

시공간패턴인식 신경회로망의 설계

Neural Network Design for Spatio-temporal Pattern Recognition

林 政 秀* · 李 種 浩**

(Chung-Soo Lim · Chong-Ho Lee)

Abstract - This paper introduces complex-valued competitive learning neural network for spatio-temporal pattern recognition. There have been quite a few neural networks for spatio-temporal pattern recognition. Among them, recurrent neural network, TDNN, and avalanche model are acknowledged as standard neural network paradigms for spatio-temporal pattern recognition. Recurrent neural network has complicated learning rules and does not guarantee convergence to global minima. TDNN requires too many neurons, and can not be regarded to deal with spatio-temporal pattern basically. Grossberg's avalanche model is not able to distinguish long patterns, and has to be indicated which layer is to be used in learning. In order to remedy drawbacks of the above networks, unsupervised competitive learning using complex number is proposed. Suggested neural network also features simultaneous recognition, time-shift invariant recognition, stable categorizing, and learning rate modulation. The network is evaluated by computer simulation with randomly generated patterns.

Key Words : 신경회로망, 자율경쟁학습, 시공간 패턴인식, 복소수

1. 서 론

자연계에 존재하는 패턴은 대부분 시공간 패턴(spatio-temporal pattern)이다. 우리의 눈과 귀를 통해서 들어오는 시각, 청각 신호도 물론 시공간 패턴이다. 시공간 패턴이란 시간적으로 변화하는 다채널 신호를 말하는데 인공지능 시스템을 만드는 데에는 외부세계로부터의 이런 신호를 인간이 하는 것처럼 지능적으로 인식하는 능력이 필요하다. 이를 위해 패턴인식에 대한 많은 연구가 이루어져 왔는데 이 분야의 여러 연구 중에서 신경회로망을 이용하여 패턴인식을 시도하는 연구가 많은 성과를 거두었다. 패턴인식에서 신경회로망이 가지는 장점은 잡음에 잘 대처할 수 있고 스스로 학습을 통해서 새로운 패턴에 대해서도 인식력을 가진다는 점이다. 문자인식과 같은 시변패턴 인식을 비롯하여 음성 인식 등 시변패턴의 인식에 이르기까지 신경회로망은 다양하게 이용되고 근래에는 RBF(radial basis function)나 wavelet, 또는 HMM(hidden Markov model) 등 여러 다른 이론들을 신경회로망에 접목시켜 신경회로망의 이론적인 측면을 보강하고 성능향상을 꾀하여 왔다. 그러나 기존의 신경회로망에 비해서 많은 계산량을 요구하는 단점이 있다.

기존의 시공간 패턴인식 신경회로망으로는 지도학습

(supervised learning)을 사용하는 TDNN, 순환신경회로망(recurrent neural network), 그리고 자율학습을 사용하는 Grossberg의 avalanche model을 기초로 한 네트워크 등이 있다. [1] 이들의 문제점인 복잡한 학습규칙, 불안정적인 인식 등을 극복하고 시스템 스스로 학습이 가능하도록 하기 위하여 본 논문에서는 비교적 간단한 경쟁학습규칙을 이용하여 시공간 패턴인식을 할 수 있는 신경회로망을 제안한다. 모두 5개의 층으로 구성되는 계층적 구조를 가지고 있고 시스템은 크게 1층에서 3층까지의 하위 시스템과 4층과 5층으로 구성되는 상위 시스템으로 나눌 수 있다. 각기 시스템은 각기 수행하는 임무가 다른데 이런 형태의 시스템을 혼성네트워크(Hybrid network)라고 한다. [2] 상위 시스템은 하위 시스템에서 인식하는 패턴의 연속으로 이루어지는 패턴을 인식한다. 두 시스템 모두 자율경쟁학습(unsupervised competitive learning)을 사용하는데 이는 외부에서 전달되는 신호를 주요한 신호와 잡음으로 분별하고 스스로 분류항목을 만들어 인식하는 것을 가능하게 한다. 또한 두 시스템 모두 복소수를 처리할 수 있도록 설계되었고 기존의 뉴런과는 다른 성질과 구조를 가지는 새로운 뉴런이 제안되었다.

2. 시스템 디자인

2.1 제안하는 신경회로망의 개요

제안하는 신경회로망은 시공간패턴을 입력으로 받아서 그에 해당하는 패턴벡터를 출력한다. 입력은 행렬의 형태로 주어지는데 행렬의 첫 번째 행부터 마지막 행까지 순차적으로 입력된다. 자율학습을 이용하기 때문에 자주 입력되는

* 正 會 員 : 仁 荷 大 電 氣 工 學 科 · 碩 士

** 正 會 員 : 仁 荷 大 電 氣 工 學 科 教 授 · 工 博

接 受 日 字 : 1999年 3月 25日

最 終 完 了 : 1999年 9月 30日

패턴을 스스로 학습하여 인식하게되는 반면 지도학습의 경우보다 학습할 수 있는 패턴의 개수가 상대적으로 적다. 그러나 새로운 패턴에 대해서도 자율적으로 대처할 수 있다는 장점이 있다.

시공간 패턴을 인식하기 위해서는 어떻게 해서든지 과거의 입력에 대한 정보를 가지고 있어야 하는데 이를 위해서는 TDNN의 경우처럼 시간지연소자를 사용하거나 아니면 순환신경회로망처럼 되먹임 루프가 필요하다. 제안하는 신경망은 첫 번째와 두 번째 층에서 복소수의 개념을 도입하여 입력행렬을 하나의 벡터로 전환시키는 역할을 한다. TDNN의 경우 정확하게 과거의 정보를 저장할 수 있지만 효율적이지가 못한 반면 recurrent network의 경우는 효율적이지만 과거의 정보를 정확히 나타내기는 힘들다. 이러한 이들의 장점만을 살려서 효율적이면서 과거의 정보의 손실 없는 복소수를 이용한 방법을 제안하였다.

두 번째 층과 세 번째 층은 전환된 입력벡터를 복소수를 이용한 자율학습을 통하여 세 번째 층의 하나의 뉴런으로 대응시키는 일을 한다. 모든 경쟁학습의 경우 입력벡터를 정규화 해주는 것은 필수적인데 이를 대신하고 뉴런들간의 경쟁을 돕기 위하여 복소수를 이용한 경쟁학습법이 이용된다.

세 번째 층과 네 번째 층은 순차적으로 활성화되는 세 번째 층상의 뉴런의 순서를 기억하기 위해 복소수를 사용한다. 복소수를 이용한 방법은 기존의 활성화 함수를 이용한 방법[3]보다 계산상 간단하며 더욱 정밀하게 순서를 나타낼 수 있다.

