한국 퍼지 및 지능시스템학회 논문지 2000, Vol. 10, No. 3 (2000)270~279

확률 펄스 신경회로망의 On-chip 학습 알고리즘

On-chip Learning Algorithm in Stochastic Pulse Neural Network

김응수 · 조덕연* · 박태진*

Eung-Soo Kim, Duk-Yun Cho* and Tae Jin Park*

대전대학교 공과대학 전자공학과
*선문대학교 대학원

요 약

본 논문은 확률 펄스연산을 이용한 신경회로망의 on-chip 학습 알고리즘에 대해 기술하였다. 확률 펄스연산은 임의의 펄스열에서 1과 0이 발생할 확률을 통해 표현된 수를 사용하여 계산하는 것을 일컫는다. 이러한 확률연산을 신경회로망에 적용하면 하드웨어 구현면적을 줄일 수 있다는 것과 확률적인 특징으로 인해 지역 최소값으로부터 빠져 나와 광역 최적해에 도달할 수 있다는 장점을 갖고 있다. 또한 본 연구에서는 칩 내부에 학습할 수 있는 on-chip학습 알고리즘을 역전파 학습 알고리즘으로부터 유도하였다. 이렇게 유도된 알고리즘을 검증하기 위하여 비선형 패턴분리문제를 모의실험 하였다. 또한 활자체 및 필기체 숫자 인식에도 적용하여 좋은 결과를 얻었다.

ABSTRACT

This paper describes the on-chip learning algorithm of neural networks using the stochastic pulse arithmetic. Stochastic pulse arithmetic is the computation using the numbers represented by the probability of 1's and 0's occurrences in a random pulse stream. This stochastic arithmetic has the merits when applied to neural networks ; reduction of the area of the implemented hardware and getting a global solution escaping from local minima by virtue of the stochastic characteristics. And in this study, the on-chip learning algorithm is derived from the backpropagation algorithm for effective hardware implementation. We simulate the nonlinear separation problem of the some character patterns to verify the proposed learning algorithm. We also had good results after applying this algorithm to recognize printed and handwritten numbers.

1. 서 론

신경회로망은 패턴인식, 시계열 예측 등과 같이 인간의 인지나 추론 능력 등을 인공적으로 모델링하여 구현한 것으로, 최근에는 이러한 신경회로망을 우리의 현실세계에서 사용하기 위해서 실시간 처리를 할 수 있는 하드웨어 구현에 대한 연구가 활발히 진행되고 있다[2-5]. 신경회로망 연구에 있어서도 단순한 알고리즘 연구 뿐만이 아니라 하드웨어적으로 구현하는 연구에 대해서도 세계적으로 많은 연구가 수행되고 있다. 특히 급속히 발달하고 있는 VLSI기술을 이용하여 신경회로망을 집적화하여 응용분야에 적용하려는 연구가 급속히 진행되고 있다.

신경회로망 칩의 구현방법에는 크게 아날로그 회로에 의한 방법, 디지털 회로에 의한 방법 그리고 아날로그와 디지털의 장점들을 이용한 하이브리드 방법이 있다[9]. 아날로그 방법은 회로의 규모를 작게 그리고 연산속도를 빠르게 낼 수 있는 장점이 있고, 디지털 방법은 아날로그 방법에 비해 잡음에 강하고, 제작된 칩들로 시스템 구성이 비교적 용이하며, 알고리즘에 의한 구현설계 또한 용이하다는 장점들을 갖고 있다[2,3]. 또한 펄스를 이용한 하이브리드 방법은 아날로그와 디지털의 장점들을 최대한 살린 방법으로 평가받고 있다. 하지만 이 방법은 신경회로망의 기본 구성요소인 곱셈기나 덧셈기 회로의 구현 면적이 커서 고밀도의 칩 제작에 문제가 있었다.

