濟州大學校 産業技術硏究所 論文集, 第8卷, 2號, pp.36~43, 1997. 12
Cheju National Univ. R.I.I.T. Jour., Vol.8, No.2, pp.36~43, 1997. 12

신경회로망을 이용한 한글 자소 인식

김 덕 주* · 임 재 윤**

Han-geul Characters Recognition by Neural Networks

Deog-Ju Kim* and Jea-Yun Lim**

Abstract

An efficient algorithm for Han-geul character recognition by neural networks is proposed. After normalizing input patterns, input neurons detect local distribution of pixels and determine input vectors. Neural networks were constructed in three layers and trained by Error BackPropagation method. During the process of training, neural network's weight were determined. Han-geul characters were recognized by neural networks after training. These algorithms are programed by C-language on IBM PC 586. The learning is repeated until the difference between object output and real output is less than 0.00001. After setting the hidden neuron's number optimally, learning velocity and recognition rate are analyzed as momentum α and learning rate η vary from 0.1 to 0.5. Through the trained neural networks, it is shown that recognition ratio is 57.6~100% for learning patterns.

Key words : Neural networks, Character recognition, Momentum, Learning rate.

I. 서 론

현대사회의 방대한 정보량의 증가는 활자를 매체로 하는 수용 한계를 넘어섰으며, 이미 컴퓨터를 활용하지 않으면 안되게 되었다. 특히, 기존에 문서화되어 있는 많은 데이터와 정보를 데이터베이스화하여 원하는 정보를 신속히 찾아내기 위해서는 사람의 손으로 입력하는 것보다 신속하고 정확하게 데이터를 입력할 수 있는 장치의 개발이 요구된다. 이러한 데이터 입력장치의 개발이 선행되지 않는다면 컴퓨터의 정보처리 속도와 연산 속도가 아무리 향상되어도 효율적인 정보처리를 할 수 없을 것이다. 이러한 분야는 패턴 인식의 발달로 정착되어 가고 있다.[1-3]

대부분의 병렬 처리 문제들은 순차적인 해결

* 제주관광전문대학 관광정보처리과
Dept. of Information Management, Cheju Tourism College
** 제주대학교 통신공학과
Dept. of Telecommunication Eng., Cheju Nat'l Univ.

알고리즘으로는 모든 경우의 수에 대한 처리시간이 시스템의 처리 시간 중 대부분을 차지하게 되어 전체 처리 시간이 길어진다. 또한 순차형 컴퓨터를 이용한 문자 인식 방법에는 처리 데이터를 순차적으로 처리하게 되므로 처리 속도에 한계가 있다. 이에 비해 분산된 뉴런들의 처리를 이용한 신경회로망에서의 병렬 처리에 의한 문제 해결 방법은, 각 뉴런의 처리 과정이 병렬로 이루어져서 처리 시간에 있어서 효율적이어서 많은 관심을 모으고 있다.[1] 특히 신경회로망은 순차형 컴퓨터의 계산 방식으로는 해결하기 어려운 여러 가지 문제 특히, 패턴 인식 문제에 성공적으로 응용되고 있으며, 필기체 영문자와 숫자 및 필기체 한글 인식에도 가능성을 제시하고 있다.[5)-7)]

본 논문에서는 신경회로망을 이용하여서 한글 자소 패턴을 분류 인식하는 시스템을 제안함으로서 패턴인식 문제에 신경회로망의 적용이 유용함을 보인다. 본 시스템에서는 마우스를 통하여 20×20개의 화소로 구성된 입력 패턴에 대해 화소의 국소 분포를 이용해 패턴의 방향 성분을 추출하며, 추출된 방향 성분은 신경회로망의 입력 벡터로 주어 문자를 학습시킨다. 문자 인식 시스템은 3층 순방향 전파 신경회로망으로 구성되며 학습에는 오류 역전파(EBP : error back-propagation) 방법을 사용한다.[8)] 문자 인식 시스템의 특성과 학습속도와 인식능력을 분석하기 위하여 중간층 뉴런의 수와 학습율 그리고 변화율을 변경시켜가면서 시뮬레이션 한다.