네 번째 층과 다섯 번째 층은 복소수를 사용한 경쟁학습을 통하여 세 번째 층의 뉴런의 활성화 순서로 이루어진 패턴을 다섯 번째 층상의 한 개의 뉴런으로 대응시키는 역할을 한다. 다섯 번째 층상의 뉴런은 입력이 복소수 일뿐 아니라 억제성 입력도 복소수의 형태를 취함으로써 상위패턴(super-pattern)과 하위패턴(sub-pattern)의 학습을 용이하게 하였다.

다시한번 정리해 보면 시스템으로 들어가는 입력행렬이 첫 번째와 두 번째 층을 통과하면서 벡터로 압축이 되고 그것을 세 번째 층의 한 뉴런이 학습을 한다. 세 번째 층상의 뉴런이 출력을 내면 네 번째 층으로 전달되어 자기되먹임 루프를 통하여 저장되고, 세 번째 층까지의 시스템이 reset 된 후 다시 입력행렬이 들어와서 동일한 과정을 거쳐 세 번째 층의 다른 뉴런이 출력을 내게된다. 이 출력이 다시 네 번째 층으로 보내져 저장이 된다. 네 번째 층에서 저장되고 있는 세 번째 층상의 뉴런이 활성화 된 순서의 정보가 다시 다섯 번째 층으로 학습이 되어서 최종적으로 출력을 내보내게 된다.

2.2 Layer1 to layer2 subsystem

첫 번째 층과 두 번째 층이 하는 일은 한 개의 $n \times m$ 행렬을 $n \times 1$ 벡터로 압축하는 일이다. 즉 시공간 패턴을 공간패턴으로 변환하는데 있어서 행렬 내의 하나의 행은 벡터의 한 개의 요소로 대응이 된다. 이 경우 얼마나 정보의 손실이 없이 변환하는 가가 중요한데 기존의 순환신경회로망에서 사용된 되먹임을 이용한 방법은 입력행렬의 행의 수가

많아질수록 변별력이 떨어지고 TDNN에서 사용된 시간지연소자를 이용하는 방법은 입력행렬을 출력벡터로 만드는데 많은 과정을 거쳐야 하는데 반하여 제안하는 방법은 복소수를 이용하여 정보의 손실이 거의 없이 간단한 과정을 거쳐 변환을 실행한다. 복소수와 필터의 개념이 도입되고 중간에 활성화함수가 추가되었지만 충분히 보상되리라고 생각된다.

2.2.1 첫 번째 층의 뉴런

첫 번째 층의 뉴런은 실수를 입력받아서 복소수를 출력하고 복소수의 크기 성분은 실수입력과 동일하고 각 성분은 시간의 지표가 된다. 즉 입력행렬은 행 단위로 순차적으로 들어오는데 각 행 벡터가 입력될 때 첫 번째 층의 뉴런은 각 행벡터 마다 다른 각 성분을 출력하는 것이다. 출력되는 각 성분은 행 벡터들이 순차적으로 입력됨에 따라 일정하게 증가한다. 뉴런의 구조는 그림1과 같다. $F_{i1}^1(\cdot)$ 는 입력이 들어오면 그 크기에 상관없이 시간에 따라 일정하게 증가하는 각 성분을 출력하는데 각 성분의 증가량은 $360/(\text{첫 번째 층의 뉴런의 개수})$ 로써 정해지며 $F_{i2}^1(\cdot)$ 는 선형함수로서 입력과 동일한 값을 출력한다.

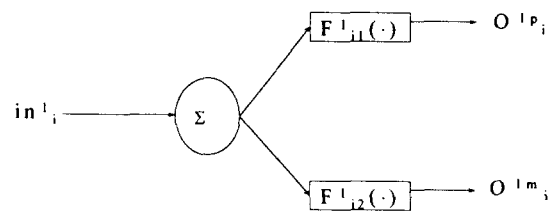


그림 1 첫 번째 층의 뉴런 i의 구조
Fig. 1 Neuron i in layer one

2.2.2 두 번째 층의 뉴런

두 번째 층의 뉴런은 복소수를 입력으로 받아서 다시 복소수를 출력하는데 그림2와 같이 첫 번째 층으로부터의 입력의 각 성분은 가중치를 거치지 않고 입력되고 크기성분은

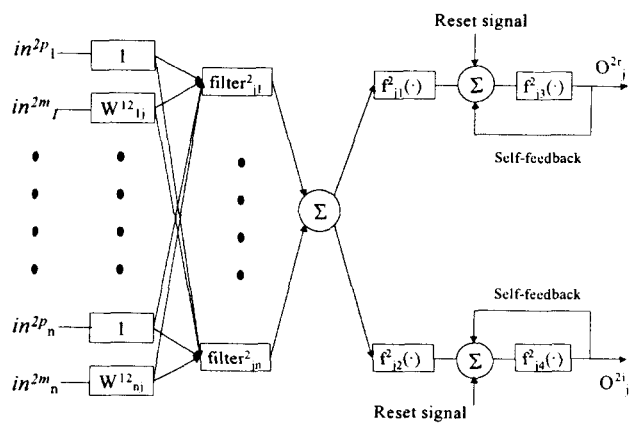


그림 2 두 번째 층의 뉴런 j의 구조
Fig. 2 Neuron j in the second layer

가중치와 곱해져서 입력된다. 첫 번째 층과 두 번째 층 사이의 가중치는 실수이며 완전히 연결되어있고(fully connected), 고정되어 있는데 그 이유는 같은 입력에 대하여 같은 값을 두 번째 층으로 전달하기 위함이다. 가중치는 첫 번째 층의 출력 벡터에 따라 두 번째 층으로의 입력이 고유한 값을 가지도록 정해지고 입력된 복소수는 필터를 거치는데 필터는 정해진 각성분을 가지는 복소수의 크기 성분만을 통과시킨다. 각 뉴런은 통과시키는 각성분이 각기 다른데 그 각성분은 가장 왼쪽에 있는 뉴런으로부터 오른쪽으로, 0도에서 시작하여, 첫 번째 층의 뉴런이 출력하는 각성분의 증가량과 동일하게 증가한다. 어떻게 보면 복잡한 것처럼 보이지만 실제 구현에 있어서는 시스템으로 들어오는 입력 행렬의 각 행 벡터와 가중치와의 곱을 두 번째 층의 왼쪽 뉴런에서부터 시작하여 오른쪽으로 순서대로 입력되게 만들면 된다. 그 후 그 곱을 복소수에서의 각 성분으로 생각하여 코사인 함수인 $f_1^2(\cdot)$ 와 사인 함수인 $f_2^2(\cdot)$ 를 통하여 복소수를 출력하는데 이렇게 하면 출력되는 복소수는 단위원상의 복소수가 된다.

