1,2}}{\partial w^2_{1,1}} & \cdots & \frac{\partial e_{1,2}}{\partial b^1_{S^M}} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ \frac{\partial e_{S^M,Q}}{\partial w^1_{1,1}} & \frac{\partial e_{S^M,Q}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{S^M,Q}}{\partial w^1_{1,R}} & \frac{\partial e_{S^M,Q}}{\partial w^1_{2,1}} & \cdots & \frac{\partial e_{S^M,Q}}{\partial w^1_{S^1,R}} & \frac{\partial e_{S^M,Q}}{\partial b^1_1} & \cdots & \frac{\partial e_{S^M,Q}}{\partial b^1_{S^1}} & \frac{\partial e_{S^M,Q}}{\partial w^2_{1,1}} & \cdots & \frac{\partial e_{S^M,Q}}{\partial b^M_{S^M}} \end{bmatrix} \quad (64)$$

其中

$$J(x)=\left[J_{h,l}\right]_{N\times n} \quad (65)$$

$$J_{h,l}=\frac{\partial v_h}{\partial x_l}=\frac{\partial e_{k,q}}{\partial w^m_{i,j}} \quad (66)$$

接著我們推演的重點在探討 Jacobian 矩陣的計算過程。利用微積分中的鏈鎖律，Jacobian matrix 之每一元素（element）可由下列（67）、（79）式求得。

若 x_l 為加權值，則

$$J_{h,l}=\frac{\partial v_h}{\partial x_l}=\frac{\partial e_{k,q}}{\partial w^m_{i,j}}=\frac{\partial e_{k,q}}{\partial n^m_{i,q}}\cdot\frac{\partial n^m_{i,q}}{\partial w^m_{i,j}}3=\tilde{\delta}^m_{i,h}\cdot\frac{\partial n^m_{i,q}}{\partial w^m_{i,j}}=\tilde{\delta}^m_{i,h}\cdot a^{m-1}_{j,q} \quad (67)$$

其中

$$\begin{aligned} &l=S^1(R+1)+S^2(S^1+1)+\cdots+S^{m-1}(S^{m-2}+1)+S^{m-1}(i-1)+j \\ &i=1,2,\cdots S^m \ \ ,j=1,2,\cdots,S^{m+1} \ \ ,m=M,M-1,\cdots 1 \end{aligned} \quad (68)$$

若 x_l 為偏權值，則

$$J_{h,l}=\frac{\partial v_h}{\partial x_l}=\frac{\partial e_{k,q}}{\partial b^m_i}=\frac{\partial e_{k,q}}{\partial n^m_{i,q}}\cdot\frac{\partial n^m_{i,q}}{\partial b^m_i}=\tilde{\delta}^m_{i,h}\cdot\frac{\partial n^m_{i,q}}{\partial b^m_i}=\tilde{\delta}^m_{i,h} \quad (69)$$

其中

$$l=S^1(R+1)+S^2(S^1+1)+\cdots+S^{m-1}(S^{m-2}+1)+S^mS^{m-1}+i \quad (70)$$

我們定義 Marquardt 靈敏度（Marquardt sensitivity）為

$$\tilde{\delta}_{i,h}^{m}=\frac{\partial v_h}{\partial n_{i,q}^{m}}=\frac{\partial e_{k,q}}{\partial n_{i,q}^{m}} \tag{71}$$

當 $m=M$，也就是在輸出層時，輸出層對第 q 個 input／target pair 之靈敏度為

$$\begin{aligned}\tilde{\delta}_{i,h}^{m}&=\frac{\partial v_h}{\partial n_{i,q}^{M}}=\frac{\partial e_{k,q}}{\partial n_{i,q}^{M}}=\frac{\partial(t_{k,q}-a_{k,q}^{M})}{\partial n_{i,q}^{M}}=\frac{-\partial a_{k,q}^{M}}{\partial n_{i,q}^{M}}\\&=\begin{cases}-f^{M}(n_{i,q}^{M}) & \text{for } i=k\\ 0 & \text{for } \mathrm{i}\neq\mathrm{k}\end{cases}\end{aligned} \tag{72}$$

以矩陣型式表示如（73）式。

$$\tilde{\Delta}_{q}^{M}=-F^{M}(n_{q}^{M}) \tag{73}$$

其中 $F^{m}(n^{m})$定義為：

$$F^{m}(n^{m})=\begin{bmatrix}f^{m}(n_1^{m}) & 0 & \cdots & 0\\ 0 & f^{m}(n_2^{m}) & \cdots & 0\\ \vdots & \vdots & & \vdots\\ 0 & 0 & \cdots & f^{m}(n_{S^m}^{m})\end{bmatrix} \tag{74}$$

當 $m=M-1,M-2,\cdots,1$ 也就是在隱藏層時，如同傳統的 BP 演算法一樣，隱藏層對第 q 個 input／target pair 之靈敏度，以矩陣型式表示為

$$\tilde{\Delta}_{q}^{m}=F^{m}(n_{q}^{m})(W^{m+1})^{T}\tilde{\Delta}_{q}^{m+1} \tag{75}$$

對全部的 input／target pair，隱藏層之靈敏度為

$$\tilde{\Delta}^{m}=\begin{bmatrix}\tilde{\Delta}_1^{m} & \tilde{\Delta}_2^{m} & \cdots & \tilde{\Delta}_Q^{m}\end{bmatrix} \tag{76}$$

如同傳統的 BP 演算法一樣，L-M 演算法也是以輸出層之靈敏度$\tilde{\Delta}^{m}$為起始點，利用、(76)式，將靈敏度由輸出層倒向傳遞，穿過網路到第一層，再利用（67)、(69）式計算 Jacobian matrix 的每一元素值後，代入（64）式求得 Jacobian matrix，再由（55）式調整加權值、偏權值。

L-M 倒傳遞演算法可重述如下：

Step1. 隨機選用倒傳遞神經網路的參數初始值 x_0，適當的 μ 值，及一常數值 φ。

Step2. 輸入 Q 筆資料：$P_1,P_1,\cdots,P_Q$。計算所有輸出值 $a_q{}^{M}$，並估計其與目標

值：$t_1,t_2,\cdots,t_Q$之誤差$e_q=t_q-a_q^M$，同時算出其誤差平方和$F(x)$。

Step3. 利用（71）式至（76）式求出各層的靈敏度，再利用（67）、（69）式，計算 Jacobian 矩陣的每一個元素值，再代入（64）式 Jacobian 矩陣。

Step4. 利用（56）式計算$\Delta x(k)$。

Step5. 使用$x(k)+\Delta x(k)$重新計算新的平方誤差和。若新的平方誤差和小於步驟 2 所求得誤差平方和，則更新權值$x(k+1)=x(k)+\Delta x(k)$，再以$\mu(k)/\varphi$取代$\mu(k)$，繼續第 2 步驟。若新的平方誤差和大於步驟 2 所求得誤差平方和，則以$\mu(k)\varphi$取代$\mu(k)$，繼續 *Step4*，直到系統的性能指標之值達到要求，或訓練次數達到預設值才停止。[5]

2.5 改善網路廣義化的方法

在使用訓練數據集來訓練類神經網路時會發生一種稱為「過度配適」（over fitting）的問題，亦即在使用訓練數據集來訓練類神經網路時，我們可以將誤差訓練到一個非常小的值，可是，當我們提供網路一組新的數據集，網路卻計算出來誤差很大的預測值。也就是說，我們的類神經網路只記得訓練數據集，它對新的數據集卻沒有能力使它計算出來的誤差像訓練數據集那樣好，亦即我們的類神經網路不夠廣義化（generalization）。因此本研究採取「規則化」（regularization）作為改善網路廣義化的方法。[2]

「規則化」為藉由修改倒傳遞類神經網路的性能函數，將 *mse* 改為 *msereg*，進而達到改善網路廣義化之能力，*msereg* 如下式所示：

$$msereg=\gamma\cdot mse+(1-\gamma)\cdot msw \tag{77}$$

$$mse=\frac{1}{m}\sum_{i=1}^{m}(T_i-O_i)^2 \tag{78}$$

$$msw=\frac{1}{n}\sum_{j=1}^{n}w_i^2 \tag{79}$$

其中

γ：性能比。

msw：平均權重平方值（mean squared weights）。

m：輸出層中神經元之總數。

n：權重值之數量。

γ 為一個由權重值、偏權值所推導出的複雜關係值，而 David J. C. Mackay 所提出的 Bayesian 結構中，假定網路的權重值和偏權值是具有特定分布之隨機變數，且能夠透過統計技巧自動找出 γ 值。Bayesian 規則化主要結合 L-M 訓練演算法，詳細推導方法請參考文獻[15]。

2.6 徑向基底網路（Radial Basis Function Network, RBFN）

2.6.1 徑向基底網路原理

徑向基底網路由輸入層、隱藏層及輸出層三層所構成的網路，基本架構如圖 2.5，一般選用高斯函數如圖 2.6，利用誤差的反向傳遞，配合梯度下降法，修正連接各層神經元的權重值，將誤差函數最小化，由於可使用簡單矩陣運算來計算網路的輸出值，且可以依照問題任意決定隱藏層神經元個數，具有快速學習與反映的特性。

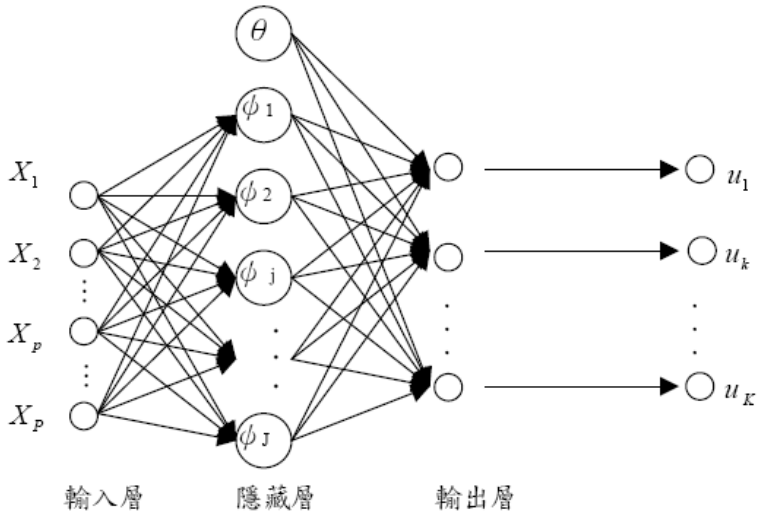

圖 2.5 徑向基底網路基本架構

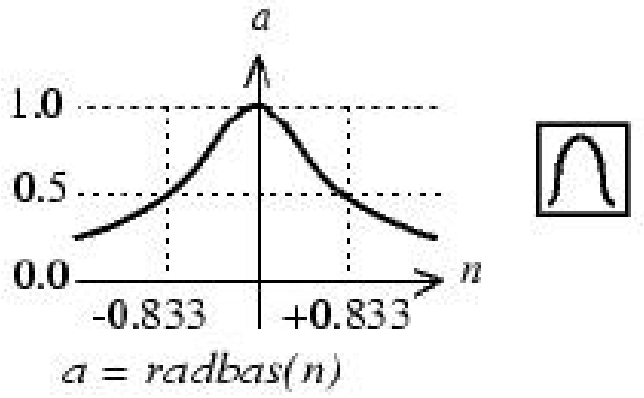

圖 2.6 徑向基底網路轉移函數

2.6.2 徑向基底網路神經元模型

圖 2.7 為一個具有 R 個輸入元素的徑向基網路。徑向基網路轉移函數 radbas 的輸入 n 是權重值向量 w 和輸入向量 p 之間的歐幾里得距離（Euclidean Distance）乘上偏權值 b，如下式：

$$n = \|w - p\| \cdot b \tag{80}$$

圖 2.7 中的 ‖ dist ‖ 就是 w 和 p 的歐幾里得距離。徑向基神經元的轉移函數為高斯函數，如下式：

$$a = radbas(n) = e^{-n^2} \tag{81}$$

當轉移函數的輸入是 0 時，徑向基轉移函數的輸出 a 有最大值 1。當 w 和 p 之間的距離降低時，輸出 a 就增加了。因此徑向基神經元的運作就像是一個偵測器，所以一旦輸入 p 和權重值向量 w 相等時，它就會產生 1。