II. 문자 인식 시스템으로서의 신경회로망

인공 신경회로망의 수학적 모델은 뉴런과 시냅스로 구성된다. 한 뉴런으로부터 전달된 출력 신호는 시냅스 연결을 통하여 다른 뉴런으로 전달된다. 한 뉴런의 입력 신호의 상태는 각각의 가중치가 시냅스 연결의 세기가 되는 다른 뉴런으로부터 전달되는 가중된 입력 신호의 선형적인 합에 의해 결정된다. Fig.1의 회로망은 전방향 회로망이다.

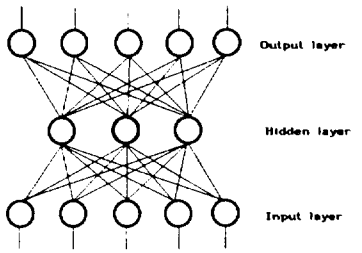

Fig. 1 Construction of neural networks

신경회로망에 존재하는 뉴런은 뇌에서의 신경세포에 해당하며 활성화 값이라 불리는 하나의 출력값을 계산하는 작용을 한다. 뉴런은 Fig.2와 같은 구조로 되어 있으며 한 뉴런의 활성화 값은 전단에서 그 뉴런에 직접 연결되어 있는 뉴런들의 출력값과 해당 연결선의 가중치를 이용하여 계산된다. 전형적인 신경회로망 모델에서는 다른 뉴런들로부터의 입력에 연결선의 가중치를 곱하여 그 합을 구하고 그 값이 임계치를 초과하는가를 응답 함수로 계산하여 초과하면 그 값을 그 뉴런의 출력값으로 한다.

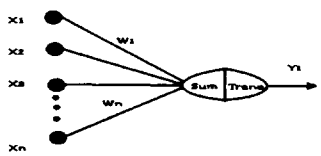

Fig. 2 Construction of processing element

$$U_i = \sum_k W_{ki} V_k \qquad (1)$$

식(1)에서 V_k는 k번째 뉴런의 출력이고, U_i는 i번째 뉴런의 입력이다. W_{ki}는 k-번째

뉴런으로부터 i-번째 뉴런으로 전달되는 시냅스 연결의 세기이다.

MaCulloch-Pitts의 뉴런 입/출력 함수는 식(2)와 같다.

$$V_i = f(U_i) = \begin{cases} 1 & \text{if } U_i > 0 \\ 0 & otherwise \end{cases} \quad (2)$$

여기서, V_i와 U_i는 각각 i-번째 뉴런의 출력과 입력이다.

이러한 응답 함수는 그 특성에 따라 Fig.3와 같이 구분할 수 있다.

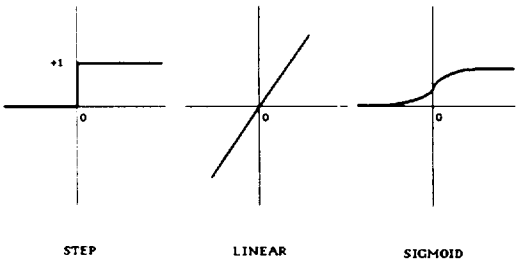

Fig. 3 Response functions of processing element

신경회로망에서 지식은 뉴런들이 어떻게 상호 연결되어 있고 각 연결선의 가중치가 어떤 값을 갖고 있느냐에 따라 결정되어 저장된다. 즉 지식이란 특정 장소의 내용이 아니라 회로망 구조에 의해 나타내어진다. 뉴런은 네트워크 상의 역할에 따라 입력 뉴런, 중간 뉴런, 출력 뉴런의 세 계층으로 나뉘어진다. 입력 뉴런은 회로망의 외부로부터 입력을 받아들여 활성화되어 입력 값을 그대로 출력한다. 출력 뉴런은 회로망 전체의 출력으로 사용된다. 입력 뉴런이나 출력 뉴런에 속하지 않는 뉴런을 중간 뉴런이라 하며 선형 비분리 함수와 같은 복잡한 기능을 수행하는데 필요하다. 신경회로망을 구조에 따라 방향성 사이클이 존재하지 않는 전방향 회로망과 방향성 사이클이 포함되어 있는 역방향 회로망으로 나눌 수 있다. 신경회로망에서 학습이란 주어진 신경회로망이 우리가 원하는 동작을 수행할 수 있는 가중치를 찾아내는 작업을 의미한다. 이러한 학습은 입력 값과 우리가 기대하는 출력값을 가지고 가중치를 조정해 나감으로써 이루어진다.