2.3 Layer2 to layer3 subsystem

만약 두 번째 층에서 복소수의 개념을 도입하지 않아서 필터를 통과한 값을 각으로 생각하지 않고 그냥 실수로 출력하면 정규화가 되지 않은 관계로 경쟁학습이 불가능하다. 이때 중간에 입력을 정규화 시켜줄 방법이 없으므로 이것을 해결하기 위하여 복소수를 이용한다. 세 번째 층으로 들어오는 입력은 단위원상의 복소수이고 또한 가중치도 단위원상의 복소수를 이용한다. 단위원상의 두 복소수간의 곱의 실수부의 최대값은 두 수가 절레의 관계일 때 1인 성질을 이용하여 정규화를 대체하는 효과를 거둘 수 있다. 가중치를 입력의 절레 복소수로 학습을 시키면 입력과 가중치의 곱의 최대값은 입력행렬의 행의 개수가 된다. 다시 말해서 어떤 입력행렬에 대해서 세 번째 층의 뉴런이 가질 수 있는 최대 활성정도는 제한되어져 있고 그 크기가 클수록 그 뉴런이 입력행렬을 잘 반영하고 있다는 뜻이다. 그밖에도 학습이 끝난 가중치에 대해서는 그 값을 고정시켜서 안정성을 보장하는 것, 습관성에 근거한 학습을 변조, 그리고 학습이 불가능한 뉴런을 방지하기 위하여 활성정도가 0보다 작더라도 최소한의 학습을 실행시킨 것, 그리고 시공간패턴의 길이의 한계를 극복하고자 리셋 시스템을 사용한 것, 버퍼의 개념을 도입한 것 등이 특징이라고 할 수 있다. 리셋 시스템은 두 번째 층의 뉴런의 개수보다 많은 행의 개수를 가지는 입력행렬을 인식하기 위하여 쓰이는데 두 번째 층의 모든 뉴런이 활성화되면 첫 번째 층에서 세 번째 층까지 리셋 시킨다.

2.3.1 세 번째 층의 뉴런

그림3은 세 번째 층의 뉴런의 구조를 나타내는데 in^{3r}_i 는 복소수 입력의 실수부를 나타내고 in^{3i}_i 는 허수부, W^{23r}_{ij} 는 복소수 가중치의 실수부, W^{23i}_{ij} 는 허수부를 나타낸다. $exin$ 은 layer3의 뉴런들로 입력되는 흥분성 입력으로 이루어진 벡터이고 IW_j^3 는 뉴런 j로 들어오는 억제성 가중치 벡터(0과 1 사이의 실수)이며 O_j^3 는 뉴런 j의 출력(실수)이다. $f_{j1}^3(\cdot)$ 은

임계 활성화 함수로서 임계치보다 큰 입력이 들어오면 1을 출력하는데 그 임계치는 학습 전에는 두 번째 층의 뉴런의 개수로 정해지고 학습 후에는 인식하는 패턴이 입력되었을 때 가능한 활성정도의 90%로 정해진다. 즉 학습이 되기 전에는 아무 값을 출력하지 못하고 학습후에 학습되어진 입력이 들어온 경우에만 출력을 낸다. $f_{j2}^3(\cdot)$ 역시 임계함수(threshold function)로서 임계값보다 큰 값에 대해서는 선형 함수로 동작하는데 이 함수는 $exin$ 이 0보다 적은 경우, 다른 뉴런을 음으로 억제, 즉 양으로 자극하는 것을 막기 위함이다

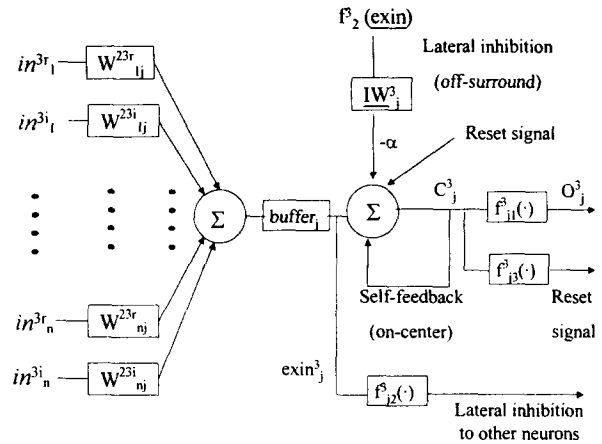


그림 3 세 번째 층의 뉴런 j의 구조
Fig. 3 Neuron j in the third layer

다. 다른 뉴런을 억제하는 신호가 출력에 연결되지 않고 $exin$ 에 연결된 이유는 첫째로 억제성 가중치가 $exin$ 에 의하여 갱신되기 때문이고 두 번째 이유는 만약 출력을 가지고 억제한다면 뉴런의 입력과 출력은 지연이 있으므로 현재 억제 신호는 현재 뉴런의 활성정도를 반영하는 것이 아니라 과거의 활성정도를 반영하는 것이기 때문에 억제신호가 현재의 활성정도를 반영하게 하기 위함이다. C_j^3 는 활성함수를 통과하기 전 뉴런의 상태이고, α 는 양의 실수로 흥분성 입력과 억제성 입력의 비율을 조절한다. $f_{j3}^3(\cdot)$ 은 네 번째 층의 뉴런으로 리셋 신호를 보내기 위한 임계 활성화함수로 임계값은 학습이 완료되면 $f_{j1}^3(\cdot)$ 의 임계값의 90%인 임계값을 가지고 학습이 완료되기 전에는 초기값으로 두 번째 층의 뉴런의 개수를 갖는다. 그럼으로써 $f_{j1}^3(\cdot)$ 을 통해서 출력이 나가기 한 단위시간 전에 네 번째 층으로 리셋 신호가 출력되는데 이는 연속되는 서로 다른 패턴들이 네 번째 층의 어떤 한 뉴런을 공통으로 포함하고 있는 경우, 이를 해결한다. $buffer_j$ 는 학습이 종료되기 전에는 선형함수로서 동작하고 학습이 종료되면 학습된 패턴의 크기 보다 조금 작은 값을 임계값으로 하는 비선형함수로 작동한다. 그럼으로써 현재 입력된 패턴이 이미 학습되어진 패턴과 충분히 비슷하지 않은 경우 다른 뉴런을 억제하지 않아 학습되어진 패턴과 유사한 패턴의 학습을 방해하지 않도록 해 준다.

