使用多階重入類神經網路辨識派群(Petri net)語言
Multi-Order Recurent Neural Networks for Petri Net Inference

李鴻璋、劉艾華、張文龍
Hung-Chang Lee, Ay-Hwa Andy Liou, Wen-Long Chang
淡江大學資訊管理研究所
Department of Information Management
Tamkang University, Tamshui, Taipei
Hclee@mail.im.tku.edu.tw

摘要

為驗證類神經網路辨認派群語言的可行性，本文是嘗試運用多階重入類神經網路來辨識派群語言，文中討論有無誤差標準值的影響、學習速率值的大小、學習速率折損因子大小、和狀態神經元數目多少的影響，由測試的錯誤率可相當的接近於零，顯示出二階重入類神經網路在派群語言辨識上的可行性。另提出三階重入類神經網路架構，並以其辨認有並行特性的 Petri net 語言。

關鍵字：多階重入類神經網路，派群，語言辨識

Abstract

This study starts the attempts of providing a practical method of recognizing Petri net language using neural networks. It also discusses the effect of error criteria. learning rate, learning reduction rate, and number of the state neuron. The error rate of test is very close to zero. This indicates that the trained neural network can learn to recognizing simple Petri net language, and that it is feasible to use neural networks to recognize Petri net language.

Key Word : Multi Order Recurrent Neural Network, Petri net Lang. Reg.

壹 緒論

本研究以 Giles et al.在 1990~1992 提出的類神經網路模型為基礎,加以適當的擴充,希望可用以辨認派群的語言。原本 Giles 等人所進行的測試只有 Tomita #4 這種文法,而且其所針對的有限狀態機(Finite State Automata)的字集(alphabet)是{0,1}之二維空間 ，本研究辨認字集超過三個元素以上有並行特性的 Petri net 語言。藉此我們可以更進一步瞭解此類神經網路的能力與 Petri net 的語言。

貳 文獻探討

關於類神經網路在文法辨識上的研究，較早的如 Cleeremans 在 1989 年[4]所提，用一階重入網路(First Order Recurrent Neural Network)預測一串連續字集元素,此網路是用時間 t-1 時的系統隱藏單元狀態上的行為樣式 和時間t之輸入元素，來預測時間t+1之輸入元素,其所訓練的網路,可以作為一個有限文法辨識器,但是其只能去預測下一個符號,而不是經由訓練之後,判斷字串是否能被接受。

而 Omlim et al. [5]則展示，經由群聚(clustering)二階重入類神經網路之輸出狀態空間 ,可以粹取出文法規則,並證明粹取出的文法規則和二階重入類神經網路的成效之間有相關性。

Giles et al. [6], 利用二階重入類神經網路來推論正規文法,並且由訓練完成的網路粹取出有限狀態機的原型。

Goodman et al. [9]在 1993 年的自群聚重入類神經網路學習有限狀態機內容的文章中，提議經由網路的離散化和虛擬坡度學習法則來訓練類神經網路，以強迫類神經網路學習穩定的狀態，此外此文章中指出一個二階重入類神經網路，可以經由兩個一階重入類神經網路再加上一個 NOT gate 的適當組合而得到，這可更清楚的顯示二階重入類神經網路的行為。

Noda et al. [8]則認為從使用類神經網路對人工智慧的觀點，根據最佳化控制的原理使用學習演算法是不適當的，主要是因為這些演算法與傳統的符號處理不符合。因此其提出 PEX model，此模型是從符合 Elman 網路[10-11]和有限狀態機之下的有限狀態機最小化過程所導出，但其仍是用以預測網路的下一個輸入。

還有一個與 Giles et al. 二階重入類神經網路相類似的二階重入類神經網路，是由 Pollack 所提出的 SCRN (seqeuntial cascaded recurrent network)[12-13]，此網路有 Context Network 和 Function Network 兩部分，也可以正確的辨認 Tomita #4 文法。但是 Giles 宣稱其網路有固定點行為，Pollack 的則沒有。