본 논문에서는 하이브리드 방법의 장점을 살리며 이러한 문제를 해결하기 위해 확률펄스를 이용한 신경회로망의 설계에 대해 기술한다. 확률 신경회로망은 자신의 확률적인 특성으로 지역 최소값에서 빠져 나올 수 있는 능력이 있어 광역 최적화에 아주 적합한 모델이다. 확률 신경회로망에서 정보는 대개 흐트러져 있고 잡음이 섞여 있다. 신경회로망의 출력은 단일 뉴런에 의해 영향을 받지 않고 다만 신경망 전체 상태에 의해 결정되어진다. 이것은 신경회로망에 전체적으로 퍼져있는 뉴런들에 의해 행해질 수 있는 것이다. 이렇게 퍼져 있는 뉴런들은 잡음이 있는 환경에서 아주 강한 신경망을 만들고 일반화 성능을 개선한다[6].

이러한 연구는 디지털 펄스에서 1과 0의 횟수를 조정함에 의해 표현되어지는 "비트 열(bit stream)"로부터 데이터 표현방법을 이끌어 내었다. 이것은 "확률 계산(stochastic computation)"의 핵심이다[1]. 시간범위에서 정보를 부호화하는 기법들은 "펄스 열 기법(pulse stream techniques)"[10], "확률 로직(stochastic logic)"[5] 그리고 "펄스밀도 조정(pulse-density modulation)"[11]들로 이미 표준 시그모이드 뉴런들이 구현되어 성공적으로 이용되었다. 본 논문에서의 수 표현은 Kondo와 Sawada의 확률 로직[5]으로부터 발전된 형태의 펄스 비를 이용한 방법을 사용할 것이다. 기존의 확률 로직은 단지 기본 논리게이트 하나만으로도 덧셈기나 곱셈기 등을 구현할 수 있었지만, OR게이트로 덧셈을 할 경우 덧셈의 입력 값들이 작은 경우에만 정확한 결과를 낸다는 단점을 갖고 있다. 하지만 여기서 구현될 펄스 비를 이용한 방법은 선형 덧셈을 구현할 수 있다는 것과 수의 표현범위가 넓다는 장점들을 갖고 있다.

최근에는 응용 분야에서 단순한 활용을 위한 정확성 뿐만이 아니라 학습하는 시간도 단축하기 위해 학습기능까지 내장한 on-chip 알고리즘에 대한 연구가 활발하다. 본 논문은 이러한 연구흐름에 따라 확률펄스 비를 이용한 on-chip학습 신경회로망에 대하여 기술하였다. 우선 확률 펄스 연산과 이들 연산을 수행하기 위한 기본 연산기들에 대해 살펴본 후, 네트워크의 실시간 학습이 가능하도록 on-chip학습 알고리즘을 다층 퍼셉트론의 대표적인 모델인 역전파 학습 알고리즘을 이용하여 유도하였다. 이렇게 얻어진 학습 알고리즘을 이용해 신경회로망을 네트워크로 구현하여 Widrow가 제안했던 문자 비선형 분리문제를 통해 검증하였다. 또한 검증된 네트워크를 활자체와 필기체의 십진수 숫자 인식에 적용시켜 성능을 알아보았다.

2. 확률펄스연산

확률연산에서는 기본적으로 확률 펄스열이 필요하다. 이는 확률연산시에 쓰이는 수가 우리들이 일반적으로 사용하는 실수가 아닌, 확률적으로 독립된 수들로써 연산이 이루어지므로 확률 펄스열의 생성이 필요하다. 그림 1에서 확률 펄스열을 생성하는 회로를 간단히 개념적으로 나타내었다.

표현하고자 하는 수를 레지스터에 저장된 값 X라 하자. 0과 X_{max}사이의 난수를 발생시키는 난수발생기(Pseudo random number generator)의 출력 X_i와 레지스터에 저장되어있는 X를 비교하여 발생된 난수가 저장된 값 보다 작으면 비교기의 출력으로 'high'를 내보내고, 난수가 더 크면 'low'를 출력한다. 이러한 방법으로 발생하는 비교기의 출력 펄스열을 L구간 동안 관측하면 이 펄스열이 1의 값을 가질 확률의 기대값 $E[X]$는 X/X_{max}이다. X_{max}가 1인 경우, 이 펄스열의 기대값은 X이고 분산은 $X(1-X)/L$이 됨을 알 수 있다.