세 번째 층의 뉴런의 출력 방정식은 다음과 같다.

$$O_j^3 = f_{j1}^3(C_j^3) \quad (1)$$

$$C_j^3(k) = (1 - \frac{C_j^3(k-1)}{A})(C_j^3(k-1) + \text{exin}_j^3(k) - a \cdot \text{inhin}_j^3(k)) \quad (2)$$

$$\text{exin}_j^3(k) = (\sum_{i=1}^n W_{ij}^{23r} \cdot \text{in}_i^{3r}(k) + W_{ij}^{23i} \cdot \text{in}_i^{3i}(k)) \quad (3)$$

$$\text{inhin}_j^3(k) = \sum_{i=1}^n IW_{ij}^3 \cdot f_p^3(\text{exin}_j^3(k)) \quad (4)$$

exin_j^3 는 세번째 층의 뉴런 j로 들어오는 흥분성 입력으로 실수값을 가진다. inhin_j^3 는 세 번째 층의 뉴런 j로 들어오는 억제성 입력이다. IW_{ij}^3 은 실수값으로 세 번째 층의 뉴런 i에서 뉴런 j로의 억제성 가중치로 대각선 성분은 모두 0이다.

2.3.2 복소수를 이용한 경쟁학습법

복소수를 이용한 경쟁학습은 근본적으로 보통 경쟁학습과 다른 점은 없다. 가장 큰 활성화 정도를 가지는 뉴런을 승자로 삼아 contrast enhancement를 통해서 winner-take-all을 이루는 것이다. 다만, 복소수를 다루기 때문에 활성화 정도는 복소수의 실수부의 크기로 정한다. 단위원상의 복소수끼리의 곱에서 큰 실수값을 가지려면 두 수의 각 성분이 서로 켈레(conjugate)의 관계를 가져야 한다. 즉 입력의 각이 θ 라면 가중치의 각은 $-\theta$ 가 되어야 하는데 이런 방향으로 학습시킬 때 주의할 점은 코사인함수가 주기함수이기 때문에 각을 증가시켜야 할지 아니면 감소시켜야 할지를 잘 판단해야 한다는 점이다. 학습 조건을 정리해 보면 다음과 같다. WP는 가중치의 각 성분이라고 하고 inp를 입력의 각 성분이라고 하면

- 1) $WP_{ij}^{23} + \text{inp}_i^3 < \pi$ 인 경우에는 $WP_{ij}^{23} + \text{inp}_i^3$ 가 0이 되도록 감소해야 한다.
 - 2) $WP_{ij}^{23} + \text{inp}_i^3 > \pi$ 인 경우에는 $WP_{ij}^{23} + \text{inp}_i^3$ 가 2π 가 되도록 증가해야 한다.
- 가중치를 갱신해 주는 식은 다음과 같다.

$$\Delta WP_{ij}^{23} = \lambda \frac{23}{ij} \cdot C_j^3(\text{inp}_i^3 - WP_{ij}^{23}) \quad (5)$$

C_j^3 가 0보다 커야 학습이 일어나고, 작거나 같으면 학습은 최소한의 양으로만 일어나서 학습이 불가능한 상태를 방지한다. 그리고 복소수의 경쟁학습에서는 각을 다루기 때문에 각을 0부터 2π 사이로 정렬하는 것이 필요하다. $\lambda \frac{23}{ij}$ 는 W_{ij}^{23} 의 학습율이다.

일단 학습이 끝난 가중치는 자기가 인식하는 패턴 외에 다른 패턴을 학습하지 못하게 하기 위하여 학습이 끝난 후 학습율을 0으로 고정시켜서 인식하는 패턴에 대한 안정성을 유지시킨다.

억제성 가중치도 contrast enhancement를 향상시키고 학습속도도 향상시키기 위하여 학습시킨다.[3] 다음은 억제성 가중치를 학습시키는 규칙이다.

$$\text{exin}_i^3 > \text{exin}_j^3 \text{ 이고 } \text{exin}_j^3 > 0 \text{ 이면,} \\ \Delta IW_{ij}^3 = \eta \frac{3}{ij} \cdot \text{exin}_j^3(1 - IW_{ij}^3) \quad (6)$$

$$\text{exin}_i^3 < \text{exin}_j^3 \text{ 이고 } IW_{ij}^3 > 0 \text{ 이면,} \\ \Delta IW_{ij}^3 = \eta \frac{3}{ij} \cdot \text{exin}_j^3(0 - IW_{ij}^3) \quad (7)$$

위의 규칙에 의해 학습이 된 억제성 가중치는 흥분성 입력에 곱해져서 자기 자신 외의 모든 다른 뉴런으로 출력된다.

2.3.3 학습율 변조

세 번째 층으로 입력되는 패턴을 시간별로 살펴보면 필터에 의하여 두 번째층의 맨 왼쪽뉴런에서부터 오른쪽으로 차례대로 출력을 내고 또한 자기되먹임에 의하여 그 출력은 계속 유지되므로 세 번째 층으로 들어오는 패턴은 각 시간마다 하나씩 덧 붙여져서 만들어지게 된다. 이런 이유로 인하여 처음에 들어오는 패턴의 원소는 많이 학습이 되어지고 나중에 들어오는 원소는 학습이 상대적으로 조금만 이루어지게 되는데 이것을 극복하기 위해서 인간의 감각기관에서 흔히 볼 수 있는 습관성에 근거한 방법을 제안하였다. 습관성(habituation)이란 동일한 자극이 계속되면 그 자극에 대하여 둔감해지는 것을 의미한다. 예를 들어 우리가 어떤 냄새에 오랫동안 노출이 되어 있으면 나중에 그 냄새에 둔감해지게 되는데 이와 마찬가지로 동일한 입력이 계속하여 들어온다면 학습속도는 감소하게 되고 입력이 없으면 최고의 학습속도로 회복되게 된다. 다음 식은 습관 방정식을 나타낸다.[4]

$$\lambda(t+1) = \lambda(t) + \tau(\beta(1 - \lambda(t)) - \lambda(t) \cdot \text{in}(t)) \quad (8)$$

τ 는 습관성 비율(ratio)이고, β 는 습관과 회복사이의 비율이다. 여기서 $\eta(t)$ 는 학습율이다. 이 식에서 $\text{in}(t)$ 가 계속 들어오면 학습율은 감소하는데 그 하한가는 다음과 같이 구해진다.