신경회로망이란 신경조직에서 착안하여 모델화한 정보처리 시스템으로서 단순한 소자들이 병렬, 분산 연결 구조를 가지고 있다. 외부로부터 입력을 받아들여 동적인 반응을 일으킴으로써 필요한 출력을 생성시킨다. 따라서 신경회로망은 기존의 순차형 컴퓨터에서 비효율적으로 처리되던 패턴 인식 문제를 효율적으로 해결하는데 매우 적합하다. 즉, 신경회로망은 패턴 인식에 필요한 많은 양의 데이터를 병렬 처리 능력을 이용하여 인식 대상 문자수에 상관없이 인식에 걸리는 시간을 일정하게 할 수 있다. 디지털 컴퓨터의 순차적인 정보처리 방식과는 달리 수많은 뉴런들에 분산 저장되어 있는 정보가 대규모 병렬 연산에 의해 즉각적으로 처리된다.

그리고 패턴 인식과 같은 수학적 알고리즘의 적용이 곤란한 문제도 학습에 의하여 효과적으로 처리할 수 있다. 신경회로망에서 가장 중요한 특징 중의 하나가 학습 능력이다. 신경회로

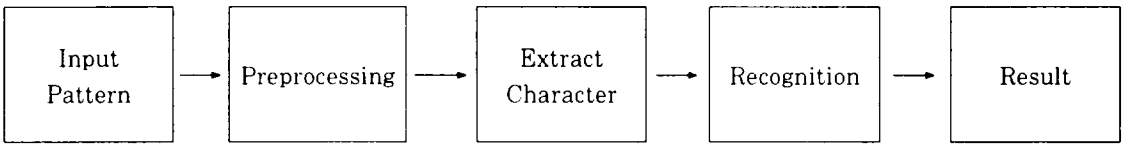

Fig. 4 Construction of pattern recognition

망에서의 학습이란 주변으로부터의 자극과 응답에 대한 경험적 정보를 일반화하고 그 데이터로부터 신경회로망 내부에 지식 기반을 형성, 입력 정보에 대해 적절하고 융통성 있는 출력을 생성하도록 하는 것이다. 학습의 궁극적인 목표는 원하는 출력과 가중치를, 목표치와 현 출력과의 오차를 최소화하도록 바꿔 나가는 것이다.

신경회로망은 패턴 인식에 있어 자주 발생하는 잡음을 포함하거나 애매한 데이터를 효과적으로 처리할 수 있다. 신경회로망을 이용한 시스템은 학습과 기억을 통하여 문자를 인식하는 동적인 시스템이다. 이러한 동적인 특성으로 인하여 훈련되지 않은 다른 문자체에도 약간의 학습 과정을 통하여 쉽게 적응할 수 있다. 신경회로망 시스템은 간단한 학습에 의해 개발이 가능하므로 다른 방식에 비해 개발 기간이 훨씬 단축된다.

III. 문자 인식용 신경회로망 모델 구현

본 논문에서 문자 인식을 위해 한글자모26자에 대한 입력 패턴을 준비하고 이를 신경회로망에 입력으로 주어 학습시킨다. 문자 인식은 우선 입력 패턴의 정형화, 잡음 제거, 세선화 과정을 거치고 특징점 추출을 하여야 한다. 이렇게 추출된 특징 정보를 신경회로망을 통하여 분류시킴으로서 문자 인식이 가능하게 된다.

일반적인 문자 인식 시스템은 Fig.4와 같이 구성된다. 입력 자료는 잡음을 가지고 있으므로, 시스템에서 인식을 위해 필요한 자료만을 얻어내는 전처리를 하게 된다. 다음은 입력된 패턴을 인식하기 용이한 작은 단위로 분할하는 과정을 거친다. 특징 추출 단계에서는 인식에 중요한 실마리가 되는 특징을 얻어내고, 이 특징을 비교하여 패턴을 식별하게 된다. Fig.4의 각 단계는 고정된 것이 아니며 순서를 일부 바꾼다든지, 일부를 생략할 수 있다.