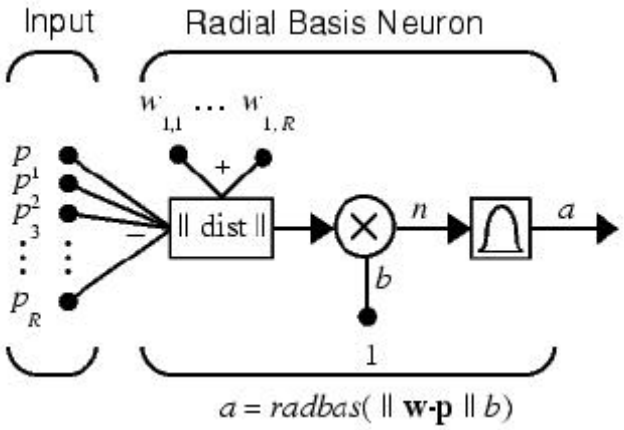

圖 2.7 徑向基底網路

2.6.3 徑向基底網路架構

圖 2.8 為一個具有 R 個輸入元素， S^1 個隱藏層神經元數目，S^2 個隱藏層神經元數目的徑向基網路的架構圖。 ‖ dist ‖ 是輸入權重值矩陣 $IW^{1,1}$ 和輸入向量 p 之間的歐幾里得距離，因此 ‖ dist ‖ 的輸出是具有 S^1 個元素的向量，而其每一個元素即為向量 ${}_iW^{1,1}$ 和輸入向量 p 之間的歐幾里得距離 Z：

$$Z = \sqrt{\sum_{i=1}^{n} (w_i - p_i)^2} \tag{82}$$

其中

W：SxR 的權重值矩陣

P：RxQ 的輸入向量矩陣

Z：SxQ 的向量距離的矩陣

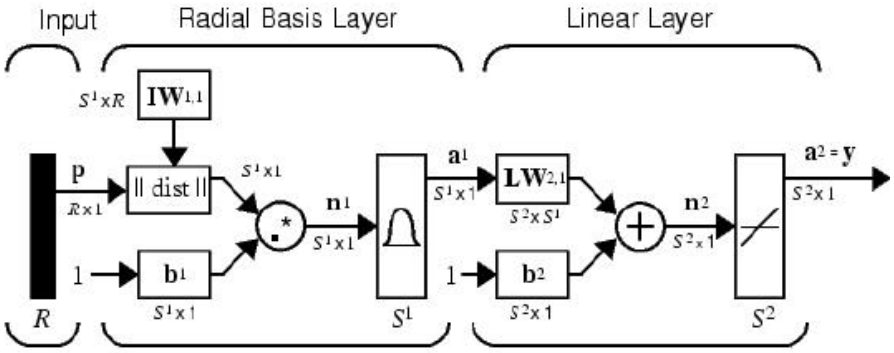

圖 2.8 徑向基網路架構圖

2.6.4 徑向基底網路演算法則

1. 梯度下降法

徑向基網路為監督式網路，利用誤差回饋修正權重，以梯度下降法使誤差函數極小化，而達到函數近似的結果。

首先假設輸入數據的維度是 P，隱藏層神經元個數 J，輸出維度為 K，符號定義

X=[$x1$　$x2$　$x3$ ‧‧‧ xp]：P 維度的輸入

C_j：第 j 個神經元之中心值

σ_j：第 j 個神經元之寬度

||~||：歐幾里德距離

η ：學習速率

α ：慣性因子

u_k：網路第 k 個輸出值

w_{jk}：第 j 個神經元對第 k 個網路輸出的權重

$\phi_j(\cdot)$：第 j 個神經元的輸出值

d_k：第 k 維之期望輸出值

θ_k：對第 k 個輸出之偏壓神經元

選用高斯函數維活化函數，則

第 j 個神經元輸出為

$$\varphi_j(\|X-C_j\|,\sigma_j)=\exp[\frac{-\|X-C_j\|^2}{2\sigma^2}] \quad j=1,2,\cdots \mathrm{J} \tag{83}$$

$$\varphi_0(\|X-C_0\|,\sigma_0)=1 \tag{84}$$

網路網路輸出為

輸出為

$$u_k=\sum_{j=1}^{J} w_{jk}\varphi_j(\|X-C_j\|,\sigma_j)+\theta_k=\sum_{j=0}^{J} w_{jk}\varphi_j \quad k=1,2,\cdots \mathrm{K} \tag{85}$$

誤差函數

$$e_k=d_k-u_k \quad k=1,2,\cdots K \tag{86}$$

定義誤差代價函數

$$e_k^2=(d_k-u_k)^2 \quad k=1,2,\cdots K \tag{87}$$

$$E=\frac{1}{2}\sum_{k=1}^{K} e_k^2=\frac{1}{2}\sum_{k=1}^{K}(d_k-u_k)^2 \tag{88}$$

誤差代價數為網路學習品質的一項重要指標。

網路開始學習之前需設定學習速率 η、初始權重、最大容許誤差 E_{max}、慣性因子 α 以及最大訓練週期 N_{max}。因為誤差代價函數為二次曲線，所以必能找到一組最佳的參數解使得誤差代價函數有最小值，以梯度下降法求取最佳參數解，將誤差代價函數分別對 w_{jk}、c 與 σ_j 作偏微計算，即可得到各參數的修正量，在求得新的參數後在帶回網路取得新的誤差代價函數，如此不斷循環訓練直到網路達到所要求的性能。

第 n+1 次訓練時

$$w_{jk}(n+1)=w_{jk}(n)+\Delta w_{jk}(n) \tag{89}$$

$$c_j(n+1)=c_j(n)+\Delta c_j(n) \tag{90}$$

$$\sigma_j(n+1)=\sigma_j(n)+\Delta\sigma_j(n) \tag{91}$$

其中權重修正量

$$\begin{aligned}\Delta w_{jk}(n)&=-\eta_1\frac{\partial E(n)}{\partial w_{jk}}=-\eta_1\frac{\partial E(n)}{\partial u_k(n)}\cdot\frac{\partial u_k(n)}{\partial w_{jk}(n)}\\&=\eta_1(d_k(n)-u_k(n))\varphi_j(n)\end{aligned} \tag{92}$$

當 j=0 時，$w_{0k}(n)=\Theta_k$ 且 φ_{0k}=1，則(3-18)式可表示成

$$\Delta\theta_{\mathrm{k}}(n)=-\eta_1\frac{\partial E(n)}{\partial\theta_{0k}(n)}=\eta_1(d_k(n)-u(n)) \tag{93}$$

若加入慣性因子權重修正量可寫成

$$\Delta w_{jk}(n)=\eta_1(d_k(n)-u_k(n))\varphi(n)+\alpha\Delta w_{jk}(n-1) \tag{94}$$

中心點修正量

$$\begin{aligned}\Delta c_j(n)&=-\eta_2\frac{\partial E_k(n)}{\partial c_j(n)}=-\eta_1\frac{\partial E_k(n)}{\partial u_k(n)}\cdot\frac{\partial u_k(n)}{\partial\varphi_j(n)}\\&=\eta_2(d_k(n)-u_k(n))w_{jk}(n)\varphi_j(n)\frac{\left\|X(n)-C_j(n)\right\|}{\sigma_j^2}\end{aligned} \tag{95}$$

中心寬度修正量

$$\begin{aligned}\Delta\sigma_j(n)&=-\eta_3\frac{\partial E_k(n)}{\partial\sigma_j(n)}=-\eta_3\frac{\partial E_k(n)}{\partial u_k(n)}\cdot\frac{\partial u_k(n)}{\partial\varphi_j(n)}\cdot\frac{\partial\varphi_j(n)}{\partial\sigma_j(n)}\\&=\eta_3(d_k(n)-u_k(n))w_{jk}(n)\varphi_j(n)\frac{\left\|X(n)-C_j(n)\right\|^2}{\sigma_j^3}\end{aligned} \tag{96}$$

RBFN 藉由誤差的反向傳遞，以（89）、（90）與（91）所得到的修正量，不斷修正網路連結，使得網路輸出值漸漸逼近目標輸出值，網路學習終止條件有 a. 誤差代價函數小於最大容許誤差值。b. 訓練次數達到特定上限。c. 誤差梯度小於預設條件。在學習過程中，權重的調整量與設定之學習速率有關，較低的學習速率，會有較穩定的收斂效果，但需要更長的訓練時間，而且較容易陷入區域最小值，較高的學習速率雖然收斂速度快，但是會導致網路穩定性降低，使得網路發生震盪無法完成學習，因此學習速率的選擇相當不易，此時可以加入慣性因子，使得網路震盪幅度降低，可以容許較高的學習速率，使網路同時具備穩定性與效能。當網路訓練完成後，可以輸入數筆訓練資料以外的樣本，以測試網路的學習成果與推廣能力。

2. 虛擬反矩陣法

也可以虛擬反矩陣求解網路權重值，因為整個網路訓練的目標是希望網路輸出 u 與期望輸出 d 符合，可以把整個架構表示成為

$$\begin{bmatrix} 1 & \varphi_1(X_1) & \varphi_2(X_1) & \cdots & \varphi_j(X_1) & \cdots & \varphi_J(X_1) \\ 1 & \varphi_1(X_2) & \varphi_1(X_2) & \cdots & \varphi_j(X_2) & \cdots & \varphi_J(X_2) \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ 1 & \varphi_1(X_n) & \varphi_2(X_n) & \cdots & \varphi_j(X_n) & \cdots & \varphi_J(X_n) \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ 1 & \varphi_1(X_N) & \varphi_2(X_N) & \cdots & \varphi_j(X_N) & \cdots & \varphi_J(X_N) \end{bmatrix} \begin{bmatrix} \theta \\ w_1 \\ w_2 \\ \vdots \\ w_j \\ \vdots \\ w_J \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_j \\ \vdots \\ d_J \end{bmatrix} \quad (97)$$

以矩陣的方式表示

$$\Phi \cdot W = D \quad (98)$$

Φ為 N 筆訓練資料之神經元輸出，W 為最佳權重，D 為網路之期望輸出，可用虛擬反矩陣運算求得 W

$$W = (\Phi^T \cdot \Phi)^{-1} \cdot \Phi^T \cdot D \quad (99)$$

通常我們所收集的訓練資料無法避免含有雜訊干擾，所以會用較多的資料量來彌補雜訊所造成的不良影響，在訓練資料越多（N>>J+1）的情況下，利用反矩陣求出的權重會越接近最佳值。在虛擬反矩陣求解過程中，類神經網路的中心值與中心點寬度並不參與學習，而是依據實際問題狀況來決定，或是由經驗法則來決定。整個網路的學習流程可以歸納成為圖 2.9 RBFN 訓練流程所示。[3]

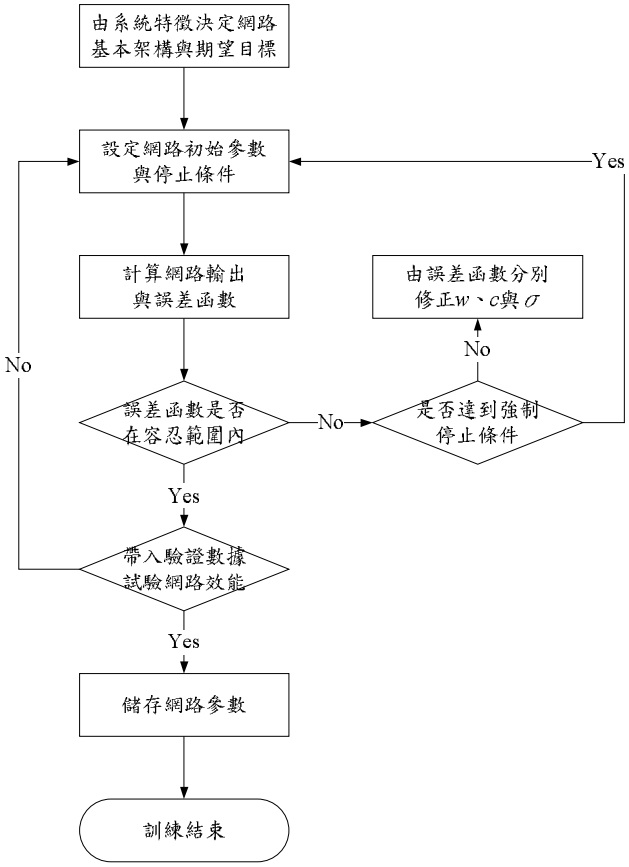

圖 2.9 RBFN 訓練流程

2.7 一般迴歸神經網路（General Regression Neural Network, GRNN）

2.7.1 一般迴歸神經網路簡介

一般迴歸神經網路是從機率神經網路（Probability Neural Network, PNN）所演變而來的，為監督式學習網路的一種。Donald F. Specht 於 1988 年提出 PNN，但是 PNN 只適用於分類問題，而無法解決連續變數問題。因而，Donald F. Specht 在 1991 便提出了 GRNN 的學習演算法。GRNN 可學習一個動態模式作為預測或控制用，因此迴歸問題便可用 GRNN 來解決，不論此迴歸模式為線性或非線性。

2.7.2 一般迴歸神經網路理論背景

相依變數（dependent variable）Y 在獨立變數（independent variable）X 上的迴歸，Y 通常代表系統的輸出值，而 X 通常為系統的輸入值。GRNN 方法並不需要像傳統的迴歸分析先假設一個明確的函數形式，只需以機率密度函數的方式來呈現。

假設 $f(x, y)$ 為變數 X 和變數 Y 的聯合機率密度函數。x 為變數 X 的一個測量值，那麼 Y 在 x 上的迴歸為：

$$E(Y|x)=\frac{\int_{-\infty}^{\infty} yf(x,y)dy}{\int_{-\infty}^{\infty} f(x,y)dy} \tag{100}$$

而 $f(x, y)$ 是未知的，我們必須從 X 和 Y 的觀測值來估計 $f(x, y)$，在這裡我們用 Parzen 所提出 Parzen window 的無母數方法來 $f(x, y)$，其結果如式(15)：

$$\hat{E}(Y|x)=\hat{y}(x)=\frac{\sum_{i=1}^{n}\exp[-\frac{(x-x_i)^T(x-x_i)}{2\sigma^2}]\int_{-\infty}^{\infty} y\exp[-\frac{(y-y_i)^2}{2\sigma^2}]dy}{\sum_{i=1}^{n}\exp[-\frac{(x-x_i)^T(x-x_i)}{2\sigma^2}]\int_{-\infty}^{\infty}\exp[-\frac{(y-y_i)^2}{2\sigma^2}]dy} \tag{101}$$

其中 x_i 與 y_i 為變數 X 和 Y 的樣本值。

σ 為平滑參數（smoothing parameter），為一大於 0 之常數；平滑參數是在 GRNN 中唯一需要以學習方式決定的參數。

將式(15)簡化可得：

$$\hat{E}(Y|x)=\hat{y}(x)=\frac{\sum_{i=1}^{n} y_i\exp[-\frac{(x-x_i)^T(x-x_i)}{2\sigma^2}]}{\sum_{i=1}^{n}\exp[-\frac{(x-x_i)^T(x-x_i)}{2\sigma^2}]}=\frac{\sum_{i=1}^{n} y_i\exp[-\frac{D_i^2}{2\sigma^2}]}{\sum_{i=1}^{n}\exp[-\frac{D_i^2}{2\sigma^2}]} \tag{102}$$