重入網路有很多種，除了二階重入類神經網路之外，倒傳遞網路也可以經由加入回饋的連結，而形成。此外還有部分重入網路和完全重入網路等等。其中部分重入網路又包含 Jordan 循序網路和簡單循序網路。一般說來，非對稱的重入網路都會有時空行為，即是能處理時空樣本。

參 基本理論介紹

時空樣本識別類神經網路

一般的類神經網路雖然也有處理時間訊息的資料的應用例子,但實際上他們是將時間訊息資料以處理空間訊息資料方式處理,並未真正處理具順序性的時間訊息資料。舉例來說,用倒傳遞網路預測天氣,假設第 n 日的天氣是前三天的降雨機率與溫度的函數[3],則用一般的倒傳遞網路架構來處理此預測問題時,這三天的資料是同時輸入,等於仍然是將順序性的時間訊息資料以處理空間訊息資料方式處理。

時空樣本識別網路則不同,它將時間訊息資料與空間訊息以不同的方式處理,例如網路只有兩個輸入處理單元,分別代表一個降雨機率與一個溫度資料構成一個二維的空間性樣本.而每個範例則分成三段,分別代表第 n-3 天、第 n-2 天、第 n-1 天資料, 則構成一個具三個時間軸上的時間性樣本。中間層的處理單元具有順序性,第 j 個中間層的處理單元不但受輸入向量的影響,也受第 1 個到第 j-1 個中間層的處理單元的影響。最後一個中間層的的處理單元的輸出值 則會傳送到一個輸出層處理單元如此的架構將使範例中各段樣本的輸入順序影響輸出層處理單元的輸出值,例如依序輸入第 n-2、n-1、n-3 天的降雨機率與溫度,和依序輸入第 n-3、n-2、n-1 天的降雨機率和溫度其結果是不同。
而多階重入類神經網路屬於時空樣本識別類神經網路的一種,其輸入的順序會影響輸出的結果,而且其輸入的資料每一筆都可以不一樣長。

派群 (Petri Net)

自從 1962 年 C.A. Petri 提出派群這個計算模型以後,經過許多學者的研究,派群被認為是能夠有效的模擬平行系統(concurrent system)的工具。派群的應用領域包含了計算機結構、通訊系統、資料庫及作業系統等等。其對於系統的模擬和分析的應用可以用圖 1[14]簡單的表示,系統首先被派群模擬,之後再對此模擬的模型加以分析,經由分析之後即可對此系統有更進一步瞭解,甚而修正或是提出更好的系統。

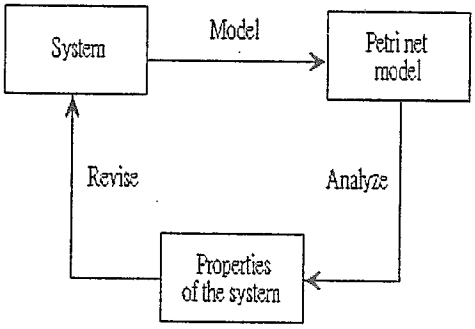

圖 1、派群應用圖

派群的定義,可以用一個五維的結構 C 表示, C=(P,T,I,O,U0)
1. P={p1,p2,p3,.....,pn}是位置 (place) 的有限集合。
2. T={t1,t2,t3,.....,tn}是移轉 (transition) 的有限集合。
3. I:T->P~是一個輸入函數。
4. O:T->P~是一個輸出函數。
5. U0:P->N 是一個函數,從位置的有限集合對應到非負的整數。

如圖 4 為一個有 3 個移轉(transition)與 4 個位置(place)的派群結構,可用上面的定義表示為:

C={P,T,I,O,U0}
P={p1,p2,p3,p4}
T={t1,t2,t3}

I(t1)={p1,p2,p3}
I(t2)={p4}
I(t3)={p3}

O(t1)={p1}
O(t2)={p2,p3}
O(t3)={p4}

U0={1,0,1,0}

肆 二階重入類神經網路

二階重入類神經網路架構

二階重入類神經網路的架構如圖 2 [6]所示:

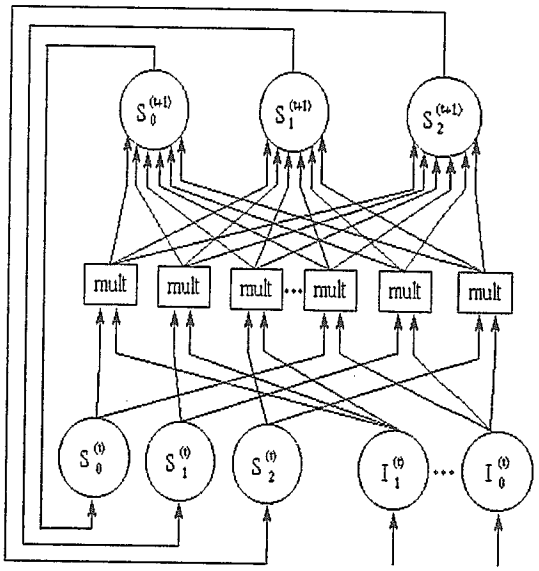

圖 2、 二階重入類神經網路的架構圖

此網路的重入之意,即是因為他的 N 個狀態神經元 (state neurons),會回饋到下一時間與不重入的輸入神經元 (input neurons) 一起當作輸入.而二階是因為它的加權值,是由狀態神經元與輸入神經元合起來再連到下一個狀態神經元,與一般的一階加權值只是連接兩個神經元有所不同,同理如果是連接四個神經元則其為三階類神經網路。

整個學習過程的重要公式如下所示:

$$net_i^{t+1} = \sum_{j,k} W_{ijk} S_j^t I_k^t$$

$$S_i^{t+1} = f(net_i^{t+1})$$

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

$$E_0 = \frac{1}{2}(T - S_0^t)^2$$

$$\Delta W_{lmn} = -\alpha \left(\frac{\partial E_0}{\partial W_{lmn}}\right)$$

$$= -\alpha (T - S_0^t) \frac{\partial S_0^t}{\partial W_{lmn}}$$

$$\frac{\partial S_i^{t+1}}{\partial W_{lmn}} = f(net_i^{t+1})$$

$$*\left[\delta_{il} S_m^t I_n^t + \sum_{j,k} W_{ijk} I_k^t \frac{\partial S_j^t}{\partial W_{lmn}}\right]$$

應用到派群上

原本此演算法的應用因為是在 FSA 上,且其字集只有 0 和 1,所以只要兩個輸入神經元即可 ,但是因為派群的輸入是移轉(transition),所以大部份都會大於二以上 ,因此我們要修改輸入神經元符合我們的需要 .我們使用多對一編碼:將資料對映到數個區間 ,每一區間用一處理單元代表之。

關於 ΔW_{lmn} 加權值的修正 ,運用下面的演算法加以實作 :

```
for (當 i 小於狀態神經元的最大值時){
  for (當 j 小於狀態神經元的最大值時){
    for (當 k 小於輸入神經元的最大值時){
      length = 轉的序列長度;
      ΔW [i][j][k] =學習速率*( T − S_0^f )*
       PD( S_0^f ,W[i][j][k])   //偏微分
    }}}
```

PD 偏微分函數內容如下:

```
PD( S_a^length ,W[i][j][k]){    //偏微分
  若長度為 0,則傳回 0;
  product = S_a^length (1 − S_a^length );
  for (當 b 小於狀態神經元的最大值時)
    for (當 c 小於輸入神經元的最大值時) {
      若[(a=i)且(b=j)且(c=k)] 時 {
          sum=sum+ S_b^length-1 I_c^length-1;
      }
      sum = sum+W[a][b][c] *  I_c^length-1
      * PD( S_b^length-1 ,W[i][j][k])
    }
  return(product * sum);
}
 }
```