$$\text{in}(t)=1 \text{ 이고 } \Delta \lambda = 0 \text{ 이 되는 } \lambda \text{ 값은,}$$

$$\Delta \lambda = \lambda(t+1) - \lambda(t) = \tau(\beta(1 - \lambda(t)) - \lambda(t)) \quad (9)$$

$$\lambda_{\min} = \frac{\beta}{\beta+1} \quad (10)$$

2.4 Layer3 to layer4 subsystem

이 시스템은 복소수를 이용하여 세 번째 층상의 뉴런이 활성화되는 순서를 저장한다. 세 번째 층의 뉴런들이 활성화되는 순서를 저장하는 까닭은 뉴런의 활성화 순서로서 형성되는 패턴을 만들기 위함이다. 즉, 궁극적으로 세 번째 층에서 인식하는 패턴의 연속으로 이루어진 패턴을 다섯 번째 층의 하나의 뉴런에 학습시키려는 것이다. 활성화되는 순서를 나타내는 기존의 방법으로는 활성화함수를 이용하여 시간이 흐름에 따라 활성정도를 일정량만큼 증가시키거나 감소시키는 방법이 있었는데[3], 증가시킨 경우 활성화 순

서로서 형성되는 패턴이 길면 처음 들어온 입력은 현재의 입력에 비해 상대적으로 아주 큰 값을 가지게 되므로 이런 패턴을 입력으로 학습을 시키면 시간적으로 나중에 들어온 입력을 잘 인식할 수가 없다. 이러한 단점을 극복하기 위하여 복소수를 이용하여 시간의 순서를 나타내는 방법을 제안하였다.

2.4.1 네 번째 층의 뉴런

그림4와 같이 세 번째 층의 실수 출력 1이 복소수 가중치를 통하여서 네 번째 층의 뉴런으로 입력되는데 이 복소수는 선형 활성화 함수 $f_1^4(\cdot)$ 와 $f_2^4(\cdot)$ 를 통하여서 출력되고 또한 자기피드백 된다. W^{34r} 는 복소수 가중치의 실수부로 $\cos \theta$, 그리고 W^{34i} 는 허수부로 $\sin \theta$ 의 형태를 가진다. 즉 단위원 상의 복소수가 네 번째 층으로 입력되고 θ 는 update 신호에 따라 갱신된다. 세 번째 층과 네 번째 층 사이의 연결은 1대1 대응이고 모든 가중치는 동일한 값을 가진다.

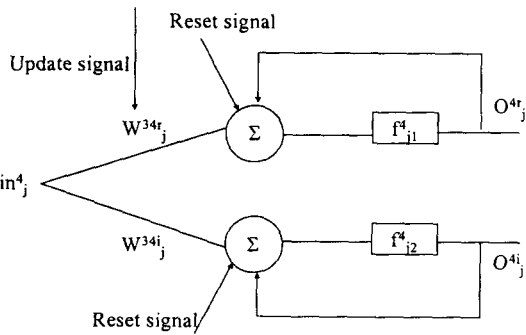


그림 4 네 번째 층의 뉴런 j의 구조
Fig. 4 Neuron j in the fourth layer

θ 를 갱신하는 식은

$$\Delta \theta = \theta^{34} \cdot update$$

$$\theta^{34} = \frac{360}{n^4} \tag{11}$$

Update는 세 번째 층 이하의 시스템이 리셋 될 때마다 발생하는 갱신신호이고 0과1의 값을 가진다. n4는 네 번째 층의 뉴런의 개수이다.

복소수를 이용하여 시간의 순서를 나타내는 방법은 다음과 같다. 예를 들어 세 번째 층의 i번째 뉴런을 N^4 로 나타낸다고 하고 활성화되는 순서가 $N^1-N^3-N^5-N^2-N^4$ 라면 네 번째 층으로 입력되는 패턴은 표1과 같다.

표에서 대괄호 안에 들어있는 패턴은 세 번째 층의 뉴런의 출력과 가중치가 곱해져서 일대일로 대응되는 바로 위의 네 번째 층의 뉴런으로 들어가는 입력이다. 즉 네 번째 층의 i번째 뉴런은 세 번째 층의 i번째 뉴런에서만 입력을 받는다.

표 1 네 번째 층으로의 입력

Table 1 An input to the fourth layer

Time	Pattern
1	$[1 \angle 0^\circ, 0, 0, 0, 0]$
2	$[1 \angle 0^\circ, 0, 1 \angle \theta^{340}, 0, 0]$
3	$[1 \angle 0^\circ, 0, 1 \angle \theta^{340}, 0, 1 \angle 2\theta^{340}]$
4	$[1 \angle 0^\circ, 1 \angle 3\theta^{340}, 1 \angle \theta^{340}, 0, 1 \angle 2\theta^{340}]$
5	$[1 \angle 0^\circ, 1 \angle 3\theta^{340}, 1 \angle \theta^{340}, 1 \angle 4\theta^{340}, 1 \angle 2\theta^{340}]$

2.5 Layer4 to layer5 subsystem

이 시스템은 복소수 입력을 받아서 실수를 출력하는데 네 번째 층에 저장되어 있는 패턴을 다섯 번째 층의 한 뉴런으로 학습시킨다. Layer2 to layer3 subsystem과 유사한 구조와 학습방법을 가지고 있고 필터의 개념도입, 패턴의 오버랩에 따른 억제성 가중치의 학습, 변수를 사용하여 패턴의 상대적 시간적 위치에 무관한 인식 등이 특징이라 할 수 있다.

2.5.1 다섯 번째 층의 뉴런

다섯 번째 층상의 뉴런은 하위패턴(sub-pattern)의 학습, 또한 기존의 패턴을 포함하는 상위패턴(super-pattern)의 학습을 위하여 뉴런에 버퍼와 필터의 개념을 도입하였다. 뉴런의 구조는 다음 그림5와 같다. 네 번째 층의 뉴런은 실수와 허수를 출력하는데 이것이 각각 실수 가중치와 허수 가중치와 곱하여져 실수입력만이 다섯 번째 층의 뉴런으로 입력된다. $f_1^5(\cdot)$ 은 선형 활성화함수이고 $f_2^5(\cdot)$ 는 같은 층상의 다른 뉴런을 억제하는 신호를 출력하는데 실수인 $exin$ 을 입력으로 하여 복소수를 출력한다. 그 복소수는 두 가지 정보를 가지고있는데 복소수의 크기는 억제의 크기이고 작은 출력을 내보내는 뉴런이 인식하는 패턴의 크기를 나타낸다. 관계식은 다음과 같다.

$$f_2^5(x) = x \times (1 \angle sz_j \cdot \theta^{34}) \quad \text{if } in > \text{threshold} \tag{13}$$

$$= 0 \quad \text{otherwise}$$

$$\text{threshold} = \text{인식하는 패턴의 크기} - 1 \tag{14}$$

필터는 들어오는 억제성 입력을 선택적으로 통과시키는 역할을 한다. 억제성 입력은 복소수로 들어오게 되는데 그림 5에서 알 수 있듯이 버퍼를 통과한 흥분성 입력에 복소수가 곱해지게 된다. 필터는 이런 복소수를 받아서 특정 각 성분만을 통과시키고 나머지 각성분은 통과시키지 않는다.