其中

$$D_i^2=(x-x_i)^T(x-x_i) \tag{103}$$

2.7.3 一般迴歸神經網路架構

圖 2.10 為 GRNN 網路架構圖。圖中所示之輸入單元（input unit）為分配單元，負責將所有 X 的測量值分配給第二層所有的單元，型態單元（pattern unit）。每一個型態單元（pattern unit）代表一個訓練範例或是一個 cluster center。當一個新的 X 向量進入網路之後，此向量減去訓練範例的向量，兩者差之平方值會被加總並輸入到非線性的作用函數，作用函數所出來之值，便是型態單元的輸出值。而型態單元的輸出值會被傳送到總和單元（summation unit）。前述之作用函數為

$$\exp[-\frac{D_i^2}{2\sigma^2}] \quad (104)$$

其中 D_i^2 如（103）式中所定義。

總和單元分別完成加權向量的加總以及 Y 的所有觀測值乘以加權向量值的加總。總和單元的兩個輸出分別為（105）式及（106）式

$$\sum_{i=1}^{n} \exp[-\frac{D_i^2}{2\sigma^2}]$$

$$\sum_{i=1}^{n} y_i \exp[-\frac{D_i^2}{2\sigma^2}] \quad (105)$$

$$\sum_{i=1}^{n} \exp[-\frac{D_i^2}{2\sigma^2}] \quad (106)$$

輸出單元則是總和單元的兩個輸出值相除（105）式除以（106）式，可得到 y 的估計值，$\hat{y}(X)$。[6]

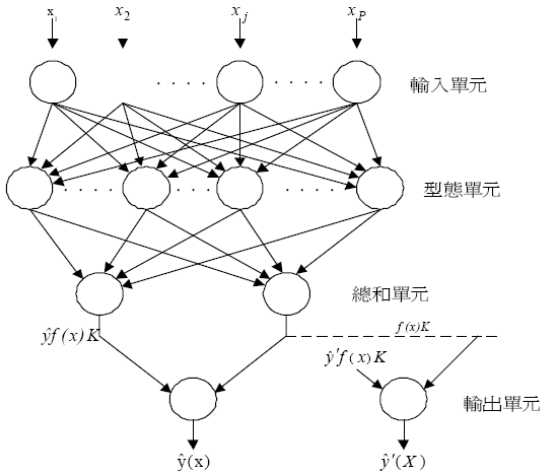

圖 2.10 GRNN 網路架構圖

2.7.4 一般迴歸神經網路演算步驟

Step1. 決定輸入變數

Step2. 蒐集訓練範例

Step3. 學習訓練範例來決定平滑參數

Step4. 將平滑參數代入 GRNN 的公式中。

Step5. 輸入一個未知樣本的輸入向量 *x*。

Step6. 計算推論輸出值 $\hat{y}$。

2.7.5. 決定 GRNN 的平滑參數

在 GRNN 中，最重要且是唯一需要決定的為平滑參數，一般使用由 Donald F. Specht (1991)所提出的 Holdout Method 來決定平滑參數，其步驟如下：

Step1. 先選定一個特定的 σ 值。

Step2. 一次只移走一個訓練範例，用剩下的範例建構一個網路，用此網路來估計移走的那個樣本的估計值 $\hat{y}$。

Step3. 重複步驟 *Step2* n 次（n 為訓練範例數），記錄每個估計值與範例值之間的 MSE（Mean Square of Error），並且把每一次的 MSE 加總。

Step4. 採用其他的 σ 值，重複步驟 *Step2* 與 *Step3*。

Step5. MSE 最小的 σ 值，即為最佳的 σ 值。

然而因為平滑參數為一大於零之數，其涵蓋範圍太大，如果只用 Holdout Method，很難找出最適合的平滑參數值，因此，經過幾次不同的 σ 值帶入後，我們可以畫出 MSE 與 σ 的函數圖，依其曲線找出能讓 MSE 最小的 σ 值的約略區間，接著我們用遞減式搜尋法。所謂遞減式搜尋法即是，先設一平滑參數初值與折減係數，並設定『學習循環』次數，例如：

初值：1.0

折減係數：0.5

學習循環：10

學習步驟會先以平滑參數 σ=1.0 開始，第二循環再以 σ=0.5，第三循環以 σ=0.25 測試，直到第 10 次 σ=0.0019，選擇會使 MSE 為最小的平滑參數為最佳平滑參數。

不同的平滑參數對 GRNN 會有不同的影響；過小的平滑參數回想範例幾乎只受鄰近範例的影響；過大的平滑參數，回想範例受全部範例影響且影響力相同，這也是說，回想範例在回想的過程中常常會受到許多雜訊的影響；而適中的平滑參數就會將雜訊數據平滑掉，對回想範例做出較正確的估計，圖 2.11~圖 2.16 說明此現象。

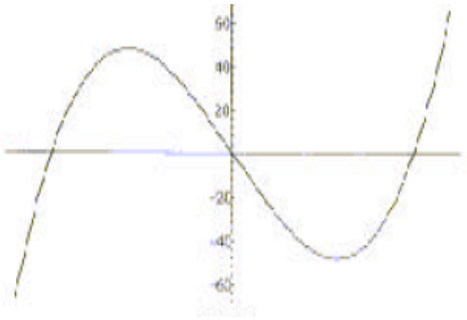

圖 2.11 平滑參數 1

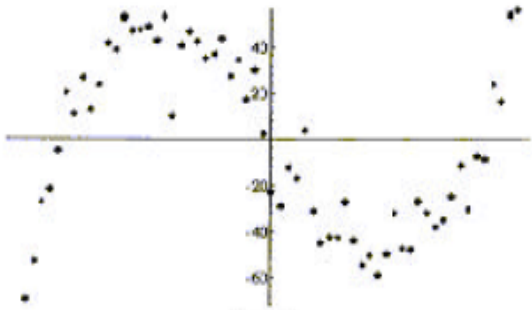

圖 2.12 平滑參數 2

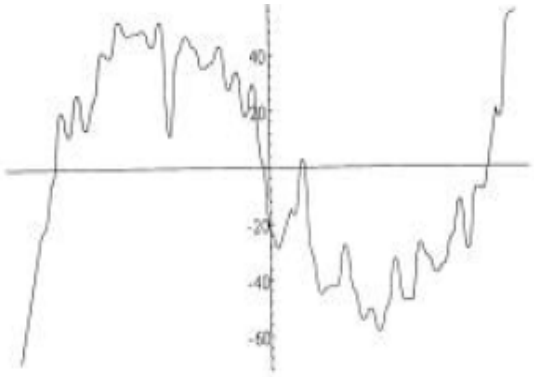

圖 2.13 平滑參數 3

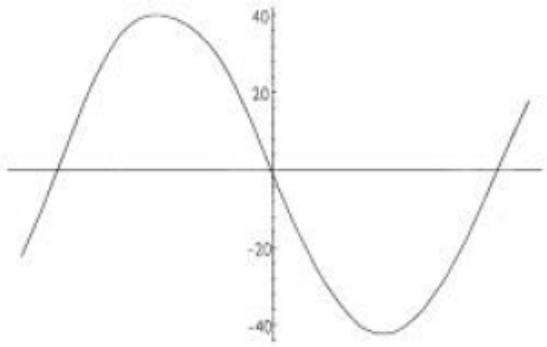

圖 2.14 平滑參數 4

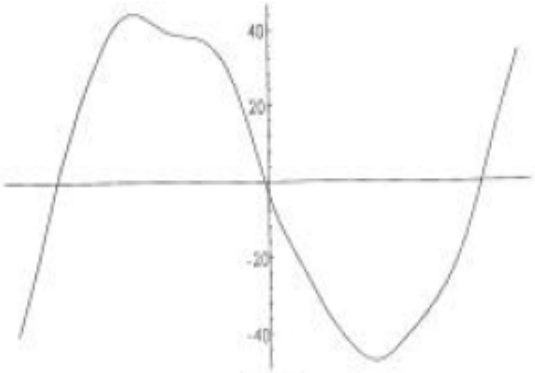

圖 2.15 平滑參數 5

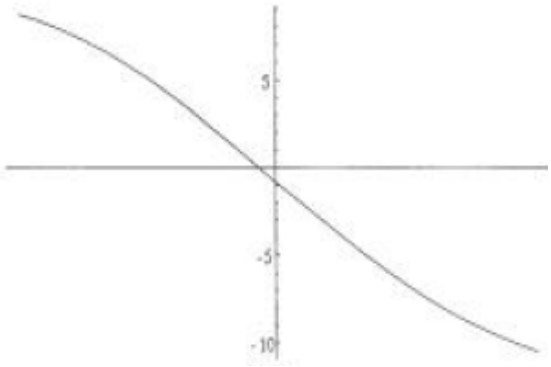

圖 2.16 平滑參數 6

圖 2.11 為二次曲線，而圖 2.12 是從圖 2.11 的二次曲線中，參雜一些隨機雜訊所蒐集到的樣本值。根據平滑參數大小的不同，我們得出 GRNN 對這些樣本估計出的估計值也會不同。如圖 2.13 是平滑參數過小；圖 2.14 是平滑參數有點小；圖 2.15 是適中的平滑參數；圖 2.1 則是平滑參數過大。

6. GRNN 與其他類神經網路之不同

一般的監督式學習網路中，其學習過程首先是以隨機亂數設定初始網路連結加權值，然後將訓練範例的輸入向量載入網路輸入層，透過回想過程計算推論輸出向量，接著再應用推論輸出向量與訓練範例的目標輸出向

量之差距修正網路連結加權值。重複著回想過程計算輸出向量、修正網路連結加權值，直到收斂為止。

GRNN 的學習過程則是截然不同地，其網路連結加權值是由訓練範例的輸出向量與輸入向量決定。與其他監督式類神經網路的不同之處可歸類如下：

（1）不用初始網路連結加權值。

（2）學習過程與回想過程無關。

（3）不使用推論輸出向量與訓練範例的目標輸出向之差距，來修正網路連結加權值。

（4）無迭代學習過程（one-pass learning）。

（5）學習過程之目的在於尋找最佳地平滑參數。

（6）網路的神經元數與訓練範例有關。

第三章　研究方法與系統建構

3.1 系統架構與系統發展

圖 3.1 為本研究提出之平行式類神經網路電力負載預測系統架構圖。首先，針對電力負載預測，分析出真正影響預測結果之相關影響變數。接著，再將每一相關影響變數對應到每一個輸入神經元。之後，再根據輸入神經元與輸入神經元的關係，將每個輸入神經元另外加以互相連結，以達到變數與變數之間相關性的調整。之後，每個輸入神經元，進行各自的類神經網路的運算，由各自的輸出神經元得到預測結果。最後，再根據各自的輸出神經元所得到的預測結果進行彙整，以得到最後整體預測之結果。

此外，每個輸入神經元在進行各自的類神經網路的運算中，均採用倒傳遞類神經網路的架構。即以倒傳遞類神經網路掌握輸入神經元與輸出神經元間非線性的關係，而以新加入的連結掌握每一輸入神經元與其他輸入神經元間相關性的調整，以進行平行輸入，平行運算，同步調整之系統功能。

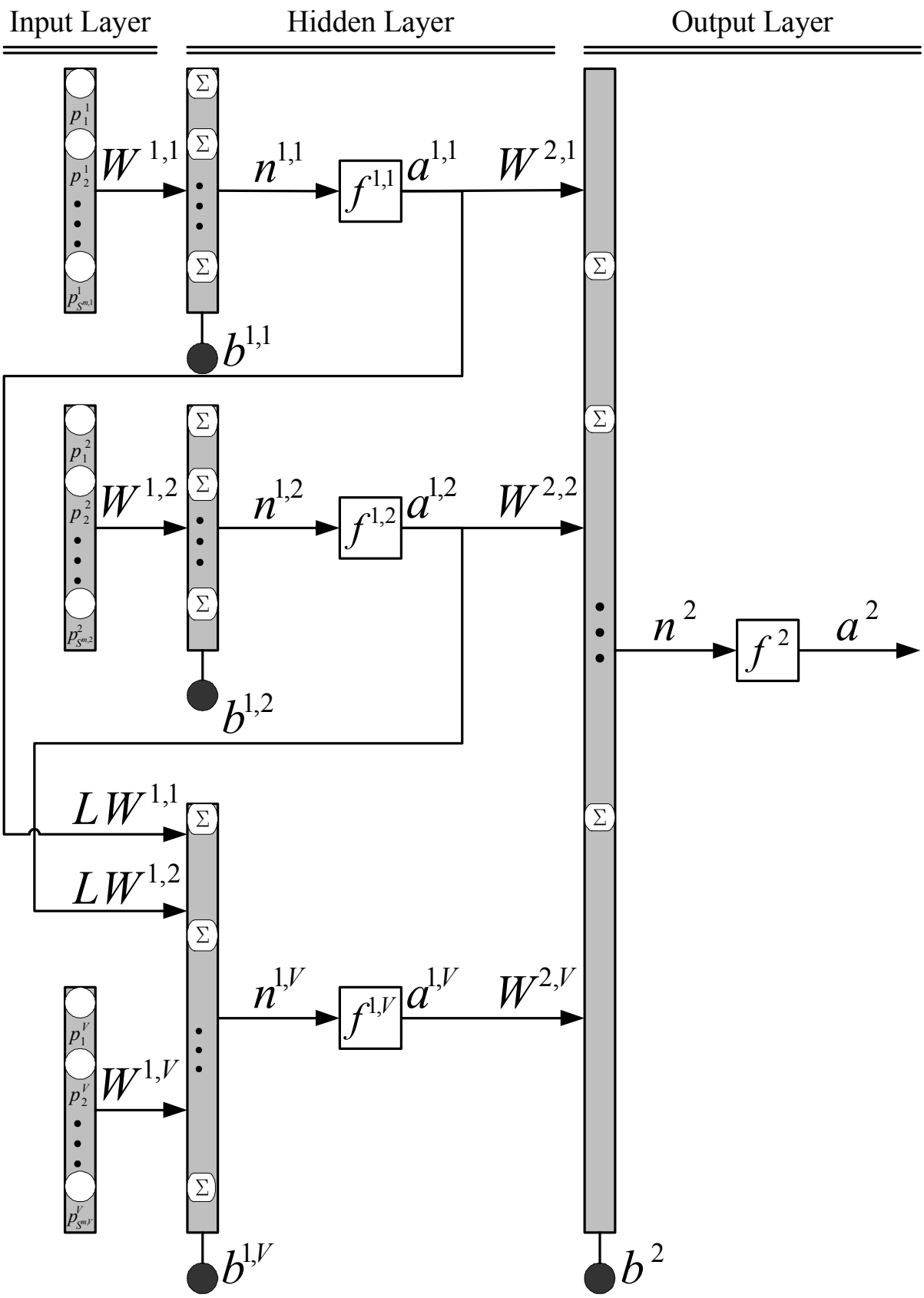

圖 3.1 平行式類神經網路架構圖

3.1.1 系統符號說明

p_i^v：第 v 個輸入變數，$v=1,2,\cdots,V$，$i=1,2,\cdots,S^v$。

m：網路層別，$m=0,1,\cdots,M$；當 $m=0$ 代表輸入層，當 $m=1\sim M-1$ 代表隱藏層，當 $m=M$ 代表輸出層，$m=0,1,\cdots,M$，$v=1,2,\cdots,V$。