研究結果與分析

範例描述

如圖 3 所示,一個有四個移轉(transition)與五個位置(place)的派群結構,可用前面的定義表示為:

C={P,T,I,O,U0}
P={p1,p2,p3,p4,p5}
T={t1,t2,t3,t4}

I(t1)={p1,p5}
I(t2)={p2,p5}
I(t3)={p3}
I(t4)={p4}

O(t1)={p3}
O(t2)={p4}
O(t3)={p1,p5}
O(t4)={p2,p5}

U0={1,1,0,0,1}

開始時,因為 p1,p2,p5 都有 token 在其中,所以 t1,t2 滿足被發射(fire)的條件,因此 t1 或 t2 都是可以被發射的,當 t1 被發射後 t2 則不能再被發射,此時的 u1={0,1,1,0,0},接下來只有 t3 能被發射,當 t3 被發射之後的狀態又回到原先的 u2={1,1,0,0,1},此時如果 t3 被發射則狀態變為 u3={1,0,0,1,0}接下來則只有 t4 能被發射,t4 發射之後,則狀態便又變為原來的 u4={1,1,0,0,1},繼續此種發射的規則。

根據此種發射規則,可推導其可能的合法字串為:

t1t3t2t4
t1t3t1t3
t2t4t2t4t1t3
t2t4t1t3t1t3t2t4
t1t3

不符合此種規則的都是不合法的字串,可能為:

t1t2t3t4
t1t1t2t2
t1t4t2t3t2t3
t4t2tt3t1

這些字串我們在利用前面的多對一編碼,則 t1t3t2t4 此一合法的字串如果要被系統所訓練,需被編碼為::

	輸入神經元			
	#1	#2	#3	#4
當時間為 1 時	1	0	0	0
當時間為 2 時	0	0	1	0
當時間為 3 時	0	1	0	0
當時間為 4 時	0	0	0	1

且因為是合法的字串,所以最後的輸出 T=1。

而圖 4 則為一個有 3 個移轉(transition)與 4 個位置(place)的派群結構,可用前面的定義表示為:

C={P,T,I,O,U0}
P={p1,p2,p3,p4}
T={t1,t2,t3}

I(t1)={p1,p2,p3}
I(t2)={p4}
I(t3)={p3}

O(t1)={p1}
O(t2)={p2,p3}
O(t3)={p4}

U0={1,0,1,0}

可推導其可能的合法字串為:

t3t2t3t2
t3t2t3t2t1
t3t2t3t2t3
t3t2t3
t3t2t1

不符合此種規則的都是不合法的字串,可能為：
t1t2t3
t1t1t2t2
t1t2t3t2t3
t2tt3t1

根據此種方法,設計出各 30 個訓練範例與 10 個測試範例，之後先利用訓練範例進行訓練,直到網路收斂或經過 n 個訓練回圈為止即停止訓練開始進行測試。

(B) 參數值的測試

(1) 狀態神經元數目的影響:

研究結果[15]顯示當狀態神經元越多時,則測試錯誤率越小,此符合一般的推論,因為當狀態神經元愈多,則網路所能記載的資料愈多,所得的推論也愈正確。但是不難發現狀態神經元愈多,其所花的時間也相對得大很多所以找到一個適中的平衡點,即是同時兼顧到正確率與訓練的時間是重要的課題。但是就倒傳遞網路的隱藏層而言，其隱藏層處理單元的數目是並無標準方法可以決定，經常需以試驗方式決定其最佳數目，所以在此的狀態神經元應該也是類似的，即狀態神經元數目並無標準方法可以決定，應該亦是需以試驗方式決定其最佳數目。