이렇게 만들어진 펄스열의 수 표현 방법에는 단극성 방법(unipolar method), 양극성 방법(bipolar method), 펄스 비를 이용한 방법(ratio pulse method)의 3가지로 나누어질 수 있다. 단극성 방법은 수를 펄스가 'high'인 상태에 있을 확률로서 나타내는 것으로, 나타낼 수 있는 수의 표현 범위는 0에서 1이다. 양극성 방법은 단극성 방법에서의 'low'인 상태를 '0'대신에 '-1'로 대응시켜 나타낸 것으로 -1에서 1까지의 수를 나타낼 수 있다. 본 논문에서 이용할 펄스 비를 이용한 방법은 'high'가 나타날 확률과 'low'가 나타날 확률의 비로써 수를 표현하는 것으로 0에서부터 양의 무한대까지의 수를 나타낼 수 있다[5,9]. 단극성 방법에 의해 표현된 수를 X_u라 했을 때, 양극성 방법의 수 X_b와 펄스 비를 이용한 방법의 수 X_r는 각각 다음과 같이 표현될 수 있다.

$$X_b = 2X_u - 1 \tag{1}$$

$$X_r = \frac{X_u}{1-X_u} \tag{2}$$

현재까지 연구되어온 대부분의 확률연산을 이용한 신경회로망들은 단극성 또는 양극성 수 표현 방법을 주로 채택하여 왔는데, 그 이유는 연산기들이 아주 간단하게 구현될 수 있기 때문이었다. 반면 펄스 비를 이용한 방법은 앞서 말한 두 방법보다 개선된 설계방법으로서 최근 많은 연구가 진행되고 있다. 본 절에서는 펄스 비를 이용한 확률연산에 대해 알아보고 이를

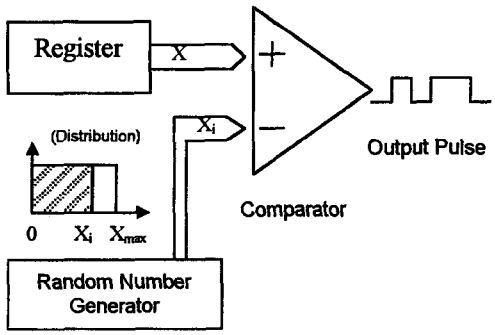

그림 1. 난수 펄스열을 발생시키는 방법
Fig. 1. Method of generating random pulse sequence

신경회로망에 적용하기 위한 기본 연산기들의 회로구현 방법에 대해 살펴보겠다.

2.1 펄스 비를 이용한 확률연산 방법

길이가 L인 펄스열에서 'high'가 나타날 확률과 'low'가 나타날 확률의 비를 이용한 방법으로, 표현된 수를 X라 하고 펄스열에서 펄스가 'high'가 될 확률을 P라고 하면 둘 사이의 관계는 다음과 같다.

$$X = \frac{P}{1-P} \quad (0 \le P \le 1,\ 0 \le X \le \infty) \tag{3}$$

펄스열이 나타내는 수 P가 가지는 기대값(expectation)과 오차의 분산(variation)은 다음과 같다.

$$E[P] = P \tag{4}$$

$$Var[P] = \sigma_P^2 = \frac{P(1-P)}{L} \tag{5}$$

펄스 비를 이용한 확률연산에서 표현하는 수 X는 식 (4)과 식 (5)로부터 다음과 같은 분포를 가지게 됨을 알 수 있다.