M：網路總層數

$S^{m,v}$：第 m 層的第 v 個變數的節點數，$m=0,1,\cdots,M$，$v=1,2,\cdots,V$。

$w_{i,j}^{m,v}$：在第 v 個變數中，第 $m-1$ 層第 j 個節點連結到第 m 層第 i 個節點的權重值，$m=0,1,\cdots,M$，$v=1,2,\cdots,V$，$i=1,2,\cdots,S^{m,v}$，$j=1,2,\cdots,S^{m-1,v}$。

$lw_{i,j}^{m,v}$：在第 m 層中，第 v 個變數第 j 個節點連結到第 V 個變數第 i 個節點的權重值，$m=1,2,\cdots,M-1$，$v=1,2,\cdots,V-1$，$i=1,2,\cdots,S^{m,V}$，$j=1,2,\cdots,S^{m,v}$。

$b_i^{m,v}$：第 m 層的第 v 個變數的第 i 個節點的偏權值，$m=0,1,\cdots,M$，$v=1,2,\cdots,V$，$i=1,2,\cdots,S^{m,v}$。

$f^{m,v}$：第 m 層的第 v 個變數的轉移函數，$m=0,1,\cdots,M$，$v=1,2,\cdots,V$。

$n_i^{m,v}$：第 m 層的第 v 個變數的第 i 個節點中集成函數的輸出值，$m=0,1,\cdots,M$，$v=1,2,\cdots,V$，$i=1,2,\cdots,S^{m,v}$。

$a_i^{m,v}$：第 m 層的第 v 個變數的第 i 個節點的輸出值，$m=0,1,\cdots,M$，$v=1,2,\cdots,V$，$i=1,2,\cdots,S^{m,v}$。

3.1.2 系統推導

對於多層網路，其每一層的輸出為下一層的輸入，

輸入層節點所接受到外部輸入為

$$\mathrm{a}^{0,v}=\mathrm{p}^v,\quad v=1,2,\cdots V \tag{107}$$

$$\text{其中 } \mathrm{p}^v=\begin{bmatrix} p_1^v & p_2^v & \cdots & p_{S^{0,v}}^v \end{bmatrix}^T,\quad v=1,2,\cdots V \tag{108}$$

隱藏層節點的輸出為

$$\mathrm{a}^{m+1,v}=\begin{cases} f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\mathrm{b}^{m+1,v}) \\ f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\mathrm{LW}^{m+1,1}\mathrm{a}^{m+1,1}+\mathrm{LW}^{m+1,2}\mathrm{a}^{m+1,2}+\cdots+\mathrm{LW}^{m+1,V-1}\mathrm{a}^{m+1,V-1}+\mathrm{b}^{m+1,v}) \end{cases}$$
$$=\begin{cases} f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\mathrm{b}^{m+1,v}) & ，v=1,2,\cdots,V-1 \\ f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\sum_{v=1}^{V-1}\mathrm{LW}^{m+1,v}\mathrm{a}^{m+1,v}+\mathrm{b}^{m+1,v}) & ，v=V \end{cases} \quad (109)$$

其中

$$\mathrm{W}^{m+1,v}=\begin{bmatrix} w_{1,1}^{m+1,v} & w_{1,2}^{m+1,v} & \cdots & w_{1,S^{m,v}}^{m+1,v} \\ w_{2,1}^{m+1,v} & w_{2,2}^{m+1,v} & \cdots & w_{2,S^{m,v}}^{m+1,v} \\ \vdots & \vdots & & \vdots \\ w_{S^{m+1,v},1}^{m+1,v} & w_{S^{m+1,v},2}^{m+1,v} & \cdots & w_{S^{m+1,v},S^{m,v}}^{m+1,v} \end{bmatrix}，v=1,2,\cdots,V \quad (110)$$

$$\mathrm{LW}^{m+1,v}=\begin{bmatrix} lw_{1,1}^{m+1,v} & lw_{1,2}^{m+1,v} & \cdots & lw_{1,S^{m+1,v}}^{m+1,v} \\ lw_{2,1}^{m+1,v} & lw_{2,2}^{m+1,v} & \cdots & lw_{2,S^{m+1,v}}^{m+1,v} \\ \vdots & \vdots & & \vdots \\ lw_{S^{m+1,V},1}^{m+1,v} & lw_{S^{m+1,V},2}^{m+1,v} & \cdots & lw_{S^{m+1,V},S^{m+1,v}}^{m+1,v} \end{bmatrix}，v=1,2,\cdots,V-1 \quad (111)$$

$$\mathrm{b}^{m+1,v}=\begin{bmatrix} b_1^{m+1,v} & b_2^{m+1,v} & \cdots & b_{S^{m+1,v}}^{m+1,v} \end{bmatrix}^T，v=1,2,\cdots,V \quad (112)$$

輸出層節點的輸出為此網路的輸出

$$\mathrm{a}^M=\mathrm{y} \quad (113)$$

令其性能指標如下：

$$F(\mathrm{x})=(\mathrm{t}-\mathrm{a}^{\mathrm{M}})^T(\mathrm{t}-\mathrm{a}^{\mathrm{M}})=\mathrm{e}^T\mathrm{e}=\sum_{j=1}^{S^M}(t_j-a_j^M)^2 \quad (114)$$

其中

x：網路之加權值及偏權值向量（vector）

t：目標值向量（vector）

a^M：輸出層之輸出值向量（vector）

$\mathrm{e}=\mathrm{t}-\mathrm{a}^M$：誤差值向量（vector）

更新網路加權值及偏權值的方法如下：

$$w_{i,j}^{m,v}(k+1)=w_{i,j}^{m,v}(k)-\alpha\frac{\partial F}{\partial w_{i,j}^{m,v}}，v=1,2,\cdots,V \quad (115)$$

$$lw_{i,j}^{v}(k+1)=lw_{i,j}^{v}(k)-\alpha\frac{\partial F}{\partial w_{i,j}^{v}}\text{，}v=1,2,\cdots,V-1 \tag{116}$$

$$b_i^{m,v}(k+1)=b_i^{m,v}(k)-\alpha\frac{\partial F}{\partial b_i^{m,v}}\text{，}v=1,2,\cdots,V \tag{117}$$

其中，α 為學習速率（learning rate）。

因為

$$n_i^{m,v}=\begin{cases}\sum_{i=1}^{S^{m,v}} w_{i,j}^{m,v}a_j^{m-1,v}+b_i^{m,v} & ,\ v=1,2,\cdots V-1\\ \sum_{j=1}^{S^{m,v}} w_{i,j}^{m,v}a_j^{m-1,v}+\sum_{v=1}^{V-1}\sum_{j=1}^{S^{m,v}} lw_{i,j}^{m,v}a_j^{m,v}+b_i^{m,v} & ,\ v=V\end{cases} \tag{118}$$

所以第（115）（116）（117）式可以簡化成第式

$$\begin{aligned}w_{i,j}^{m,v}(k+1)&=w_{i,j}^{m,v}(k)-\alpha\frac{\partial F}{\partial n_i^{m,v}}\cdot\frac{\partial n_i^{m,v}}{\partial w_{i,j}^{m,v}}\\&=w_{i,j}^{m,v}(k)-\alpha\cdot\frac{\partial F}{\partial n_i^{m,v}}\cdot a_j^{m-1,v}\qquad\text{，}v=1,2,\cdots,V\\&=w_{i,j}^{m,v}(k)-\alpha\cdot\delta_i^{m,v}\cdot a_j^{m-1,v}\end{aligned} \tag{119}$$

$$\begin{aligned}lw_{i,j}^{m,v}(k+1)&=lw_{i,j}^{m,v}(k)-\alpha\frac{\partial F}{\partial n_i^{m,v}}\cdot\frac{\partial n_i^{m,v}}{\partial lw_{i,j}^{m,v}}\\&=lw_{i,j}^{m,v}(k)-\alpha\frac{\partial F}{\partial n_i^{m,v}}\cdot a_j^{m,v}\qquad\text{，}v=1,2,\cdots,V-1\\&=lw_{i,j}^{m,v}(k)-\alpha\frac{\partial F}{\partial n_i^{m,v}}\cdot a_j^{m,v}\\&=lw_{i,j}^{m,v}(k)-\alpha\cdot\delta_i^{m,v}\cdot a_j^{m,v}\end{aligned} \tag{120}$$

$$b_i^{m,v}(k+1)=b_i^{m,v}(k)-\alpha\frac{\partial F}{\partial n_i^{m,v}}\cdot\frac{\partial n_i^{m,v}}{\partial b_i^{m,v}}=b_i^{m,v}(k)-\alpha\cdot\frac{\partial F}{\partial n_i^{m,v}}=b_i^{m,v}(k)-\alpha\cdot\delta_i^{m,v} \tag{121}$$

其中，定義靈敏度 $\delta_i^{m,v}\equiv\frac{\partial F}{\partial n_i^{m,v}}$ 為性能指標 F 在第 m 層第 v 個變數的淨輸入的第 i 個單元的變化量，上三式若以矩陣形式表示，則可寫成：

$$\mathrm{W}^{m,v}(k+1)=\mathrm{W}^{m,v}(k)-\alpha\cdot\delta^{m,v}\left(\mathrm{a}^{m-1,v}\right)^{\mathrm{T}}\text{，}v=1,2,\cdots,V \tag{122}$$

$$\mathrm{LW}^{m,v}(k+1)=\mathrm{LW}^{m,v}(k)-\alpha\delta^{m,v}\left(\mathrm{a}^{m,v}\right)^{\mathrm{T}}\text{，}v=1,2,\cdots,V-1 \tag{123}$$

$$\mathrm{b}^{m,v}(k+1)=\mathrm{b}^{m,v}(k)-\alpha\delta^{m,v}，v=1,2,\cdots,V \tag{124}$$

以下介紹靈敏度的計算，利用下列 Jacobian 矩陣

$$\frac{\partial \mathrm{n}^{m+1,v}}{\partial \mathrm{n}^{m,v}}=\begin{bmatrix} \frac{\partial n_1^{m+1,v}}{\partial n_1^{m,v}} & \frac{\partial n_1^{m+1,v}}{\partial n_2^{m,v}} & \cdots & \frac{\partial n_1^{m+1,v}}{\partial n_{S^{m,v}}^{m,v}} \\ \frac{\partial n_2^{m+1,v}}{\partial n_1^{m,v}} & \frac{\partial n_2^{m+1,v}}{\partial n_2^{m,v}} & \cdots & \frac{\partial n_2^{m+1,v}}{\partial n_{S^{m,v}}^{m,v}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial n_{S^{m+1,v}}^{m+1,v}}{\partial n_1^{m,v}} & \frac{\partial n_{S^{m+1,v}}^{m+1,v}}{\partial n_2^{m,v}} & \cdots & \frac{\partial n_{S^{m+1,v}}^{m+1,v}}{\partial n_{S^{m,v}}^{m,v}} \end{bmatrix} \tag{125}$$

其第 *i,j* 個單元如下：

$$\frac{\partial n_i^{m+1,v}}{\partial n_j^{m,v}}=\frac{\partial\left(\sum_{j=1}^{S^{m,v}} w_{i,j}^{m+1,v}a_j^{m,v}+b_i^{m+1,v}\right)}{\partial n_j^{m,v}}=w_{i,j}^{m+1,v}\frac{\partial a_j^{m,v}}{\partial n_j^{m,v}} \tag{126}$$

$$=w_{i,j}^{m+1,v}\frac{\partial f^{m,v}(n_j^{m,v})}{\partial n_j^{m,v}}=w_{i,j}^{m+1,v}\dot{f}^{m,v}(n_j^{m,v})$$

其中，

$$\dot{f}^{m,v}(n_j^{m,v})=\frac{\partial f^{m,v}(n_j^{m,v})}{\partial n_j^{m,v}} \tag{127}$$

因此，Jacobian 矩陣可寫成：

$$\frac{\partial \mathrm{n}^{m+1,v}}{\partial \mathrm{n}^{m,v}}=\mathrm{W}^{m+1,v}\dot{F}^{m,v}(\mathrm{n}^{m,v}) \tag{128}$$

其中，

$$\dot{F}^{m,v}(\mathrm{n}^{m,v})=\begin{bmatrix} \dot{f}^{m,v}(\mathrm{n}_1^{m,v}) & 0 & \cdots & 0 \\ 0 & \dot{f}^{m,v}(\mathrm{n}_2^{m,v}) & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \dot{f}^{m,v}(\mathrm{n}_{S^{m,v}}^{m,v}) \end{bmatrix} \tag{129}$$

藉由使用在矩陣形式中之連鎖律，寫出靈敏度的遞迴關係：

$$\delta^{m,v}=\frac{\partial F}{\partial \mathrm{n}^{m,v}}=\left(\frac{\partial \mathrm{n}^{m+1,v}}{\partial \mathrm{n}^{m,v}}\right)^{T}\frac{\partial F}{\partial \mathrm{n}^{m+1,v}}=\tilde{\mathrm{F}}^{m}(\mathrm{n}^{m,v})(\mathrm{w}^{m+1,v})^{T}\frac{\partial F}{\partial \mathrm{n}^{m+1,v}} \tag{130}$$
$$=\tilde{\mathrm{F}}^{m}(\mathrm{n}^{m,v})(\mathrm{w}^{m+1,v})^{T}\delta^{m+1,v}$$