而且由研究結果顯示可知當網路的狀態神經元為 3 時,其錯誤率已不算大,約在 0.3 至 0 之間,而且所用的錯誤容忍程度是 0.2,即是如果目標輸出值是 1 而推論輸出值是 0.79,則在此不予列入正確,而將其算入錯誤的行列,必須其推論輸出值是大於 0.8 才算正確;相同的情形,如果目標輸出值是 0 而推論輸出值是 0.25。則在此不予列入正確,而將其算入錯誤的行列,必須其推論輸出值是小於 0.2 才算正確。因此如果把錯誤容忍度定為 0.5,則所得的錯誤率也就小得多了。因此能看出此系統在派群語言的辨識成效。

同樣的研究結果顯示其狀態神經元為 3 時的錯誤率幾乎已經跟狀態神經元為 4 時差不多，而且時間上要快許多，表示此種派群 3 個狀態神經元已經能夠紀錄其行為。

(2) 學習速率值的影響:

研究結果顯示當學習速率值到 15 都還能被網路所接受，所測試的正確率最高。經由測試得知此網路的學習速率與一般的倒傳遞網路有一點不一樣,雖然兩者都是用最陡坡降法,可是倒傳遞網路通常介於 0 與 1 之間。而從表中可以發現當學習速率值為負的時後，錯誤率有趨近百分之百的，此顯示學習速率值為負的不可行，因為如此會把其推向能量函數大的地方，導致目標輸出值與推論輸出值相差愈多。研究結果顯示所測試的正確率也還略比學習速率值 4 還高，而二者的正確率都明顯的比學習速率值為 1 時高許多。

(3) 學習速率折損因子的影響

學習速率折損因子即是讓學習速率能隨著訓練的時間增加而慢慢減少學習速率的值，此是為了在能量函數最小化的過程能夠更順利，避免因為一直維持太大而導至無法收斂。其作法是給一個介於 0 與 1 之間的值，當做學習速率折損因子的值，而後在每一次的學習循環之後，再以學習速率乘以學習速率折損因子的值當做新的學習速率值。研究結果顯示學習速率折損因子影響表，由表不難發現學習速率折損因子越大的其測試的錯誤率越小。

(4) 考慮學習速率折損因子的影響

由研究結果可以發現有考慮學習速率折損因子時，且用適當的折損因子，比完全不考慮時的效果還要好一點，即折損因子是 0.9 比折損因子是 1 時要好，的效果要好。因為學習速率折損因子為 1，就是學習速率在每個學習循環之後還是原來本身並沒有改變。

(C) 對於考慮誤差標準演算法的描述與測試:

考慮誤差標準的演算法，即是一般所稱的補充學習 (supplementary learning)，因為在二階重入網路的演算法中，主要的執行時間是用在遞迴的修正加權值，因此對於已經學習成功的訓練範例，即不執行加權值的修正應可節省執行時間，此即其原理。

因此網路的學習過程是使上述能量函數最小化的過程 ,通常以最陡坡降法來使能量函數最小 ,而與原本的演算法所不同的是，只有當此範例所導出的推論輸出值要比誤差標準值(ε)的某個倍率平方($E_0 \geq 0.5\varepsilon^2$)還大，再小幅調整加權值的大小 ：

$$\Delta W_{lmn} = -\alpha\left(\frac{\partial E_0}{\partial W_{lmn}}\right)$$

$$\text{if} \quad E_0 \geq 0.5\varepsilon^2$$

其中 W_{lmn} = 介於第 n-1 層的第 m 個狀態神經元和第 n 個輸入神經元 ,與第 n 層的第 l 個狀態神經元間的連結加權值。α = 學習速率(learning rate) ,控制每次以最陡坡降法最小誤差函數的步幅。