$$\text{Filter}(in^5_j) = ||in^5_j|| \quad \text{if } \angle in^5_j \geq sz_j \cdot \theta^{34} \tag{15}$$

$$= 0 \quad \text{otherwise}$$

위의 식은 학습이 완전히 종료되기 전 필터의 식이고 학습이 끝난 후의 식은 다음과 같다.

$$Filter(in_i^5) = |in_i^5| \quad \text{if } \angle in_i^5 > sz_i \cdot \theta^{34} \quad (16)$$

$$= 0 \quad \text{otherwise}$$

필터는 학습 시에는 자기자신보다 크기가 크거나 같은 뉴런으로부터의 억제 신호만을 통과시키고, 학습이 끝난 후에는 자기 자신보다 크기가 큰 뉴런으로부터의 억제 신호만 통과시킨다. 그럼으로써 어떤 학습된 패턴을 포함한 더 큰 패턴을 학습할 때 학습을 순조롭게 해준다.

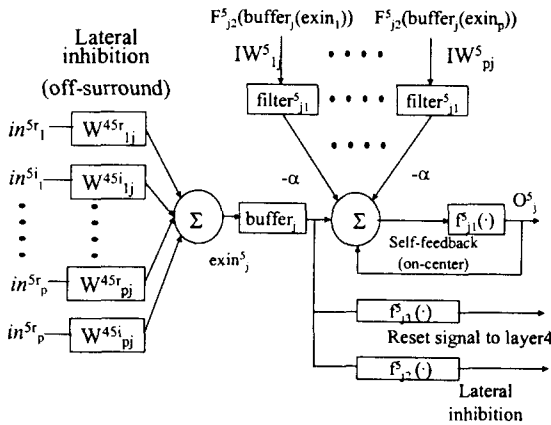


그림 5 다섯 번째 층의 뉴런 j의 구조
Fig. 5 Neuron j in the fifth layer

2.5.2 가중치의 학습

Layer4 to layer5 subsystem의 흥분성 가중치의 학습 방법은 layer3 to layer4 subsystem의 학습방법과 동일하고 억제성 가중치의 학습 방법은 패턴간의 오버랩을 이용한다. 이것은 한 뉴런이 다른 뉴런에 억제 신호를 보낼 때, 서로 인식하는 패턴사이에 오버랩이 존재할 경우 그 정도에 의해서 억제의 양을 조절하는 방법이다. 이렇게 함으로써 서로 오버랩이 없는 패턴을 인식하는 뉴런들은 서로 억제하지 않아서 서로 오버랩이 없는 패턴들은 동시에 인식이 가능하다.[3] 예를 들어 패턴 AB를 인식하는 뉴런과 패턴 CD를 인식하는 뉴런이 있다고 할 때 패턴 ABCD가 입력된다면 두 뉴런은 서로 억제하지 않으므로 모두 활성화가 되어 동시인식이 가능하다. 이로써 자율학습이 가지는 용량의 한계를 어느 정도 극복할 수 있다. 앞의 예에서 패턴 ABCD를 새로이 학습할 뉴런이 남아 있지 않다고 하더라도 AB와 CD를 각각 인식하는 뉴런들이 함께 활성화됨으로써 새로운 ABCD라는 패턴을 인식하게 된다. 또한 AB와 CD가 각각 의미있는 패턴들이고 AB와 CD가 중간에 섞이 없이 ABCD로 입력이 되면 AB와 CD를 따로따로 인식함으로써 자동적으로 segmentation이 일어난다. 오버랩을 구하는 식과 그에 따른 억제성 가중치의 갱신식은 다음과 같다.

$$overlap_{ij} = \sum (\Delta exin_i^5 \times \Delta exin_j^5) \quad (17)$$

if $exin5i > exin5j$

$$\Delta IW_{ij}^5 = \eta_1^5 \cdot exin_j^5 (overlap_{ij} - IW_{ij}^5) \quad (18)$$

if $exin5i < exin5j$

$$\Delta IW_{ij}^5 = \eta_2^5 \cdot exin_j^5 (0 - IW_{ij}^5) \quad (19)$$

2.5.3 시간적 위치에 무관한 패턴인식

어떤 패턴의 시간적 위치가 변한다고 하더라도 인식이 가능한 것을 시간적 위치에 무관한 인식이라고 한다. 예를 들어 ABCD라는 패턴과 CDAB라는 패턴 안에서 패턴 AB의 시간적 위치는 다르다. 즉 네 번째 층의 A를 나타내는 뉴런과 B를 나타내는 뉴런은 ABCD의 경우 각각 0과 θ^{34} 의 각 성분을 출력한다. CDAB에서는 각각 $2\theta^{34}$ 와 $3\theta^{34}$ 의 각 성분을 출력한다. 이 경우에 AB를 인식하는 다섯 번째 층상의 뉴런이 존재하고 더 이상 학습이 가능한 뉴런이 남아 있지 않은 경우, ABCD와 CDAB가 입력되었을 때 AB를 인식하는 뉴런은 두 경우 모두 활성화가 되어야 한다. 이를 가능하게 하기 위하여 인식 시에 k라는 변수를 사용하여 가중치의 각 성분을 조절해 준다. 다음 식에서 WP는 가중치의 각 성분이고 wp는 학습에 의하여 얻어진 가중치의 각 성분이다.

$$WP_{ij}^5 = wp_{ij}^5 + k \cdot \theta^{34} \quad (20)$$

$$k_j = - \frac{wp_{ij}^5 + inp_i^5}{\theta^{34}} \quad (21)$$

$\theta^{34} = 360 / (\text{네 번째 층의 뉴런의 개수})$ 이고, inp_i^5 는 입력의 각 성분이다. 여기서 중요한 것은 인식하는 패턴의 맨 처음 요소를 나타내는 네 번째 층의 뉴런이 활성화되면 k가 구해지고 그 다음, 시간적으로 정렬된 가중치가 계산되어 입력이 그 가중치와 곱해져서 다섯 번째 층의 뉴런으로 입력된다. 이렇게 구해진 k값은 고정되어 다섯 번째 층상의 그 뉴런에 연결된 다른 가중치를 구할 때도 사용된다.

3. 실험 및 고찰

제안된 시스템의 성능을 알아보기 위하여 layer1 to layer3 system은 Avalanche network[5]과 그리고 layer3 to layer5 system은 실수경쟁학습과 활성화 함수를 이용하여 시간의 정보를 저장하는 방법을 쓴 시스템과 비교를 하였다.