最後，介紹靈敏度的起始點δ^{M}

$$\delta_i^M=\frac{\partial F}{\partial n_i^M}=\frac{\partial\sum_{j=1}^{S^M}\left(t_j-a_j^M\right)^2}{\partial n_i^M}=-2\left(t_i-a_i^M\right)\frac{\partial a_i^M}{\partial n_i^M} \tag{131}$$
$$=-2\left(t_i-a_i^M\right)\frac{\partial f^M(n_i^M)}{\partial n_i^M}=-2\left(t_i-a_i^M\right)\tilde{f}^M(n_i^M)$$

其中，

$$\tilde{f}^M(n_i^M)=\frac{\partial f^M(n_i^M)}{\partial n_i^M} \tag{132}$$

（131）式若以矩陣形式表示，則可寫成：

$$\delta^M=-2\tilde{\mathrm{F}}^M(\mathrm{n}^M)\left(\mathrm{t}-\mathrm{a}^M\right) \tag{133}$$

令

$$\delta^{M-1}=\left[\delta^{M-1,1}\quad\delta^{M-1,2}\quad\cdots\quad\delta^{M-1,V}\right] \tag{134}$$

根據（133）式，則

$$\delta^{M-1}=\tilde{\mathrm{F}}^{M-1}(\mathrm{n}^{M-1})(\mathrm{w}^M)^T\delta^M \tag{135}$$

由上式可知，靈敏度從最後一層到第一層，被倒向傳遞傳回網路，如下所示：

$$\delta^M\rightarrow\begin{Bmatrix}\delta^{M-1,1}\\ \delta^{M-1,2}\\ \vdots\\ \delta^{M-1,V}\end{Bmatrix}\rightarrow\cdots\rightarrow\begin{Bmatrix}\delta^{2,1}\\ \delta^{2,2}\\ \vdots\\ \delta^{2,V}\end{Bmatrix}\rightarrow\rightarrow\begin{Bmatrix}\delta^{1,1}\\ \delta^{1,2}\\ \vdots\\ \delta^{1,V}\end{Bmatrix} \tag{136}$$

3.1.3 系統演算步驟

綜合上述，平行式類神經網路演算摘要如下：

Step1. 隨機選用倒傳遞神經網路的參數初始值，適當的學習速率值。

Step2. 將輸入資料經由網路順向傳遞至輸出層。

$$\mathrm{a}^{M}=\mathrm{y} \tag{137}$$

$$\mathrm{a}^{0,v}=\mathrm{I}^{v},\quad v=1,2,\cdots V \tag{138}$$

$$\mathrm{a}^{m+1,v}=\begin{cases} f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\mathrm{b}^{m+1,v}) \\ f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\mathrm{LW}^{m+1,1}\mathrm{a}^{m+1,1}+\mathrm{LW}^{m+1,2}\mathrm{a}^{m+1,2}+\cdots+\mathrm{LW}^{m+1,V-1}\mathrm{a}^{m+1,V-1}+\mathrm{b}^{m+1,v}) \end{cases}$$

$$=\begin{cases} f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\mathrm{b}^{m+1,v}) & ,\ v=1,2,\cdots,V-1 \\ f^{m+1,v}(\mathrm{W}^{m+1,v}\mathrm{a}^{m,v}+\sum_{v=1}^{V-1}\mathrm{LW}^{m+1,v}\mathrm{a}^{m+1,v}+\mathrm{b}^{m+1,v}) & ,\ v=V \end{cases}$$

$$,\quad m=0,1,\cdots M-1 \tag{139}$$

Step3. 將靈敏度經由網路倒傳遞至各層。

$$\delta^{M}=-2\tilde{\mathrm{F}}^{M}(\mathrm{n}^{M})\left(\mathrm{t}-\mathrm{a}^{M}\right) \tag{140}$$

$$\delta^{m,v}=\tilde{\mathrm{F}}^{m}(\mathrm{n}^{m,v})(\mathrm{w}^{m+1,v})^{T}\delta^{m+1,v},\quad m=0,1,\cdots M-1 \tag{141}$$

Step4. 更新權值（weights）和偏權值（biases）。

$$\mathrm{W}^{m,v}(k+1)=\mathrm{W}^{m,v}(k)-\alpha\cdot\delta^{m,v}\left(\mathrm{a}^{m-1,v}\right)^{\mathrm{T}}\quad ,v=1,2,\cdots,V \tag{142}$$

$$\mathrm{LW}^{m,v}(k+1)=\mathrm{LW}^{m,v}(k)-\alpha\delta^{m,v}\left(\mathrm{a}^{m,v}\right)^{\mathrm{T}}\quad ,v=1,2,\cdots,V-1 \tag{143}$$

$$\mathrm{b}^{m,v}(k+1)=\mathrm{b}^{m,v}(k)-\alpha\delta^{m,v},v=1,2,\cdots,V \tag{144}$$

Step5. 重覆 *Step2* 至 *Step4*，直到系統的性能指標之值達到要求，或訓練次數達到預設值才停止。

3.2 系統建立與規劃

在預測之前，必須先利用歷史數據建立網路的訓練資料，而訓練資料建立的優劣直接影響了預測精度。此外，類神經網路內的一些參數，如學習速率對預測精度也佔了極重要的角色。因此，必須在預測之前對訓練資料的選擇與參數的設計予與詳盡的規劃。圖 3.2 所示的為本次實驗的網路規劃流程圖，包括了選擇訓練資料、數值正規化以及網路參數設計。

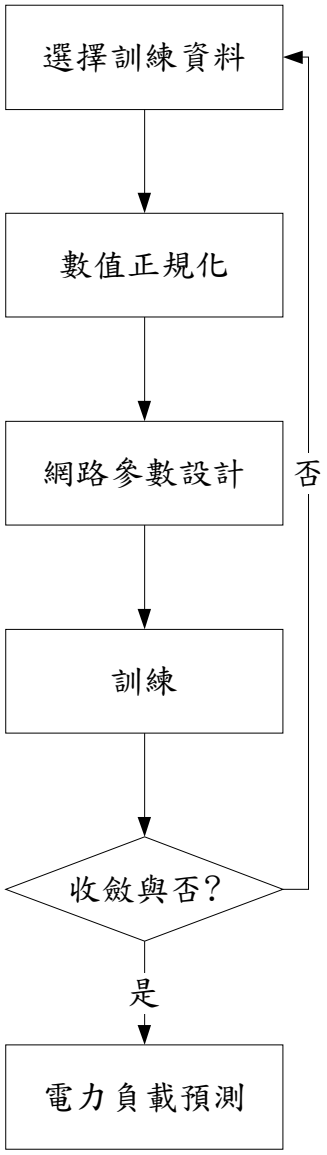

圖 3.2 平行式類神經網路電力負載預測系統規劃流程圖

3.2.1 訓練資料選擇

類神經網路運用的觀念就是學習輸入變數與輸出值之間的關係，所以輸入變數的選擇直接地影響到學習的精度；若採用的輸入變數不適當，不但網路難以收斂，學習的時間延長，更難以得到精確的預測結果。以用電預測而言，隨著預測週期不同，所考慮的輸入變數便不相同。以中長期為例，所考慮的變數通常包括電費、國民生產毛額、人口變動與氣候因子等；而短期負載預測所考慮的就有氣溫、日子型態（工作日、週末假日等）或小時指數等因素。在決定輸入變數時，通常是利用關聯性分析（correlation analysis），或憑藉著操作者的經驗而定。但總而言之，並沒有一定建模的規則可適用於所有的用電預測型態。

由於類神經網路的性能與訓練資料的建立有極大的關係，若訓練樣本相似性高，所訓練出來的網路對於與訓練資料相似的測試資料固然效果良好，但不具一般性，對於預測不同型態的用電趨勢時效果即會降低；所以提供網路學習不同的樣本特性，包含了適度的雜訊，以降低訓練樣本的同質性也是必要的。但是當訓練樣本同時存在幾種不同的特性，有可能某些情況的特徵太弱，產生了混淆的狀況，便難以得到精確的規則。為了避免這種的狀況發生，還是得適度將訓練樣本分類，由不同的網路分別學習。

關聯度分析在多變數分析中佔了極重要的角色。兩個變數之間，可以由關聯度係數 γ 來確定兩者之間的關聯度。一般而言，關聯度的強弱可藉由下列的規則來判定：

- ◆ $1 \geq \gamma > 0.7$ ：兩變數之間有高度的正相關性
- ◆ $0.7 \geq \gamma > 0$ ：兩變數之間有低度的正相關性
- ◆ $\gamma = 0$ ：兩變數之間沒有任何的線性相關性
- ◆ $0 > \gamma \geq 0.7$ ：兩變數之間有低度的負相關性
- ◆ $-0.7 > \gamma \geq -1$ ：兩變數之間有高度的負相關性

若兩變數 X 和 Y 之間的關聯度係數為 γ_{XY} ，則

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{145}$$

$$\overline{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i \tag{146}$$

$$Cov_{XY} = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y}) \tag{147}$$

$$S_X^2 = Cov_{XX} = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})^2 \tag{148}$$

$$S_Y^2 = Cov_{YY} = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \overline{Y})^2 \tag{149}$$

$$\gamma_{XY} = \frac{Cov_{XY}}{S_X S_Y} \tag{150}$$

其中

$\overline{X}$ ：X 的平均值

$\overline{Y}$ ：Y 的平均值

S_Y^2 ：X 的變異數

S_X^2 ：Y 的變異數

Cov_{XY} ：X 和 Y 之間的共變數

γ_{XY} ：X 和 Y 之間的關聯度

3.2.2 數值正規化

在由歷史數據獲得網路的訓練資料後，必須將樣本內的數據轉換成網路的輸入與輸出訊號，以做為學習過程之輸入值。在類神經網路中，非線性轉換函數是用來決定神經運算元收到訊號，經轉換後是否要輸出訊號，或者輸出訊號的強弱。

運算元的上下限值的決定與所選擇的轉換函數有關，一般而言，原始數據的極大值可轉換為運算元的上限值，極小值可轉換為運算元的下限值，而其他的數據就座落於這範圍內。但為了防止實際運算時，數據值會超過此範圍，所以必須預留運算的充裕度，以防止運算時運算元飽和的現象發生。

由於本研究的隱藏層轉換函數是採用如圖 3.3 所示的雙彎曲正切函數（Hyperbolic Tangent）轉移函數，這種函數當自變數趨於正負無限大時，函數值趨於常數，值域在[-1，1]之間。所以輸入變數必須先經由正規化，轉換成介於-1 到 1 之間的數值。正規化方法如下：

$$y = 2 \times \frac{(x - Min)}{(Max - Min)} - 1 \qquad (151)$$

其中

x ：原始數據

Max：原始數據的最大值

Min：原始數據的最小值

y ：原始數據經轉換後之數據

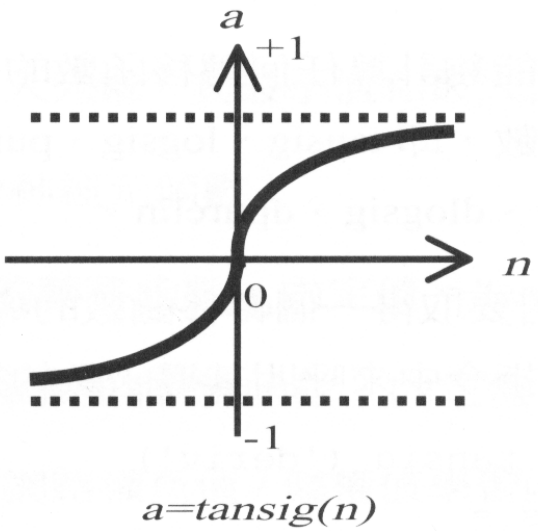

圖 3.3 雙彎曲正切函數

3.2.3 網路參數設計

在類神經網路之中有些參數，例如學習速率、隱藏層神經元數目以及隱藏層數目必須事先規劃。然而目前仍未有明確法則來定義類神經網路各參數之設定規則，以找到最佳網路形態；所以，本研究僅依下列方式設定各參數，以試誤法來進行比較。

1. 隱藏層數目

Pratap 在其應用類神經網路於預測的研究上建議，為了減少網路複雜性，縮短程式訓練時間，隱藏層可先設定為單層[30]。因此本研究以單層隱藏層進行分析。

2. 隱藏層節點數

隱藏層神經元數目的選擇並沒有一定的準則，一般而言，如果問題複雜性高，神經元數目宜多；如果網路測試結果誤差遠高於訓練範例的誤差，神經元數目宜減少。Sharda 及 Patil 在其應用類神經網路於預測的研究上，建議類神經網路隱藏層節點數等於輸入層節點數[31]。因此本研究隱藏層節點數等於輸入層節點數。

第四章 系統實證

本章節所要探討的重點，在於探討依據第三章所述的系統規劃流程進行系統實證，並探討平行式類神經網路與其他方法對於電力負載預測的適用性比較。圖 4.1 為系統實證步驟流程圖。