$$E[X] \approx f(P) + \frac{\partial^2 (P)\sigma_P^2}{\partial P^2}\frac{\sigma_P^2}{2} = \frac{P}{1-P} + \frac{\sigma_P^2}{(1-P)^3}$$

$$= X + \frac{(1+X)X}{L} \tag{6}$$

$$Var[X] \approx \left|\frac{\partial f(P)}{\partial P}\right|^2 \sigma_P^2 = \frac{P}{(1-P)^3 N} = \frac{X(1+X)^2}{L} \tag{7}$$

식 (6)에서 $L \gg 1$인 경우 $E[X] \cong X$로 근사될 수 있다. 식 (7)로부터 펄스열에서 펄스 수의 비를 이용한 확률연산에서는 표현하는 수 X가 커질수록 오차가 커지게 됨을 알 수 있다. 이 수 표현방법을 이용하면 AND와 OR같은 기본 논리 게이트들과 플립플롭같은 지연 소자만으로도 부동소수점 곱셈과 덧셈, 그리고 나눗셈을 하는 연산자들을 구현할 수 있다. 일반적으로 사용되는 디지털 연산자들에 비하면 회로의 규모를 줄일 수 있는 대단히 매력적인 이점이 아닐 수 없다.

펄스 비를 이용하는 방법이 기존의 확률연산방법에서 주로 사용하는 단극성 표현방법이나 양극성 표현방법과 비교하여 장점이라고 할 수 있는 것은 0에서부터 무한대까지의 수를 나타낼 수 있어 수의 표현범위가 넓은 것 이외에도 단극성 표현방법에서는 하지 못했던 선형덧셈을 구현할 수 있다는 것이다. 이에 비해 단점으로 지적될 수 있는 것은 식 (7)에서 알 수 있듯이 표현하는 수가 커질수록 오차가 커진다는 점이다[9]. 즉 기존의 방법에 비해 수의 범위에 대한 제한이 없어지는 대신 절대값의 크기가 큰 수를 표현하는데에 있어서는 오차가 커지므로 이러한 영향을 고려해야 한다. 하지만 펄스 비를 이용하여 신경회로망을 구현할 때에는 이러한 약점이 크게 영향을 끼치지 않는다. 뉴런의 중간 연산과정에서 덧셈기의 결과나 나눗셈기의 결과는 상당히 큰 수까지 넓은 분포를 가지지만, 각 뉴런들의 출력은 시그모이드 함수를 거치면서 0에서 1사이의 값으로 변환되고, 대부분의 가중치들 또한 0을 중심으로 분포하기 때문에 오차 특성이 신경회로망의 연산에 크게 영향을 주지 않을 것임을 예상할 수 있다.

2.2 신경회로망의 기본 연산기 구현

2.2.1 곱셈기의 구현

일반적인 곱셈기 회로와 달리 확률연산을 이용한 방법, 특히 펄스 비를 이용한 수의 표현 방법에서 입력이 n개인 곱셈기 회로는 그림 2와 같이 구현된다. 곱셈기의 i번째 입력을 X_i, 펄스열에서 펄스가 'high'일 확률을 P_i라고 하면 $P_i = X_i/(1+X_i)$이고, 피드백이 있기 전의 연산결과를 g라고 하면 덧셈기의 출력 s는 $s = g/(1-g)$로부터 다음과 같이 표현된다.

$$g = g\left(1 - \prod_i (1-P_i)\right) + (1-g)\prod_i P_i \tag{8}$$

$$s = \frac{g}{1-g} = \frac{\prod_i P_i}{\prod_i (1-P_i)} = \prod_i \frac{P_i}{1-P_i} = \prod_i X_i \tag{9}$$

식 (9)의 결과를 보면 곱셈이 되는 것을 쉽게 알 수 있다.