3.1 Layer1 to layer2 subsystem과 Avalanche network과의 비교

본 실험에서는 layer1 to layer3 system과 Avalanche network과의 필요뉴런의 수, 복잡도, 용량, 학습과정, 인식과정을 비교하였다. 실험에 사용된 입력데이터는 임의적으로 발생된 12개의 12x6 행렬로 행렬의 각 원소는 -1에서 1 사이의 실수로 이루어져 있다. 12개의 패턴 중 2개는 서로 오버랩이 83%인 행렬을 사용하여 패턴간 오버랩이 많아도 학습이 가능한지 알아보았다. 표2는 실험에 사용된 입력행렬의 예이다.

실험에 사용한 시스템은 모두 12개의 패턴을 인식할 수 있도록 설계되었고 그에 따라 layer1 to layer3 system의 경우 40개, Avalanche network의 경우는 180개의 뉴런이 필요했다. 표3을 보면 제안하는 시스템이 Avalanche network보다 뉴런의 개수에서 훨씬 효율적이라는 것을 알 수 있고 입력에서 시작해서 출력을 내기까지 곱셈의 복잡도는 제안된 시스템이 조금 낮은 것을 알 수 있는데 그 이유는 세 번째 층으로의 입력을 계산할 때 단위원상의 두 복소수의 곱은 그 복소수들이 가지는 각성분의 덧셈으로 대체될 수 있기 때문이다. 표에서 n은 입력행렬의 개수이고 p는 입력행렬의 열의 개수이다. 학습방법을 비교해 보면 제안하는 시스템이 경쟁에 의한 자율학습을 사용하는 반면 Avalanche network는 미리 학습할 층을 정해주어야 하는 단점이 있다.

표 2 실험에 사용된 12x6 행렬

Table 2 12x6 input matrix used in the simulation

	1	2	3	4	5	6
1	-0.2301	0.1293	0.4762	-0.5774	0.0784	-0.6034
2	-0.1758	0.1413	0.3380	-0.2224	0.3900	0.7958
3	-0.5690	-0.3887	-0.3170	-0.4574	-0.4505	0.1117
4	-0.1143	-0.3625	-0.1298	-0.5158	-0.2250	0.7225
5	0.0478	0.4372	-0.4798	0.5948	-0.1060	0.4597
6	0.1198	-0.0500	-0.8378	0.0212	-0.4615	-0.2603
7	0.5072	0.1370	0.3466	-0.4277	-0.4636	-0.4539
8	-0.3109	0.0780	-0.4595	-0.0365	0.8250	-0.0646
9	0.4126	-0.4207	0.2167	-0.7469	0.0074	-0.2188
10	0.4018	-0.2814	0.1177	-0.6785	-0.3244	-0.4242
11	0.6575	0.3100	0.0150	-0.1825	0.5631	0.3479
12	-0.4026	-0.3855	-0.4074	0.3869	0.0940	0.6040

제안된 시스템은 오버랩이 적은 패턴뿐 아니라 오버랩이 많은 패턴도 버퍼의 도움으로 학습에 성공하였다.

표 3 Layer1 to layer3 system과 Avalanche network과의 성능비교

Table 3 Performance comparison between layer1 to layer3 system and Avalanche network

	Layer1 to layer3 system	Avalanche network
뉴런의 개수	6+3n 개	6+n ² 개
곱셈의 복잡도	O(n ³ -n ² +pn)	O(pn ³ +n)
용량	Layer3의 뉴런의 개수	입력층을 제외한 층의개수
패턴의 학습성공률	100%	100%
학습된 패턴의 인식률	100%	100%

Avalanche network의 경우 학습방법상 당연히 100% 학습이 되었지만 본질적으로 비교대상이 될 수 없었다. 학습된 패턴의 인식률의 경우 두 시스템 모두 만족스런 결과를 보여주었다. 표에는 없지만 학습된 패턴에 임의로 노이즈를 첨가하여 인식률 시도해 보았는데 Avalanche network의 경우 패턴행렬의 마지막 행벡터가 변형이 심할 경우에는 패턴

을 인식하지 못하는 경우가 발생하였으나 제안된 시스템은 노이즈의 위치에 상관없이 잘 인식하였다. 또한 제안하는 시스템은 입력패턴이 학습된 패턴에 어느 정도 가깝지 않으면 전혀 활성화 되지않고 그 정도를 조절할 수 있는 반면 Avalanche network는 입력패턴과 학습된 패턴의 차이가 있더라도 활성화가 되었고 그 정도를 조절할 방법이 없었다.

결과적으로 제안된 시스템은 Avalanche network보다 뉴런의 개수면에서 효율적이었는데 이것은 첫 번째 층과 두 번째 층에서 수행하는 시공간 패턴을 공간패턴으로 압축하는 방법이 효율적이었다는 것을 의미한다. 또한 완전히 자율적으로 학습을 수행하였으며 학습된 패턴의 인식도 다소 우수하였다.

3.2 Layer3 to layer5 system과 자기조직화 경쟁네트워크의 비교

본 실험에서는 두 시스템간의 성능을 비교하기 위하여 복잡도, 학습성공률, 인식율을 비교하였다. 특히 학습된 패턴의 상위패턴(super-pattern)과 하위패턴(sub-pattern)의 학습과 인식이 가능한지 실험해 보았다. 비교대상 시스템은 기존의 자기조직화 경쟁학습 네트워크에 활성화 함수를 이용해 점진적으로 증가하는 (monotonically increasing)입력패턴을 만드는 방법[3]을 접합시킨 시스템이다. 이 시스템을 비교대상으로 삼은 이유는 제안하는 시스템이 이 기존의 시스템에서 발전된 것이기 때문이다. 실험에 사용된 시스템은 모두 12개의 패턴을 인식할 수 있도록 설계되었고 입력 데이터는 두 시스템의 성능을 비교하기 위하여 표4와 같이 정해졌다. 표4는 세 번째 층의 i번째 뉴런과 자기조직화 경쟁학습 네트워크의 입력층의 뉴런을 Ni로 나타낸다고 하고 이 뉴런들이 활성화되는 순서를 나타낸 것이다.