研究結果顯示此種網路結構在有無誤差標準值上的比較。也可以看出有誤差標準值考慮時其執行速度比不考慮時要快上許多,但是測試錯誤率也比沒有考慮誤差標準值還要差。所以為了要有比較的依據，結果顯示考慮誤差標準值的演算法，其測試的錯誤率似乎比較好。

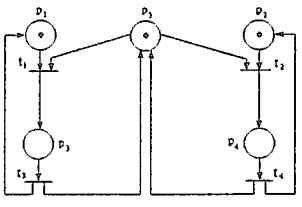

圖 3、派群圖例 1

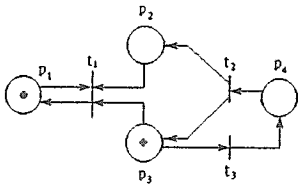

圖 4、派群圖例 2

伍 三階重入類神經網路

前所討論的派群語言，雖然在同一個時間點上可能有兩個以上的移轉可以觸發，但是實際上在模擬其輸入時，仍然得一次一個輸入，如果其一次能有兩個以上的輸入，則似乎更能夠表現系統的動作，基於此種概念，本研究欲探討一次能有兩個以上的輸入的可行性。

如圖 5 起始的狀態 U0={1,0,0,0,0}，當 t0 觸發之後，系統的狀態為 U1={0,1,1,0,0}，此時 t1,t2 都能被觸發，於是選擇同時觸發 t1,t2，之後系統的狀態為 U2={0,0,0,1,1}此時則只有 t3 能被觸發，當 t3 觸發之後系統的狀態為 U3={1,0,0,0,0}。

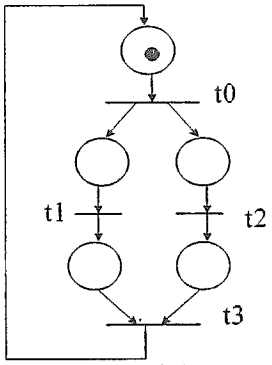

圖 5：派群圖例 3

(一) 二個平行度的派群語言

基於上述的描述可以定義另一種更描述派群動作的語言，如圖 5 若其終點的狀態集合為 Uf={{1,0,0,0,0}}，則經由起始的狀態 U0={1,0,0,0,0}，當 t0 觸發之後，系統的狀態為 U1={0,1,1,0,0}，此時 t1,t2 都能被觸發，於是選擇同時觸發 t1,t2，之後系統的狀態為 U2={0,0,0,1,1}此時則只有 t3 能被觸發，當 t3 觸發之後系統的狀態為 U3={1,0,0,0,0}，之後 t0 再度被觸發，系統的狀態為 U4={0,1,1,0,0}，此時 t1,t2 都能被觸發，於是選擇同時觸發 t1,t2，之後系統的狀態為 U5={0,0,0,1,1}此時則只有 t3 能被觸發，當 t3 觸發之後系統的狀態為 U6={1,0,0,0,0}，而後系統符合終止的條件終止，則描述的語言可以如 t0(t1t2)t3t0(t1t2)t3，此處以括弧來代表同時觸發的移轉動作。所以像 t0(t1t2)t3，t0(t1t2)t3t0(t1t2)t3t0(t1t2)t3 兩個在此都可以描述系統在有可以同時觸發的兩個移轉動作時，選擇了同時觸發，而如果不選擇同時觸發也可以如 t0(t1t2)t3t0t1t2t3 一般。

(二) 三階重入類神經網路架構

對於有二個平行度的派群語言，此處以三階重入類神經網路進行辨認，而如圖 6 所示即為一個三階重入類神經網路的架構，其中下面的的 S 是第 t 個時間的狀態神經元，I 是第 t 個時間的輸入神經元，P 是第 t 個時間的並行輸入神經元，上面的 S 是第 t+1 個時間的狀態神經元。