정형화 및 세선화 과정을 거친 패턴은 20×20의 비트 행렬이 되는데 이를 특징점 추출과 분류를 위해 특징점 추출 알고리즘을 이용한다. 문자는 획을 분류해 보면 가로, 세로 혹은 원 등의 선분들의 집합이다. 이러한 선분들의 특징 및 국소적 분포 성분을 그 문자의 특징 벡터로 추출하기 위하여 국소 분포 성분 행렬을 Fig.5와 같이 추출한다.

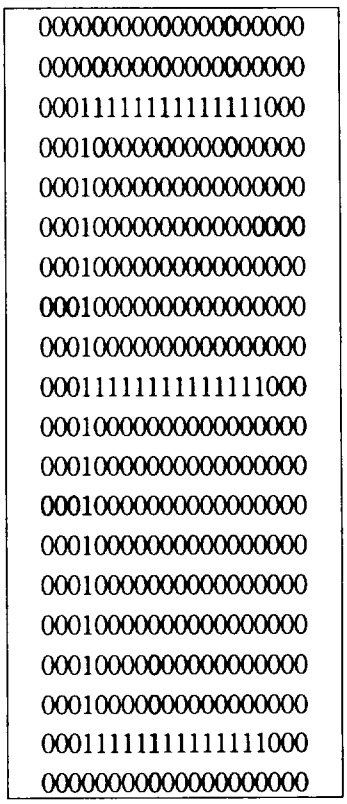

Fig. 5 Detection of local distribution

4비트의 검색창이 입력 패턴을 50개의 국소 분포 성분으로 특징 행렬을 정의할 수 있다. 검색창에 '1'인 화소가 검출되면 그 지역의 특징 행렬은 '1'의 값을 갖는다. Fig. 6는 국소 분포로부터 얻어진 50개의 특징 행렬이다.

본 논문에서 사용한 신경회로망의 구조는 입력층과 출력층 그리고 그 사이에 중간 처리층을 두어 3개의 층으로 구성되어진다. 입력층은 50개의 입력단을 갖고 있고 이 입력단을 통하여

특징 추출 행렬을 입력으로 받아들여진다. 그리고 중간층은 10개에서 60개의 뉴런으로 변화시키면서 구성하였고 출력단은 26개의 뉴런으로 문자 갯수와 일치 시켰다. 각 뉴런간의 연결은 전방향으로 모두 연결되어 있다. 각 뉴런은 연결되어 있는 전단의 출력과 가중치를 곱한 값의 총 합을 식(3)과 같이 구하여 이 값을 입력으로 하여 문턱치와 비교 처리한 후 이를 출력하여 상위층 뉴런으로 전달된다.

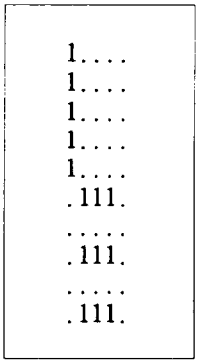

Fig. 6 Detection feature on local distribution

$$net_j = \sum_i o_i \cdot w_{ji} + \theta_j \tag{3}$$

$$o_j = \frac{1}{(1+e^{-net_j})} \tag{4}$$

여기서 첨자 i와 j는 뉴런의 번호이고 o_i는 i번째 뉴런의 출력이다. w_{ji}는 j번째 뉴런과 i번째 뉴런간의 가중치이며, θ_j는 j번째 뉴런의 문턱치이다. 따라서 net_j는 j번째 뉴런으로 들어오는 입력의 합이며 이 값은 식(4)와 같은 응답함수를 통과하여 o_j라는 j번째의 출력값이 결정되어 진다. 식(4)의 응답함수는 시그모이드 함수로서 미분 가능하고 단조 증가하는 특징을 가지고 있다.