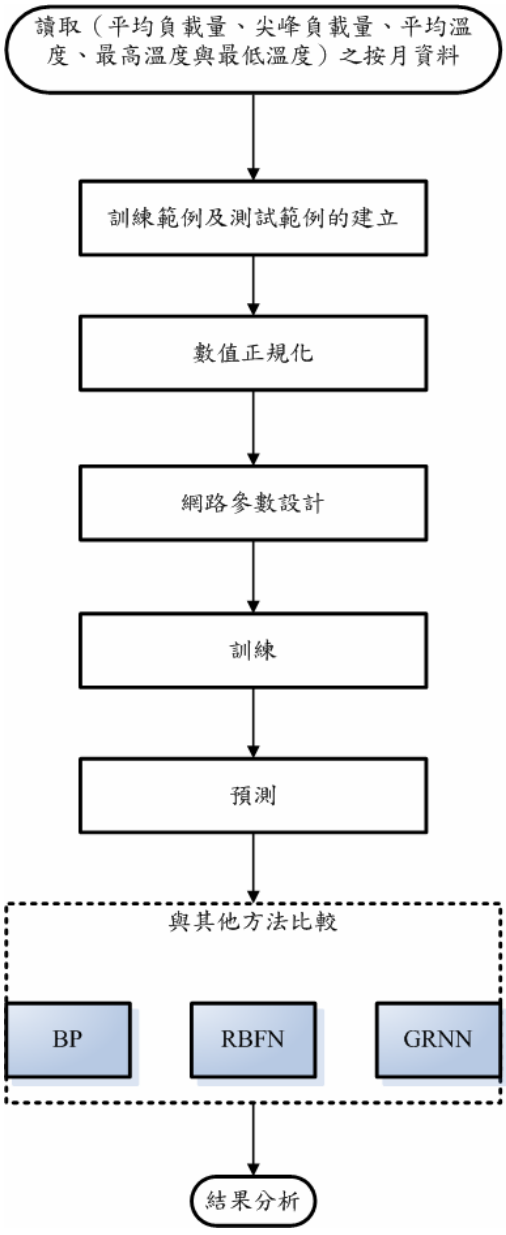

圖 4.1 系統實證步驟流程圖

4.1 訓練範例及測試範例的建立

由於本研究預測的項目，主要為未來 12 個月逐月的平均電力負載量，故本研究考慮之輸入變數包括（1）電力負載量與（2）溫度。其中電力負

載量變數包括平均負載量與尖峰負載量；而溫度包括平均溫度、最高溫度與最低溫度。本研究網路輸入變數如表 4.1 所示。

表 4.1 網路輸入變數表

輸入變數	變數
Input1 〈電力負載量〉	平均負載量 尖峰負載量
Input2 〈溫度〉	平均溫度 最高溫度 最低溫度

此外，本研究分別選取了前 12 個月、前 24 個月與前 36 個月電力負載量與溫度的歷史資料作為輸入單元，依此建立三個平行式類神經網路架構。希望利用類神經網路的非線性推論能力，分析出待預測的項目受前幾期歷史資料之複雜交互作用，藉此進行預測。

由於本研究預測的項目為民國 91 年 1 月至民國 91 年 12 月逐月的平均負載量，因此在訓練範例及測試範例的建立上，本研究分別選取民國 89 年 1 月至民國 89 年 12 月、民國 88 年 1 月至民國 89 年 12 月、民國 87 年 1 月至民國 89 年 12 月每個月之平均負載量、尖峰負載量、平均溫度、最高溫度與最低溫度做為網路之訓練範例，而以民國 90 年 1 月至民國 90 年 12 月每個月之平均負載量、尖峰負載量、平均溫度、最高溫度與最低溫度做為網路之測試範例，用以測試預測之準確度。網路訓練範例原始資料如表 4.2，測試範例原始資料如表 4.3 所示。

圖 4.2 為負載量訓練資料圖，而圖 4.3 為溫度訓練資料圖。根據關聯度分析，得到平均負載量與尖峰負載量之間的關連度係數為 0.894031；平均負載量與平均溫度之間的關連度係數為 0.862；平均負載量與最高溫度之間的關連度係數為 0.81411；平均負載量與最低溫度之間的關連度係數為 0.852241。因此代表尖峰負載量、平均溫度、最高溫度、最低溫度與平均負載量之間的關連度均有高度的正相關性。故在本研究所考慮之輸入變數均適合當作影響變因。

表 4.2 網路訓練原始資料表

時間	數入變數				
	Input1（電力負載量）單位：千瓦		Input2（溫度）單位：℃		
	平均負載量	尖峰負載量	平均溫度	最高溫度	最低溫度
87.1	13103000	17288000	17.6	29.4	9.5
87.2	14145000	17243000	17.8	28.8	9.5
87.3	14595000	18060000	20.7	32.4	13.6
87.4	15873000	20450000	25.2	33.6	16.2
87.5	16909000	21334000	27	33.6	21.5
87.6	17832000	22703000	27.7	35.8	22.3
87.7	19438000	23830000	29.4	35.5	22.4
87.8	19525000	23788000	29.1	35.2	23.4
87.9	17697000	22108000	27.6	34.1	22.1
87.10	16291000	21725000	25.9	32.9	19.9
87.11	16291000	21725000	23.6	31.2	16
87.12	14769000	18637000	20.1	29.1	12.9
88.1	14542000	17783000	17.8	28	11
88.2	13364000	17634000	18	29.8	6.8
88.3	15437000	19266000	21.5	32.8	13.7
88.4	16150000	19972000	24.4	32	17.4
88.5	16651000	22068000	24.9	33.6	16.4
88.6	18743000	24025000	28.1	34.3	23.1
88.7	19378000	24206000	28	35.1	23.3
88.8	19243000	23879000	27.8	35.9	22.7
88.9	17241000	23421000	27.7	34	21.4
88.10	16755000	21996000	26	32.8	20.3
88.11	16306000	19688000	22.5	30.2	15.3
88.12	15661000	19023000	17.6	27.7	4.7
89.1	15563000	18932000	17.5	28.6	9.7
89.2	14594000	18489000	16.8	26.3	11.2
89.3	16063000	19554000	19.5	29.7	13.8
89.4	16713000	22031000	23.2	31.1	15.1
89.5	18237000	24270000	26.2	34.4	19.8
89.6	20129000	25834000	27.8	34.9	18.4
89.7	20644000	25854000	28.6	35.5	22.6
89.8	20029000	25290000	27.6	34.2	22.9
89.9	19294000	24430000	27.4	34.1	22
89.10	19007000	24111000	26.5	34	20.7
89.11	17136000	21393000	22.8	31	16.2
89.12	16261000	19841000	20.1	28.4	13.5

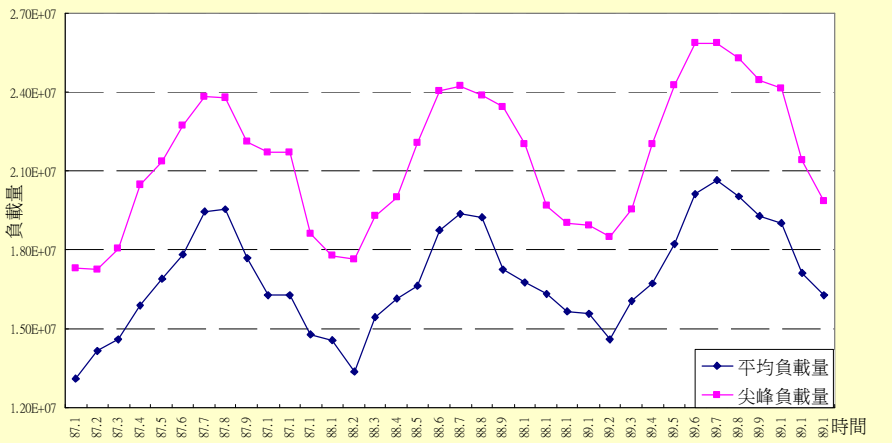

圖 4.2 負載量訓練資料圖

表 4.3 網路測試原始資料表

時間	數入變數				
	Input1（電力負載量）單位：千瓦		Input2（溫度）單位：℃		
	平均負載量	尖峰負載量	平均溫度	最高溫度	最低溫度
90.1	14769000	19554000	17.7	28.2	10.6
90.2	16293000	19795000	18.7	30	11.7
90.3	16758000	20359000	20.9	32.5	11.6
90.4	17199000	21860000	23	33.1	15.4
90.5	19128000	23997000	26.6	33.5	21.1
90.6	20083000	25454000	27.8	34.6	22
90.7	20869000	26177000	28.4	36.1	22.7
90.8	21767000	26290000	29.2	35.3	23.7
90.9	18467000	23359000	26.7	33.8	21.5
90.10	17820000	21952000	25.1	32.1	19.5
90.11	16832000	21033000	20.7	32.4	11.7
90.12	16378000	20381000	18.7	30.5	6.7

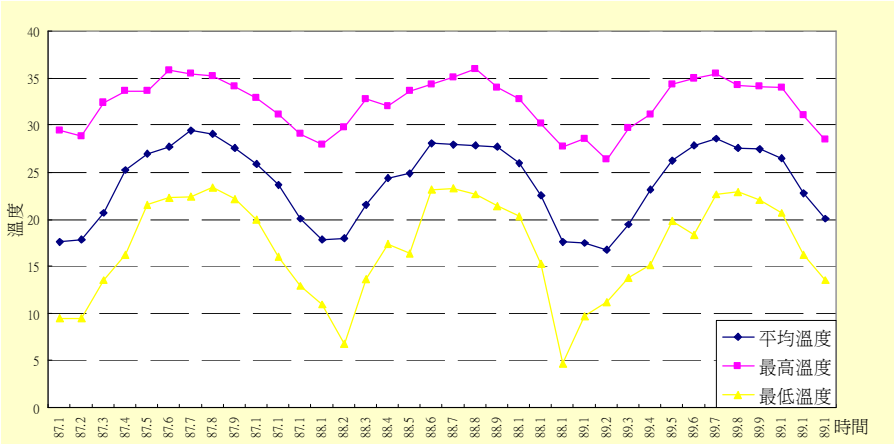

圖 4.3 溫度訓練資料圖

4.2 網路參數設計

圖 4.4 為系統實證網路架構圖，其中變數一為電力負載量變數，變數二為溫度變數；變數一的隱藏層轉移函數為正切雙彎曲轉移函數，變數二的隱藏層轉移函數為正切雙彎曲轉移數，輸出層的轉移函數為線性轉移函數；變數一的隱藏層數目為單層，變數二的隱藏層數目為單層，變數一的隱藏層節點數為 12，變數二的隱藏層節點數為 12；網路訓練方法為 L-M 演算法搭配 BasDavid Mackay 的 Bayesian 結構，網路性能目標為 10^{-10}，網路訓練的最大循環次數為 6000，整個網路參數設計如表 4.4 所示。

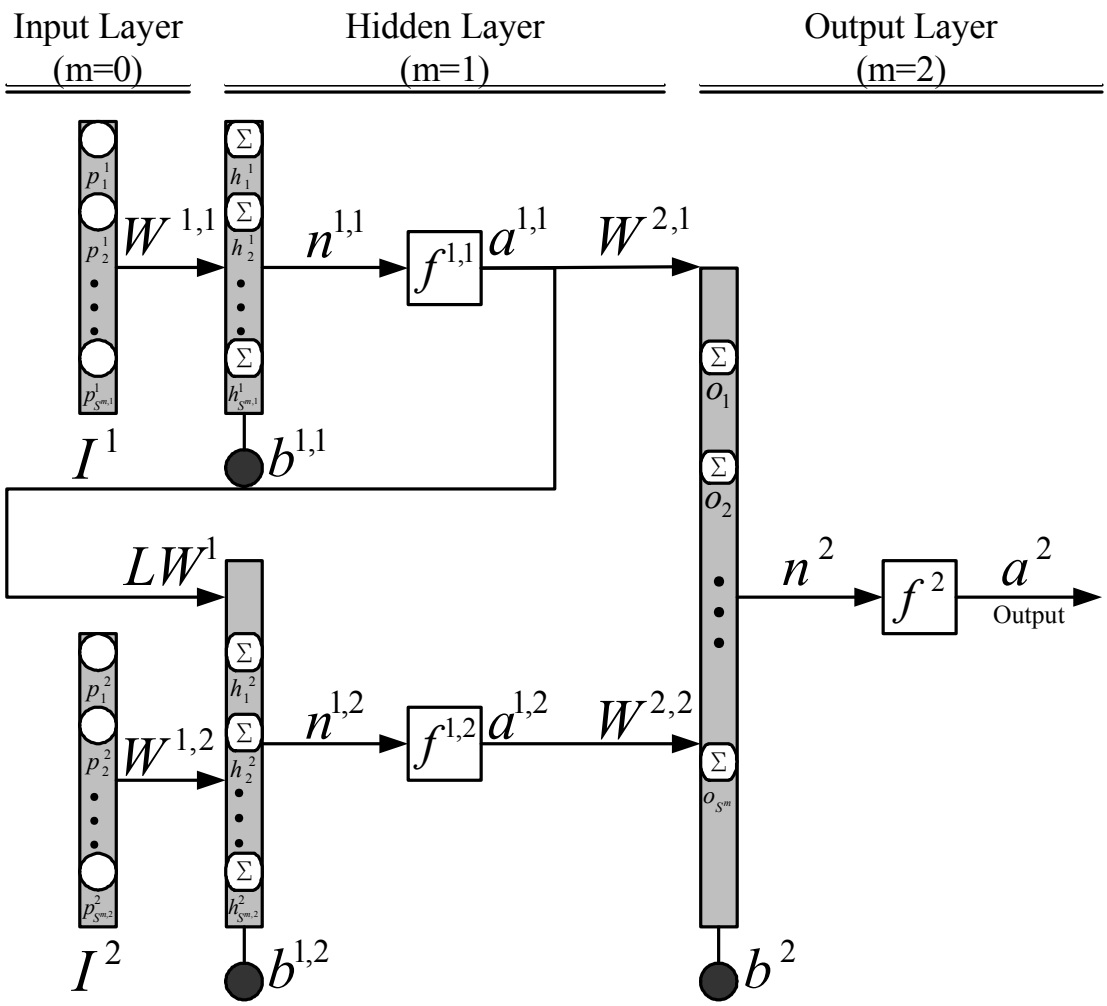

圖 4.4 系統實證網路架構圖

表 4.4 網路參數設計表

變數一	隱藏層轉移函數	正切雙彎曲轉移函數
	隱藏層數目	1
	隱藏層節點數	12
變數二	隱藏層轉移函數	正切雙彎曲轉移函數
	隱藏層數目	1
	隱藏層節點數	12
輸出層	轉移函數	線性轉移函數
網路訓練參數	訓練方法	L-M 演算法搭配 Bayesian 結構
	網路性能目標	10^{-10}
	最大循環次數	6000
L-M 演算法參數	μ 的初始值	0.001
	μ 的減少係數	0.1
	μ 的增加係數	10
	μ 的最大值	10^{10}