2.2.2 덧셈기의 구현

그림 3에 펄스 비를 이용한 덧셈기 회로가 나타나

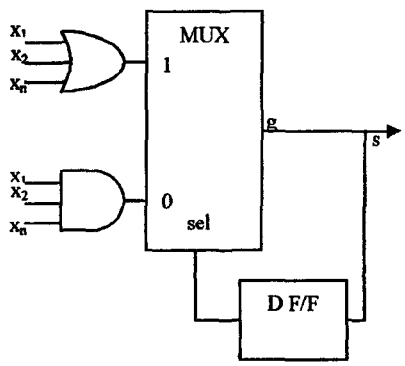

그림 2. 곱셈기 회로
Fig. 2. A multiplier circuit

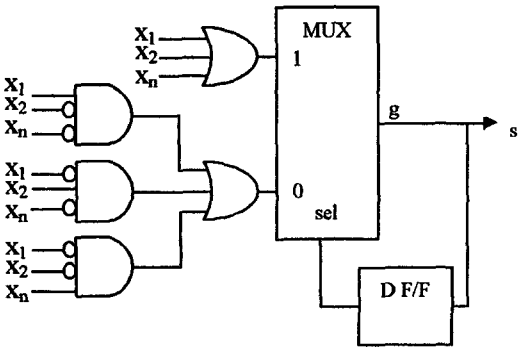

그림 3. 덧셈기 회로
Fig. 3. An Adder circuit

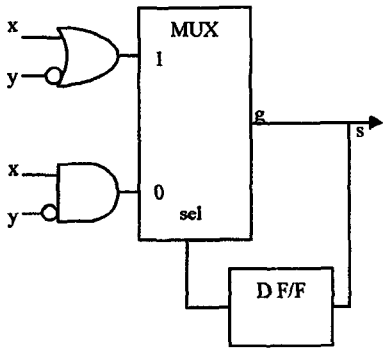

그림 4. 나눗셈 회로
Fig. 4. A divider circuit

있다. 입력이 n개인 덧셈기 회로의 구현방법을 나타낸 것으로 곱셈기의 경우와 마찬가지로 다음과 같이 표현된다.

$$g = g\left(1-\prod_i (1-P_i)\right)+(1-g)\left\{\sum_{i=1}^{n}\left(P_i \prod_{j\neq 1}(1-P_j)\right)\right\} \tag{10}$$

$$s = \frac{g}{1-g} = \frac{\sum_i^n \left[P_j \prod_{j\neq 1}(1-P_i)\right]}{\prod_i (1-P_i)} = \sum_i^n \frac{P_i}{1-P_i} = \sum_i^n X_i \tag{11}$$

2.2.3 나눗셈기의 구현

나눗셈기의 구현은 아주 간단하다. 펄스 비를 이용한 방법에서 표현하려는 수가 인버터(inverter)를 거치면 그 수의 역수(reciprocal)가 되기 때문이다. 그러므로 입력이 두개인 곱셈기에서 분모로 할 입력에 인버터를 달아주면 된다. 그림 4에 회로가 구현되어 있다.

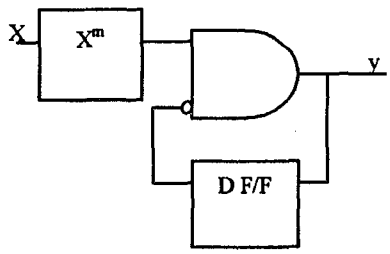

그림 5. 시그모이드 회로
Fig. 5. A sigmoidal function circuit

표 1. m값에 따른 학습률
Table 1. Learning Rate by m

M	4	5	6	7	8
학습률(%)	96.79	97.87	99.96	98.93	99.36

$$s = \frac{x}{y} \tag{12}$$

2.2.4 시그모이드 함수의 구현

역전파 네트워크에는 활성화 함수로 시그모이드 함수를 이용하기 때문에, 이와 동일한 출력 성질을 갖는 회로를 구현하였다(그림 5). 이 회로의 입력으로 들어오는 수를 X라고 할 때, 곱셈기를 거쳐서 X^m으로 만들어 피드백이 있는 AND 게이트를 통과 시켜주면 시그모이드와 같은 0과 1사이의 출력을 내는 출력함수를 얻을 수 있다. 식은 다음과 같이 유도된다.

$$\frac{X^m}{1+X^m}\left(1-\frac{Y}{1+Y}\right) = \frac{Y}{1+Y} \tag{13}$$

$$Y = \frac{X^m}{1+X^m} \tag{14}$$

여기서 지수 m값은 그래프의 기울기를 나타내는데 이에 따른 학습률을 아래의 표 1에서 알아보았다. 이때 학습시킨 데이터는 XOR문제였다. 그래프를 보면 알 수 있듯이 m이 6일때 가장 좋은 특성을 나타내었다.