신경회로망의 학습 방법은 Rumelhart (1986)가 제안한 오류 역방향 전파(EBP : Error BackPropagation)를 사용하였다. 입력층과 중간층 그리고 출력층으로 구성된 신경회로망을 오류 역방향 전파(EBP) 알고리즘을 통하여 필요한 정보를 저장하도록 학습시킬 수 있다. 역방향 전파는 출력의 기대치와 실제 값의 오차를 감소하는 방향으로 연결 강도를 조절하고 상위층의 오차를 현재의 연결 강도를 가중치로 하여 다음 하위층에 역전파하며 하위층에서는 이를 근거로 하여 자기층의 연결 강도를 조정해 나간다. 이 신경회로망을 3계층으로 구성할 경우 이론상으로는 어떠한 형태의 패턴도 형성할 수 있으나, 학습시 국소극소에 빠질 우려가 있고, 학습 과정이 너무 오래 걸리고, 또 기억된 패턴의 수정, 추가 학습 등이 불가능하다는 단점이 있다. 하지만 이 신경회로망은 구현이 쉽고 학습 방식이 용이하기 때문에 현재 가장 폭넓게 응용되고 있다. 어떤 입력을 주었을 때 의도 출력값 (t_j)와 실제의 출력값 (o_j)와의 차를 식(5)와 같이 정의한다.

$$E = \frac{1}{2}(t_j - o_j)^2 \tag{5}$$

식(5)에서 E는 j번째 출력단의 오차값을 나타낸다. 신경회로망의 학습은 이 오차를 줄이도록 접합 가중치를 조정하는 것이다. 여기서 입력을 주었을 때의 j번째와 i번째의 가중치 w_{ji}의 변화량은 식(6)과 같다.

$$\Delta w_{ji} = -\frac{\partial E}{\partial w_{ji}} \tag{6}$$

식(6)의 우변은 식(7)과 같이 다시 쓸 수 있다.

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial net_j} \frac{\partial net_j}{\partial w_{ji}} \tag{7}$$

식(3)에서 i대신 k를 변수로 쓴다면

$net_j = \sum_k o_k \cdot w_{jk}$ 이므로

식(7)의 우변의 우측은 $\frac{\partial net_j}{\partial w_{ji}} = \frac{\partial}{\partial w_{ji}} \sum_k o_k \cdot w_{jk} = o_i$가된다.

또한 좌변 식은 δ_j를 j번째 항의 입력 합의 변화량에 대한 오차의 변화량이라두면 $\delta_j = -\frac{\partial E}{\partial net_j}$ 이고, $\frac{\partial net_j}{\partial w_{ji}} = o_i$ 로 부터 $-\frac{\partial E}{\partial w_{ji}} = \delta_j \cdot o_i$가 되어 간략화 된다.

그러므로 $\Delta w_{ji} = \eta \cdot \delta_j \cdot o_i$가되며 여기서 η는 상수로서 학습률이다. δ_i는 뉴런이 중간층이냐 출력층이냐에 따라 다르다. 출력층 뉴런일 경우 $E = \frac{1}{2}(t_j - o_j)^2$, $o_j = f_j(net_j)$이므로 식(8)과 같다.

$$\begin{aligned} \delta_j &= -\frac{\partial E}{\partial net_k} \\ &= -\frac{\partial E}{\partial o_j} \cdot \frac{\partial o_j}{\partial net_k} \\ &= (t_j - o_j) \cdot f_j'(net_j) \end{aligned} \quad (8)$$

중간층일 경우, $(t_j - o_j)$에 대응하는 항은 식(9)와 같다.

$$\begin{aligned} & -\sum_k \frac{\partial E}{\partial net_k} \cdot \frac{\partial net_k}{\partial o_j} \\ &= -\sum_k \frac{\partial E}{\partial net_k} \cdot \frac{\partial}{\partial o_j} \sum_i w_{ki} \cdot o_i \\ &= -\sum_k \frac{\partial E}{\partial net_k} \cdot w_{kj} \\ &= \sum_k \delta_k \cdot w_{kj} \\ \delta_j &= f_j'(net_j) \sum_k \delta_k \cdot w_{kj} \end{aligned} \quad (9)$$

Δw의 계산은 출력단으로부터 시작하여 중간층 뉴런에 전달된다. 중간층 뉴런의 Δw를 계산하기 위해서는 앞단의 Δw가 계산되지 않으면 안된다. 그러므로 최후의 입력단까지 거슬러 올라가야만 처음으로 계산이 가능하게 된다. 이러한 역방향 전파가 이루어져야 학습을 할 수 있으므로 역방향 전파 학습법이라 한다. 일반적인 Δw 를 정식화한 것이 식 (10)과 같다.

$$\Delta w_{ji}(n+1) = \eta \cdot \delta_j \cdot o_j + \alpha \cdot \Delta w_{ji}(n) \quad (10)$$

여기서 n은 학습 횟수, α는 변화율이며, $\alpha \cdot \Delta w_{ji}(n)$은 오차의 진동을 적게 하여 수렴속도를 빨리 하기 위하여 첨가한 것이다.