표 4 실험에 사용된 입력패턴

Table 4 An input pattern used in the simulation

1	N ₂ - N ₃
2	N ₇ - N ₁₀
3	N ₅ - N ₁₂
4	N ₁ - N ₃ - N ₅ - N ₇
5	N ₂ - N ₉ - N ₁₀ - N ₁₂
6	N ₅ - N ₆ - N ₇ - N ₁₂
7	N ₂ - N ₄ - N ₅ - N ₇ - N ₈ - N ₉ - N ₁₁ - N ₁₂
8	N ₃ - N ₂ - N ₁ - N ₆ - N ₁₀ - N ₇ - N ₁₁ - N ₁₂
9	N ₃ - N ₅ - N ₆ - N ₇ - N ₂ - N ₁₀ - N ₉ - N ₁₂
10	N ₁₁ - N ₂ - N ₃ - N ₁ - N ₁₂ - N ₄ - N ₆ - N ₅ -
	N ₈ - N ₇ - N ₉ - N ₁₀
	N ₃ - N ₁ - N ₄ - N ₂ - N ₇ - N ₁₀ - N ₁₂ - N ₁₁ -
11	N ₆ - N ₅ - N ₈ - N ₉
	N ₁₂ - N ₁₀ - N ₈ - N ₆ - N ₄ - N ₂ - N ₁ - N ₃ -
12	N ₅ - N ₇ - N ₉ - N ₁₁

표5에서 보는 바와 같이 곱셈의 복잡도는 제안된 시스템이 다소 낮은 수준이나 제안된 시스템의 경우 필터를 구현

할 때 분기문을 사용해야 하므로 실제로는 복잡도가 조금 높다고 할 수 있다. 표에서 n은 입력층의 뉴런의 개수이다. 그러나 패턴의 학습성공율은 버퍼와 필터의 도움으로 기존의 시스템에 비하여 탁월하였다. 기존의 시스템의 경우 학습하는 패턴간의 오버랩이 많고 길이가 길어질수록 학습이 어려웠다. 학습된 패턴에 대해서는 두 시스템 모두 좋은 인식율을 나타내었다. 표4의 입력패턴과는 별도로 기준패턴을 학습시킨 후 상위패턴과 하위패턴을 학습시키는 실험을 실행하였는데 기존의 시스템으로는 전혀 불가능하였다. 그밖에도 동시인식이나 시간적 위치에 무관한 인식 또한 기존 시스템으로는 거의 불가능하였다. 결과적으로 제안된 시스템은 기존의 시스템보다 다소 계산량이 많지만 학습성공율, 인식율 등 모든면에서 월등한 것이 입증되었다.

표 5 Layer3 to layer5 system과 기존의 시스템과의 성능비교

Table 5 Performance comparison between layer3 to layer5 system and conventional self-organizing competitive network with time order coding by the activation function

	Layer3 to layer5 system	기존의 시스템
복잡도	$O(n^3 - n^2 + 2n)$	$O(3n^3 - 2n^2 + n)$
패턴의 학습성공율	100%	42%
학습된패턴의 인식율	100%	100%
상위, 부분패턴인식	가능	불가능
시간의 위치에 무관한 인식	가능	시간적 위치의 차이가 크면 불가능
동시인식	가능	불가능

4. 결론 및 향후연구

본 논문에서 제안한 시공간 패턴 인식 신경회로망은 -1에서 1사이의 실수로 이루어진 시공간 패턴을 입력으로 받아서 인식하는 역할을 잘 수행하였다. 시간의 흐름을 나타내는데 있어서 기존방법의 한계를 극복하고 정규화 없이 경쟁학습을 가능하게 하기 위하여 복소수를 사용하였다. 또한 복소수를 이용한 뉴런간의 억제를 이용하여 기존의 어떤 경쟁학습 신경회로망보다 효율적으로 상위패턴의 학습을 수행하였다. 또한 오버랩을 이용하여 억제의 정도를 결정함으로써 동시인식을 가능하게 하였고 k 변수를 이용하여 시간적 위치에 무관한 인식을 가능하게 하였다. 그밖에도 습관성에 근거한 학습을 변조와 학습된 가중치의 고정 등으로 시스템의 성능을 향상시켰다. 자율학습을 채택해서 스스로 새로운 입력에 대하여 클래스를 생성하는 것도 장점 중의 하나이다. 임의의 값 즉 -1과 1사이의 실수로 이루어지는 제한된 크기의 시공간 패턴을 인식하려면 세 번째 층 이하의 하위 시스템만으로도 충분하지만, 그 패턴들의 연속으로 이루어지는 더 큰 패턴의 인식을 위해서는 계층적으로 상위에 있는 네 번째 층 이상의 상위시스템이 필요하다.

우리가 살아가면서 가장 많이 하는 것이 바로 패턴인식이다. 보는 것, 듣는 것, 냄새 맡는 것, 만지는 것, 이 모두가 패턴인식이라고 할 수 있다. 이렇게 인간생활의 근간이 되는 패턴인식을 수행하는 신경회로망은 적용분야가 매우 많다고 생각되며 앞으로 인공지능을 가진 시스템의 개발 등 많은 분야에서 쓰여질 것이라고 확신한다. 앞으로 신경회로망을 구현하는데 있어서 현재 가지고 있는 속도의 문제, 뉴런의 개수의 문제 등을 극복할 수 있다면 신경회로망은 앞으로 더 많이 적용되어질 것이다.

앞으로 시스템을 음성인식 등 실제 패턴인식 분야 응용분야에 맞게 수정, 응용하는 것이 필요하고, 나아가서는 보다 효율적으로 패턴인식을 수행하도록 하드웨어 제작이 수행되어야 하겠다.

감사의 글

본 연구는 1997년도 인하대학교 교내 연구비 지원에 의하여 수행되었습니다.

참고 문헌

- [1] James A.Freeman and David M.Skapura, Neural networks Algorithms, Applications, and Programing techniques, Addison-Wesley publishing company, 1991.
- [2] Yifeng Chen and Yuanda Cao, "A Hybrid Neural Network for Spatio-temporal Pattern Recognition", Proceedings of ICNN, Vol13. pp.1414-1417, 1995.
- [3] Albert Nigrin, Neural Networks for Pattern Recognition, The MIT Press, 1993.
- [4] Bryan W.Stiles and Joydeep Ghosh, "Habituation Based Neural Networks for Spatio-temporal Classification", Neurocomputing, No15, pp.273-307, 1997.
- [5] 김현기, 박경철, 이종호, "시공간 패턴 인식기를 이용한 숫자음 인식", 인공지능, 신경회로망 및 퍼지 관련 학술대회 논문집, pp.122-127.

저 자 소 개



임 정 수(林政秀)

1971년 1월 30일 생. 1996년 인하대 전기공학과 졸업. 1999년 동 대학원 전기공학과 졸업(석사)

Tel : 587-2936

E-mail : koguma@unitel.co.kr

이 종 호(李種浩)

전기학회 논문지 48A권 2월호 참조

Tel : (032) 860-7396