在三階重入類神經網路中，第 n 個時間（或第 n 層）的第 j 個輸出單元為第 n-1 個時間（或第 n-1 層）單元輸出值的非線性函數：

$$S_j^{t+1} = f(net_j^{t+1})$$

其中 net_j^{t+1}=集成函數 $= \sum_{j,k} W_{ijkl} S_j^t I_k^t P_l^t$

f = 轉換函數

其學習的過程旨在降低網路輸出單元，其目標輸出值與推論輸出值之間的差距，所以一般以下列能量函數(或稱誤差函數)表示學習的品質：

$$E_0 = \frac{1}{2}(T - S_0^f)^2$$

其中 T= 輸出層的目標輸出值

S_0^f = 輸出層的推論輸出值

因為此網路的學習過程變成使上述能量函數最小化的過程，通常以最陡坡降法來使能量函數最小化，即每當輸入一個訓練範例，網路即小幅調整加權值的大小，調整的幅度和誤差函數對該加權值的敏感度成正比，即誤差函數對加權值偏微分值大小成正比：

$$\Delta W_{lmno} = -\alpha \left(\frac{\partial E_0}{\partial W_{lmno}} \right)$$

其中 $_{lmno}$ = 介於第 n-1 層的第 m 個狀態神經元、第 n 個輸入神經元和第 o 個並行輸入神經元，與第 n 層的第 l 個狀態神經元間的連結加權值。α = 學習速率(learning rate)，控制每次以最陡坡降法最小化誤差函數的步幅。

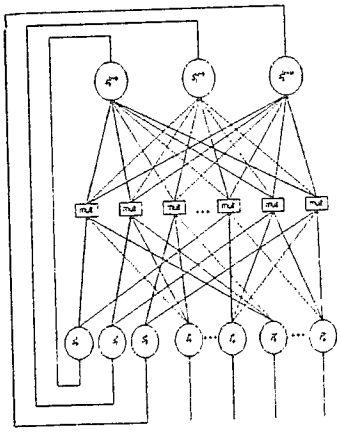

圖 6、 三階重入類神經網路的架構圖

整個學習過程的重要公式如下所示：

$$net_i^{t+1} = \sum_{j,k} W_{ijkl} S_j^t I_k^t P_l^t$$

$$S_i^{t+1} = f(net_i^{t+1})$$

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

$$E_0 = \frac{1}{2}(T - S_0^f)^2$$

$$\Delta W_{lmno} = -\alpha \left(\frac{\partial E_0}{\partial W_{lmno}} \right)$$

$$= -\alpha (T - S_0^t) \partial S_0^t / \partial W_{lmno}$$

$$\partial S_i^{t+1} / \partial W_{lmno} = f(net_i^{t+1})$$

$$*[\delta_{il} S_m^t I_n^t P_o^t + \sum_{j,k} W_{ijkl} I_k^t P_l^t \partial S_j^t / \partial W_{lmno}]$$

其中 net 右邊的 S 是第 t 個時間的狀態神經元，I 是第 t 個時間的輸入神經元，P 是第 t 個時間的並行輸入神經元。

此處之所以設了並行輸入神經元，是為了因應在同一個時間點上有兩個移轉動作同時發生時，然而並不是在每一個時間點時都會有兩個移轉動作同時發生，而當在一個時間點上只有一個移轉動作發生時，此並行神經元又該如何處理，如果把在一個時間點上只有一個移轉動作發生時，其並行輸入神經元全都用零表示的話，其想法是因為沒有並行輸入神經元的存在，所以就給他們全都是零的值，表示沒有任何一個移轉動作能產生作用，可是驗證之後發現當全部的並行神經元都是零，則連輸入神經元的作用都會被抵銷，而生不了任何的刺激作用，因為它是三階的加權值，所以並行神經元的值會影響輸入神經元的值，如果並行神經元的值是零，則整個集成函數的值也會是零。以上的分析可以知道，當在一個時間點上只有一個移轉動作發生時，把並行輸入神經元的值都設為零是不可行的。