본 논문에서는 학습 횟수는 오차가 0.00001 이하 일 때까지로 정하였으며 중간층 뉴런의 갯수와 학습속도 및 인식률에 따른 관계를 살펴보고 최적의 중간층 뉴런수를 선정하여 다시, 변화율 α를 0.1에서 0.5까지 변화시키면서 학습속도와 인식률을 분석하였다. 설정된 중간층 뉴런수에서의 학습률 η의 영향을 분석하기 위하여 1.5 - 2.0을 주어 학습 및 인식을 시켰다.

IV. 알고리즘 흐름도 및 시뮬레이션 결과

본 논문에서 제시한 문자 인식 알고리즘을 구현해 보기 위하여 C언어로 시뮬레이션 하였으며 Fig.7은 알고리즘 흐름도 이다.

Table 1은 문자 인식 시스템의 중간층 뉴런의 수에 따른 학습속도와 인식률의 변화를 알아보기 위하여 중간층 뉴런의 수를 10개에서 60개까지 변화시키면서 학습시키고 학습이 완료된 후 인식률을 알아보았다. 이때의 학습률은 1.5이고 변화률도 0.5로 정하였다. Table 1에서 알 수 있듯이 중간층 뉴런의 수가 많아지면 학습속도는 빨라지지만 인식률에 있어서는 30개 이상일 경우 오히려 떨어지고 있다. 이는 적당한 수까지의 증가는

정보의 분산효과가 커서 각 뉴런이 학습할 부분이 적어지므로 효율적이지만 필요이상으로 많아질 경우 필요이상의 정보분산으로 학습률이 떨어져 인식률도 떨어지고 있음을 알 수 있다.

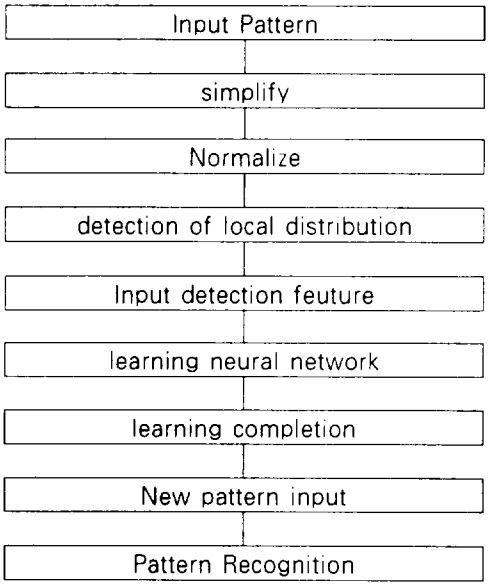

Fig.7 System flowchart

Table 1 Number of learning by hidden neurons (momentum=0.5)

Number of hidden neuron	Number of learning	recognition ratio % (learning pattern)
10	933	96.2
20	405	100.0
30	236	65.4
40	223	69.2
50	210	61.5
60	138	57.6

Table 2는 영문자 인식 시스템의 변화율 α의 변화에 따른 학습속도와 인식률의 변화를 알아보기 위하여 변화율을 0.1에서 0.5까지 변화시키면서 학습시키고 학습이 완료된 후 인식률을 알아보았다. 이때의 중간층 뉴런의 수는 인식률을 높게 나타낸 20개로 정하였고 학습율은 1.5로 정하였다. Table 2에서 보면 변화율의 증가는 학습속도를 빠르게 하였다. 그러나 필요이상의 학습은 필요치 않으므로 변화율의 조정으로 적정의 학습횟수를 정하는 것이 좋다.