4.3 實驗結果與分析

本研究利用絕對平均誤差（Mean Absolute Percentage Error, MAPE）

進行預測效益評估，而 MAPE 越小者代表其預測能力較為精確。

4.3.1 訓練結果分析

表 4.5、表 4.6 與表 4.7 分別為利用平行式類神經網路利用過去 12 個月、24 個月與 36 個月不同的訓練範例所得之訓練結果，並將結果分別繪製如圖 4.5、圖 4.6 與圖 4.7 所示。

表 4.5 平行式類神經網路訓練結果 1（單位：千瓦）

日期	實際值	PNN
88.01	14769000	15677000
88.02	16293000	15703000
88.03	16758000	16201000
88.04	17199000	17099000
88.05	19128000	18805000
88.06	20083000	19986000
88.07	20869000	20528000
88.08	21767000	20200000
88.09	18467000	19513000
88.10	17820000	19140000
88.11	16832000	17118000
88.12	16378000	16339000
MAPE		**3.32%**

表 4.6 平行式類神經網路訓練結果 2（單位：千瓦）

日期	實際值	PNN
88.01	15563000	15447000
88.02	14594000	15000000
88.03	16063000	16283000
88.04	16713000	17344000
88.05	18237000	17819000
88.06	20129000	19903000
88.07	20644000	20051000
88.08	20029000	19860000
88.09	19294000	19252000
88.10	19007000	18459000
88.11	17136000	16837000
88.12	16261000	15538000
89.01	14769000	15692000
89.02	16293000	15567000
89.03	16758000	16283000
89.04	17199000	17446000
89.05	19128000	19178000
89.06	20083000	19942000
89.07	20869000	20599000
89.08	21767000	20281000
89.09	18467000	19857000
89.10	17820000	19447000
89.11	16832000	17447000
89.12	16378000	16447000
MAPE		**2.89%**

表 4.7 平行式類神經網路訓練結果 3（單位：千瓦）

日期	實際值	PNN
88.01	14542000	14522000
88.02	13364000	14743000
88.03	15437000	15215000
88.04	16150000	16655000
88.05	16651000	16599000
88.06	18743000	18479000
88.07	19378000	19472000
88.08	19243000	19528000
88.09	17241000	18205000
88.10	16755000	17557000
88.11	16306000	15983000
88.12	15661000	15562000
89.01	15563000	15117000
89.02	14594000	14576000
89.03	16063000	15911000
89.04	16713000	16615000
89.05	18237000	17641000
89.06	20129000	19436000
89.07	20644000	19668000
89.08	20029000	19432000
89.09	19294000	18674000
89.10	19007000	17840000
89.11	17136000	16488000
89.12	16261000	15633000
90.01	14769000	15749000
90.02	16293000	15463000
90.03	16758000	16299000
90.04	17199000	17660000
90.05	19128000	19254000
90.06	20083000	20343000
90.07	20869000	20631000
90.08	21767000	21309000
90.09	18467000	17720000
90.10	17820000	17444000
90.11	16832000	17526000
90.12	16378000	16515000
MAPE		**2.78%**

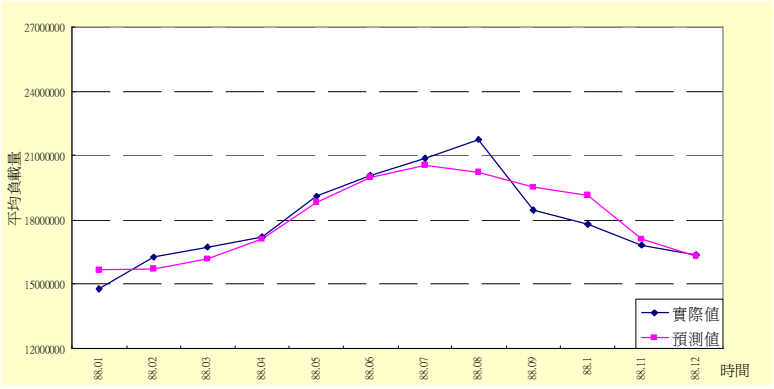

圖 4.5 平行式類神經網路訓練結果 1

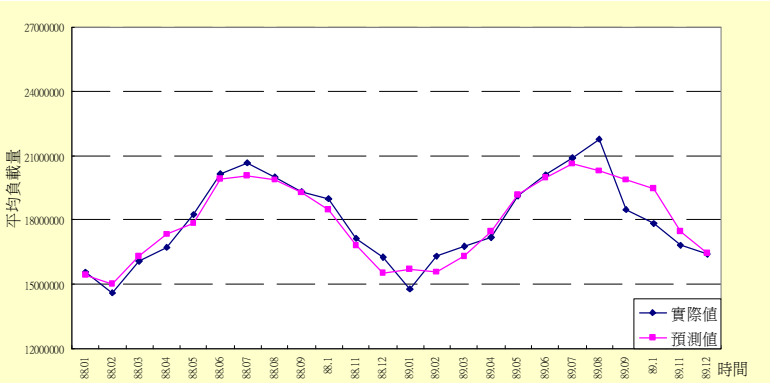

圖 4.6 平行式類神經網路訓練結果 2

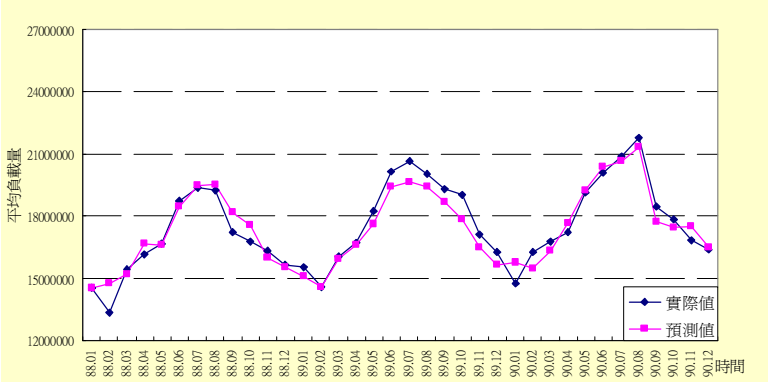

圖 4.7 平行式類神經網路訓練結果 3

4.3.2 測試結果分析

表 4.8、表 4.9 與表 4.10 分別為利用平行式類神經網路利用過去 12 個月、24 個月與 36 個月不同的訓練範例所得之測試結果，並將結果分別繪製如圖 4.8、圖 4.9 與圖 4.10 所示。

表 4.8 平行式類神經網路測試結果（單位：千瓦）

日期	實際值	PNN
91.01	16369000	15761000
91.02	15357000	16083000
91.03	17321000	16221000
91.04	18501000	17114000
91.05	19649000	19202000
91.06	21029000	20181000
91.07	21846000	20678000
91.08	22017000	21089000
91.09	20247000	18712000
91.10	19241000	17804000
91.11	17862000	16425000
91.12	17538000	15789000
MAPE		**6.22%**

表 4.9 平行式類神經網路測試結果（單位：千瓦）

日期	實際值	PNN
91.01	16369000	15746000
91.02	15357000	16138000
91.03	17321000	16482000
91.04	18501000	17477000
91.05	19649000	19512000
91.06	21029000	20254000
91.07	21846000	20666000
91.08	22017000	21050000
91.09	20247000	19318000
91.10	19241000	18413000
91.11	17862000	16615000
91.12	17538000	15961000
MAPE		**5.01%**

表 4.10 平行式類神經網路測試結果（單位：千瓦）

日期	實際值	PNN
91.01	16369000	15865000
91.02	15357000	16363000
91.03	17321000	16671000
91.04	18501000	17660000
91.05	19649000	19460000
91.06	21029000	20340000
91.07	21846000	20789000
91.08	22017000	21097000
91.09	20247000	18994000
91.10	19241000	18107000
91.11	17862000	17013000
91.12	17538000	16421000
MAPE		**4.67%**

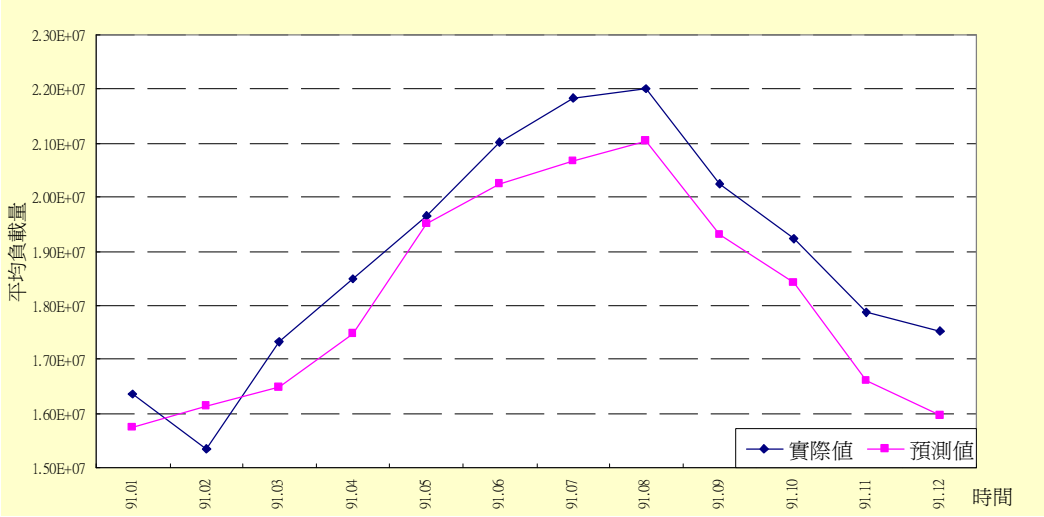

圖 4.8 平行式類神經網路測試結果 1

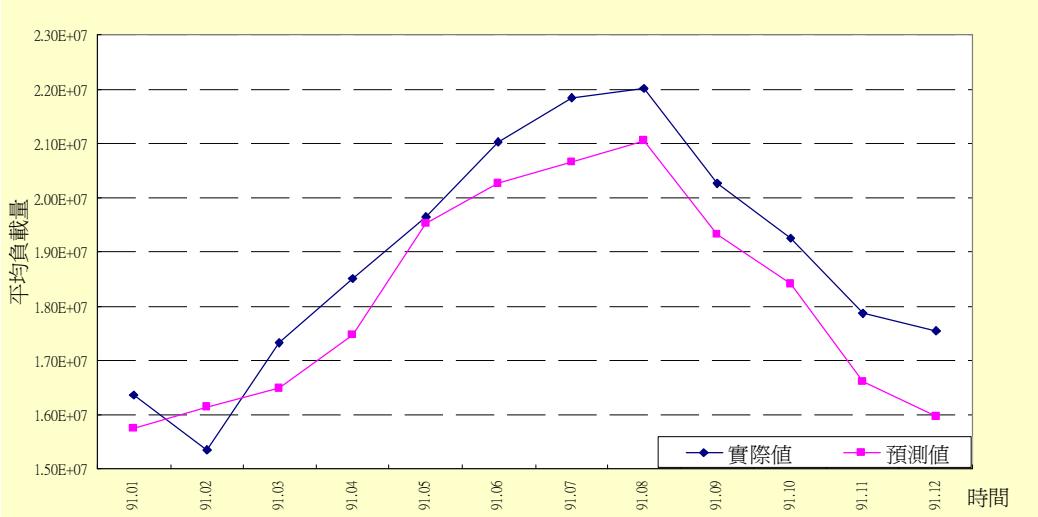

圖 4.9 平行式類神經網路測試結果 2

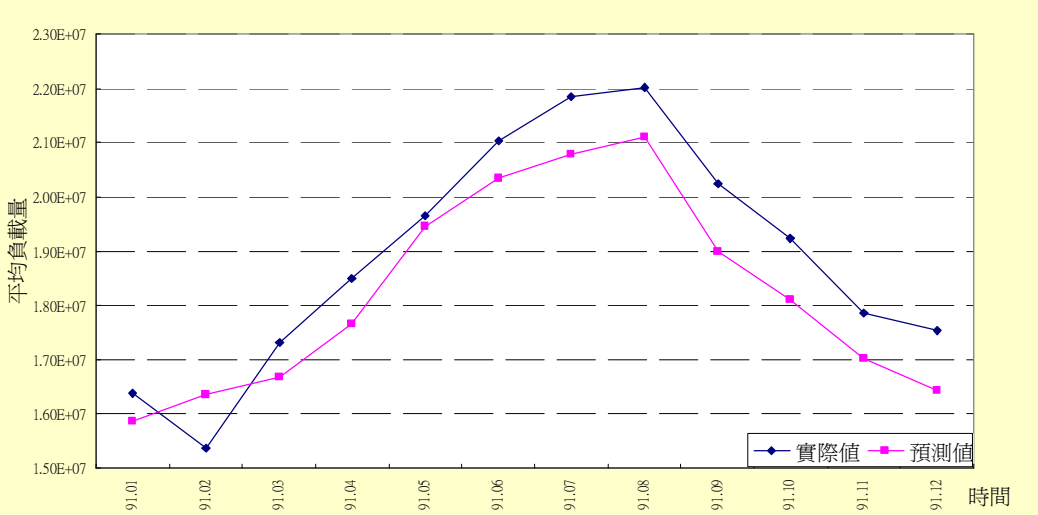

圖 4.10 平行式類神經網路測試結果 3

配合圖 4.8、圖 4.9 與圖 4.10 的觀察，本研究發現，平行式類神經網路的圖形，普遍較有平滑（smooth）的現象，而在預測上，似乎在初始波峰預測擬合時表現較佳，而在末期波峰預測有逐漸低估的現象。

表 4.12 為各平行式類神經網路架構之訓練範例及測試範例 MAPE 之記錄。由結果顯示，當輸入單元增加時，各範例 MAPE 普遍有下降的情形。此外，由於根據歷史資料的擬合至實際的預測常發生誤差放大的情形，故訓練範例之 MAPE 有小於測試範例之 MAPE 的現象。

表 4.11 平行式類神經網路訓練範例與測試範例之絕對平均誤差

MAPE（%）	以前 12 個月歷史資料作為輸入單元	以前 24 個月歷史資料作為輸入單元	以前 36 個月歷史資料作為輸入單元
訓練範例	3.32	2.89	2.78
測試範例	6.22	5.01	4.67