若是把在一個時間點上只有一個移轉動作發生時，並行輸入神經元的值設成全都是 1 又如何，此時並行輸入神經元的作用因為每一個移轉動作都發生，所以就沒有任何一個移轉動作特別突出，能把其它的移轉動作蓋掉，而另一方面，輸入神經元的值也可以不受到任何的影響，就跟沒有並行輸入神經元是一樣的，這似乎能夠符合此訓練的要求，而根據測試也的確如此，把在一個時間點上只有一個移轉動作發生時，其並行輸入神經元用全部是真值代入能夠正確的訓練此種型態的派群語言。

陸 結論

文中嘗試擴充二階重入類神經網路於派群上，並討論多項網路參數值對於測試正確率所造成的影響，由各項測試證實，經由某一個所給予的派群的字串集合，經過訓練之後，可以辨認我們的測試範例，表示此網路可用以辨認派群的語言。

另外定義了平行觸發的移轉動作語言，以及用三階重入類神經網路來辨認有兩個平行度的語言，此三階重入類神經網路的架構及其實作方法，經由測試也確實可行。

雖然大致上此種二階和三階重入類神經網路在派群的語言辨認上效果都可達百分之九十以上，甚至百分之百，但是由於此種時空樣本識別網路的一般特性，其最主要的限制是網路執行成本太大，使得我們無法辨認太長的字串 ，仍然是急待克服的 。

柒 參考文獻

[1] 毛維凌, 陳振鈞, "類神經網路與結構性時間數列之比較與研究", 政大經濟研究所, 1993 年

[2] 邱顯明, 鄭博王, "以遺傳演算法及類神經網路應用於旅行推銷員問題上", 淡大土木工程研究所, 1995 年

[3] 葉怡成, "類神經網路模式應用與實作", 儒林, 1995 年

[4] Axel Cleeremans, David Servan-Schreiber & James L. McClelland, "Finite State Automata and Simple Recurrent Networks", Neural Computation, 1, pp.372-381 (1989)

[5] C.W. Omlin, C.L. Giles & C.B. Miller, "Heuristics for the Extraction of Rules from Discrete-Time Recurrent Neural Netwroks", International Joint Conference Neural Network, pp.33-38 (1992)

[6] C.L. Giles, C.B. Miller, D. Chen, H.H. Chen, G.Z. Sun, Y.C. Lee, "Learning and Extracting Finite State Automata with Second-Order Recurrent Neural Networks", Neural Computation, 4, pp.393-405 (1992)

[7] Christian W. Omlin & C. Lee Giles, "Stable Encoding of Large Finite-State Automata in Recurrent Neural Networks with Sigmoid Discriminants", Neural Computation, 8, 675-696 (1996)

[8] Itsuki Noda & Makoto Nagao, "A learning Method for Recurrent Networks Based on Minimization of Finite Automata", International Joint Conference Neural Network, pp.27-31 (1992)

[9] Rodney M. Goodman & Padhraic Smyth,"Learning Finite State Machines With Self-Clustering Recurrent Networks", Neural Computation, 5, pp.976-990 (1993)

[10] Elman ,J.L.[1990],"Finding structure in time", Cognitive Sci. 14:179-211

[11] Elman ,J.L.[1990],"Distributed representations, simple recurrent networks, and grammar structure", Mach Learn, 7:195-225

[12] Pollack, J.B. [1987]."Cascaded backpropagation on dynamical connectionist networks" Proc. Ninth Conf. Cognitive Sci. Soc. ,391-404,Seattle,WA.

[13] Pollack, J.B. [1990]."Recursive distributed representation ",Artif. Intell. 46:77-105

[14] James L. Peterson [July 1981], " Petri net theory and the modeling of systems", Prentice Hall ,Inc.

[15] 張文龍, "使用多階重入類神經網路辨識派群(Petri net)語言", 碩士論文, 淡江大學資訊管理研究所, 1997 年