Table 2 Number of learning by momentum

Momentum	Number of learning	recognition ratio % (learning pattern)
0.1	815	100
0.2	730	100
0.3	588	100
0.4	502	100
0.5	405	100

Table 3. Number of learning by learning rate

Learning rate	Number of learning	Recognition ratio % (learning pattern)
1.5	405	100
1.6	354	100
1.7	338	100
1.8	337	100
1.9	364	100
2.0	355	92.0

Table 3는 문자 인식 시스템의 학습률 η의 변화에 따른 학습속도와 인식률의 변화를 알아보기 위하여 학습률을 1.5에서 2.0까지 변화시키면서 학습시키고 학습이 완료된 후 인식률을 알아보았다. 이때의 중간층 뉴런의 수는 인식률을 높게 나타낸 20개로 정하였고 변화율은 0.5로 정하였다. Table 3에서 보면 학습률의 증가는 학습속도를 빠르게 하지만 인식률에 있어서

는 많은 변화를 보이지 않아 적정한 학습률은 학습속도를 빠르게 하는데 중요한 요인이 됨을 알 수 있다. 그러나 학습률이 2.0이 이상에서는 학습속도는 빨라지나 인식률이 떨어져 더이상 문자 인식시스템에 좋은 요인으로 작용하지 못했다.

V. 결 론

본 논문에서는 문자 인식을 위해 전처리 과정 중 입력 패턴에 대해 특징을 추출하였다. 입력 패턴의 화소의 국소적 분포를 특성 벡터로 표현하여 이를 신경회로망에 입력으로 사용하였다. 이 입력 벡터로부터 역방향 전파 학습법으로 학습된 신경회로망은 새로운 입력 패턴에 대해서도 학습된 가중치와 문턱값, 그리고 특성행렬을 통하여 문자 인식을 수행함으로써 에러가 섞인 문자에 대해서도 인식할 수 있다. 본 논문에서는 학습 횟수는 오차가 0.00001 이하일 때까지로 정하였으며 중간층 뉴런의 갯수와 학습속도 및 인식률에 따른 관계를 살펴보고 최적의 중간층 뉴런수를 선정하여 다시, 변화율 α를 0.1에서 0.5까지 변화시키면서 학습속도와 인식률을 분석하였다. 설정된 중간층 뉴런수에서의 학습률 η의 영향을 분석하기 위하여 1.5 - 2.0를 주어 학습및 인식을 시켰다. 학습결과 중간층 뉴런의 수가 10일 경우 933회에서 학습을 마쳤고, 60일 경우는 138회에 학습을 마쳤다. 학습된 신경망을 통하여 학습 데이터의 인식률은 57.6~100%의 인식률을 보였다. 추후 연구 과제로는 필기체 한글의 인식과 다양한 패턴 분류에 관한 연구 등이다.

참 고 문 헌

1) Hussain, Basit and Kabuka,M. R., 1994.1, "A Novel Recognition Neural Network and its Application to Character Recognition", IEEE transactions on pattern analysis and machine intelligence, Vol.16 No.1, pp.98-106.

2) 오영환, 1993.10,"패턴 인식의 개관", 정보과학지 제 11권 5호, pp. 11-20.

3) 이광노, 장명욱, 박치선, 이훈복, 1993.1,"패턴 인식을 위한 신경망- 지식 기반 융합 모델", 전자 통신 제 14권 4호, pp. 125-136.

4) Kosko, Bart, 1987.12, "Adaptive bidirectional associative memories", APPLIED OPTICS Vol.26 No.23, pp. 4947-4959.

5) 남호원, 정호선, 1990, "영문자 인식 및 전처리용 신경칩의 설계", 한국전자통신학회논문지, Vol.15 No.6, pp. 455-466.

6) Fukushima, K., 1988, "A Neural Network for Visual Pattern Recognition", IEEE Computer, Vol. 21 No.3, pp. 65-75.

7) 이광노, 김명원, 1990.10, "Neural network을 이용한 필기체 한글 자소 인식", 전자 통신 제12권 3호, pp. 77-91.

8) Rumelhart, D. E., Hinton, G. E., and Williams, R. J., 1986, "Learning Internal Representations by Error Propagation", Parallel Distributed Processing, D.E. Rumelhart and J. L. McClelland, eds., Ch. 8, Cambridge, MA : MIT Press.