4.3.3 與其他方法比較結果分析

表 4.12 為分別利用平行式類神經網路（PNN）、傳統倒傳遞網路（BP）、徑向基底網路（RBFN）、一般迴歸神經網路（GRNN）對於測試範例（民國 91 年 1 月至民國 91 年 12 月）之實驗結果，以及絕對平均誤差之紀錄，並將結果繪製如圖 4.11。

表 4.12 與其他方法比較結果（單位：千瓦）

日期	實際值	PNN	BP	RBF	GRNN
91.01	16369000	15865000	16611000	15421000	15315000
91.02	15357000	16363000	16229000	15666000	15905000
91.03	17321000	16671000	16280000	18173000	16521000
91.04	18501000	17660000	17409000	18179000	17212000
91.05	19649000	19460000	17826000	20548000	19298000
91.06	21029000	20340000	21514000	22578000	20394000
91.07	21846000	20789000	20808000	26183000	20539000
91.08	22017000	21097000	19794000	24182000	20308000
91.09	20247000	18994000	17774000	23178000	18928000
91.10	19241000	18107000	17570000	19173000	17684000
91.11	17862000	17013000	16905000	18167000	16617000
91.12	17538000	16421000	16452000	18159000	16101000
MAPE（%）		**4.67%**	**6.97%**	**6.39%**	**6.17%**

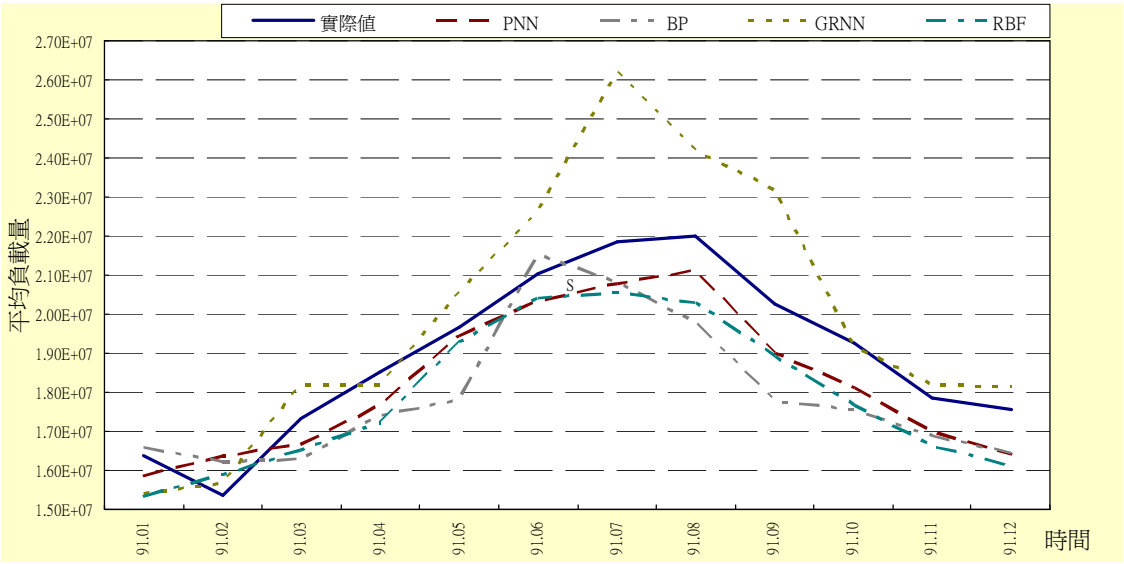

圖 4.11 與其他方法比較結果

根據預測結果發現，平行式類神經網路架構之 MAPE 為 4.67%，較傳統倒傳遞網路（6.97%）、徑向基底網路（6.39%）、一般迴歸神經網路（6.17%）均為低，而 MAPE 越小者代表其預測能力較為精確，因此利用平行式類神經網路架構進行電力負載預測較其他三者方法更為精確。

第五章 結論與未來研究方向

5.1 結論

由於電力負載預測不論是在電力輸送作業、儲存能源調配、線上調控或緊急事故處理中均扮演著極為重要的角色。然而，傳統的類神經網路雖然能利用影響電力負載預測的相關變數建立訓練資料，卻不能針對各個相關變數間的關係進行探討並調整之。因此本研究針對電力負載預測之議題，架構相對應之平行式類神經網路電力負載預測系統，以改善利用傳統倒傳遞網路進行電力負載預測之缺失，並求得更精確的電力負載預測結果。經實驗驗證，平行式類神經網路整體的絕對平均誤差大約 4.67%左右，較其他方法均低，因此可證明平行式類神經網路較其他三者方法更為精確。

5.2 未來研究方向

由於本研究利用平行式類神經網路進行電力負載預測時，僅考慮負載量與溫度兩種主要影響電力負載之變數，然而其他變數包括電費與其他氣候因子或多或少還是影響電力負載之預測結果。因此，以輸入變數而言，未來研究方向可嘗試將其他變數納入網路予與考慮。

此外，平行式類神經網路除了可進行電力負載預測，也可嘗試應用於其他領域。因此，未來研究方向可應用平行式類神經網路進行其他領域之預測議題，以得到更精確之預測結果。

參考文獻

[1] 王進德、蕭大全，1999。類神經網路與模糊控制理論入門，全華科技圖書，台北。

[2] 危家康，2003。以類神經網路模擬受純扭力作用下鋼筋混凝土梁之強度，國立成功大學土木工程學系，碩士論文。

[3] 陳惠玉，2004。模糊徑向基網路及其在授信評等之應用，大葉大學工業工程學系，碩士論文。

[4] 葉怡成，2000。類神經網路模式應用與實作，儒林圖書，台北。

[5] 葉瑞峰，2003。以類神經網路對高雄地區調頻廣播電台電場強度估測之研究，義守大學機工程學系，碩士論文。

[6] 劉亭宜，2000。GRNN在晶圓製造裡良率模式之建構與分析，元智大學工業工程所，碩士論文。

[7] A. G. Bakirtzls et al, 1996. “A Neural Network Short Term Load Forecasting Model for the Greek Power System”, *IEEE Transactions on Power Systems*, Vol. 11, No. 2, pp. 858-863.

[8] A. Khotanzad et al, 1995. “An Artificial Neural Network Hourly Temperature Forecaster with Applications in Load Forecasting”, *IEEE Transactions on Power Systems*, Vol. 11, No. 2, pp. 870-876.

[9] A. Khotanzad, A. R. Reza and T. L. Lu, 1997. “ANNSTLF-A Neural-Network-Based ElectricLoad Forecasting System”, *IEEE Transactions on Neural Networks*, Vol. 8, No. 4, pp. 835-846.

[10] A. Khotanzad, R. Afkhami-Rohani and D. Maratukulam, 1998. “ANNSTLF – Artificial Neural Network based Short Term Load Forecaster – Generation Three”, *IEEE Transactions on Power Systems*, Vol. 13, No. 4, pp. 1413-1422.

[11] A. S. Chandrashekara, T. Ananthapadmanabha and A. D. Kulkarni , 1999. “A Neuron expert System for Planning and Load Forecasting of Distribution Systems”, *Electric Power and Energy Systems*, Vol. 21, pp. 309-314.

[12] B. Kermanshahi, 1998. “Recurrent Neural Network for Forecasting Next 10 Year Loads of Nine Japanese Utilities”, *Neurocomputing*, Vol. 23, pp. 125-133.

[13] C. N. Lu, H. T. Wu and S. Vemuri, 1993. “Neural Network based Short Term Load Forecasting," *IEEE Transactions on Power Systems*, Vol. 8, No. 1, pp. 336-342.

[14] C.C. Chin, L.J. Kao and D.F. Cook, 1997. “Combining a Neural Network with a Rule-Base Expert System Approach for Short-Term Power Load Forecasting in Taiwan”, Expert System With Application, Vol. 13, No. 4, pp. 229-305.

[15] D. J. C. MacKay, 1992. “A Practical Bayesian Framework for Back-propagation Networks”, *Computation*, Vol. 4, No. 3, pp. 448-472.

[16] D. Srinivasan, A. C. Liew and J. S. P. Chen, 1991. “Short Term Forecasting Using Neural Network Approach”, The 1st International Forum on Applications of Neural Networks to Power Systems, Seattle, USA, pp. 12-16.

[17] E. A. Mohamad, M. M. Mansour, S. El-Debeiky, K. G. Mohamad and N. D. Rao, 1996. “Result of Egyptian Unified Grid Hourly Load Forecasting Using an Artificial Neural Network with Expert System Interface”, *Electric Power Systems Research*, Vol. 39, pp. 171-177.

[18] E. Doveh et al, 1999, “Experience with FNN models for medium term power demand predictions”, *IEEE Transactions on Power Systems*, Vol. 14, No. 2, pp. 538-546.

[19] El-Keib, X. Ma, H. Ma, 1995. “Advancement of Statistical Based Modeling Technique for Short-Term Load Forecasting”, *Electric Power Systems Research*, Vol. 35, pp. 51-58.

[20] H. B. Gooi et al, 1993. “Adaptive Short-term Load Forecasting Using Artificial Neural Networks”, IEEE Region 10 Conference on Computer, Communication, Control and Power Engineering, Beijing, China, Vol. 2, pp. 787-790.

[21] K. Lee, Y. Cha and J. Park, 1990. "Artificial neural network methodology for short term load forecasting", NSF Workshop on Artificial Neural Network Methodology in Power System Engineering, Clemson University.
[22] K. Liu et al, 1996. "Comparison of Very Short-term Load Forecasting Techniques", *IEEE Transactions on Power Systems*, Vol. 11, No. 2, pp. 877-882.
[23] K. Liu, S. Subbarayan, R. R. Shouts, M. T. Manry, C. Kwan, F. L. Lewis and J. Naccarino, 1996. "Comparison of Very Short-Term Load Forecasting Techniques", *IEEE Transactions on Power Systems*, Vol. 11, No. 4, pp. 877-882.
[24] L. D. Paarmann. and M. D. Najar, 1995. "Adaptive Online Load Forecasting Via Time Series Modeling", *Electric Power Systems Research*, Vol. 32, pp. 219-223.
[25] L. D. Voss, M. M. A. Salama and J. Reeve, 1995. "A Practical Approach to Electric Load Forecasting Using Artificial Neural Networks with Corrective Filtering", Canadian Conference on Electrical and Computer Engineering, Montreal, Canada, Vol. 1, pp. 370-373.
[26] M. Tamimi and R. Egbert, 2000. "Short Term Electric Load Forecasting via Fuzzy Neural collaboration", *Electric Power Systems Research*, Vol. 56, pp. 243-248.
[27] P. Caire, G. Hatabian and C. Muller, 1992. "Progress in Forecasting by Neural Networks", International Joint Conference on Neural Networks, Baltimore, USA, Vol. 2, pp. 540-545.
[28] P. Shamsollahi et al, 2001. "A Neural Network Based Very Short Term Load Forecaster for the Interim ISO New England Electricity Market System", The 22nd IEEE Power Engineering Society International Conference on Industry Computer Applications, Sydney, Australia, pp.217-222.
[29] Q. Chen et al, 2001. "Implementation and Performance Analysis of Very Short Term Load Forecaster Based on the Electronic Dispatch Project in ISO New England", 2001 Large Engineering Systems Conference on Power Engineering, Halifax, Canada, pp.98-104.
[30] R. J. Pratap, D. Staiculescu, S. Pinel, J. Laskar and G. S. May, 2005. "Modeling and sensitivity analysis of circuit parameters for flip-chip interconnects using neural networks", *IEEE Transactions on Advanced Packaging*, Vol. 28, pp. 71 – 78.
[31] R. Sharda and R. B. Patil, 1990. "Neural Networks as Forecasting Expert: An Empirical Test", International joint conference on neural networks, Vol. 2, pp. 491-494.
[32] S. H. Fan, T. Y. Xiao and C. Guo, 2004. "A model of the expanded grey theory", networking, Sensing and Control, 2004 IEEE International Conference on, Vol. 1, pp. 339-342.
[33] S. J. Kiartzis, A. G. Bakirtzis, V. Petridis, 1995. "Short-term Load Forecasting Using Neural Networks", *Electric Power Systems Research* , Vol. 33, pp. 1-6.
[34] S. M. Islam, S. M. Al-Alawi and K. A. Ellithy, 1995. "Forecasting Monthly Electric Load and Energy for a Fast Growing Utility Using an Artificial Neural Network", *Electric Power Systems Research*, Vol. 34, pp. 1-9.
[35] T. Al-Saba. and I. El-Amin, 1999. "Artificial Neural Networks as Applied to Long-term Demand Forecasting", *Artificial Intelligence in Engineer*, Vol. 13, pp. 189-197.
[36] T. M. Peng, N. F. Hubele and G. G. Karady, 1990. "Conceptual Approach to the Application of Neural Network for Short-term Load Forecasting", IEEE International Symposium on Circuits and Systems, New Orleans, USA, Vol. 4, pp. 2942-2945.
[37] T. M. Peng, N. F. Hubele and G. G. Karady, 1992. "Advantage in the Application of Neural Networks for Short Term Load Forecasting", IEEE Transactions on Power Systems, Vol. 17, No. 1, pp. 250-257.
[38] T. W. S. Chow and C. T. Leung, 1996. "Neural Network Based Short-Term Load ForecastingUsing Weather Compensation", *IEEE Transactions on Power Systems*, Vol. 11, No.4, pp. 1736-1742.
[39] T. W. S. Chow, C. T. Leung, 1996. "Neural network based Short-term Load Forecasting

Using Weather Compensation", IEEE Transactions on Power Systems, Vol. 11, No. 4, pp. 1736-1742.

[40] W. Charytoniuk and M. S. Chen, 2000. "Very Short-term Load Forecasting Using Artificial Neural Networks", *IEEE Transactions on Power Systems*, Vol. 15, No. 1, pp. 263-268.

[41] Y. H. Fung and V. M. Rao Tummala, 1993. "Forecasting of electricity consumption: acomparative analysis of regression and artificial neural network models", The 2nd International Conference on Advances in Power System Control, Operation and Management, Hong Kong, Vol. 2, pp. 782-787.