이와 같은 확률연산을 이용한 계산이 이루어질 수 있는 것은 기본적으로 연산기의 입력으로 들어오는 값들이 서로 확률적으로 독립이어야 한다는 가정하에 만들어졌기 때문이다. 따라서 이러한 확률연산에 이용되는 수를 만들어 내는데 필요한 난수발생기가 확률 연산기의 구현에 있어 매우 중요한 요소중의 하나가 된다. 그러므로 이에 대한 연구도 병행되어야 하고 현재 많은 연구들이 진행중이다[12-14]. 하지만 본 연구에서는 이상적으로 동작하는 확률 펄스 발생

기를 가정하여 on-chip동작을 실현시킬 수 있는 학습 알고리즘의 구현을 목표로 한다는 가정하에 이루어 졌다.

2.2.5 뺄셈기의 구현

뺄셈기는 역전파시에 목표값과 출력 뉴런사이의 차를 구할 때 필요한 연산기로 그림 6과 같이 Eisman이 제안한 monus함수[8]를 이용하여 구현될 수 있다. monus함수의 특성은 다음 식 (15)와 같다.

$$monus(1,x) = \begin{cases} 1-x, & x<1 \\ 0, & otherwise \end{cases} \tag{15}$$

여기서 주목할 점은 뺄셈기의 출력이 절대 값을 갖는다는 것이다. 즉, 부호가 없다는 뜻이다. 따라서 하드웨어 구현시에는 가중치에 부호 비트를 덧붙이면 된다.

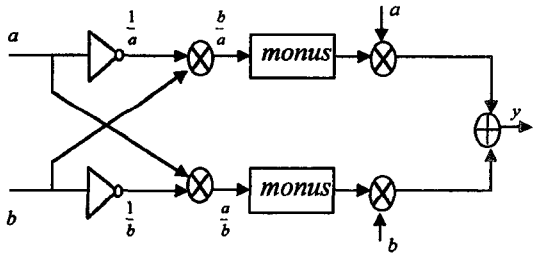

그림 6. 뺄셈기 회로
Fig. 6. A subtractor circuit

3. 확률 펄스연산을 이용한 학습 알고리즘의 구현

앞에서 알아본 역전파 신경회로망을 확률연산기로 구현하기 위해, 본 절에서는 은닉층과 출력층의 구성요소인 뉴런 모델을 선보일 것이다. 그리고 보조 컴퓨터의 도움없이 학습할 수 있는 기능을 내장한 on-chip 알고리즘을 구현하기 위한 펄스 비를 이용한 역전파 학습 알고리즘을 유도한다.

3.1 뉴런 모델

앞장에서 기술하였듯이 펄스 비를 이용한 수의 표현 방법에서는 양의 수밖에 표현할 수 없다. 따라서 다음과 같은 새로운 뉴런 모델이 필요하게 된다. 이것은 생물학적인 뉴런과 마찬가지로 흥분성과 억제성의 시냅스를 가지고 있다. 그림 7이 제안된 뉴런이고 일련의 식들은 이를 수학적으로 나타낸 것이다.

$$net^{+} = \sum_i w_{ji}^{+} x_i \tag{16}$$

$$net^{-} = \sum_i w_{ji}^{-} x_i \tag{17}$$

$$Y = \frac{net^{+}}{net^{-}} \tag{18}$$

$$o = f(Y) = \frac{Y^m}{1+Y^m} \tag{19}$$

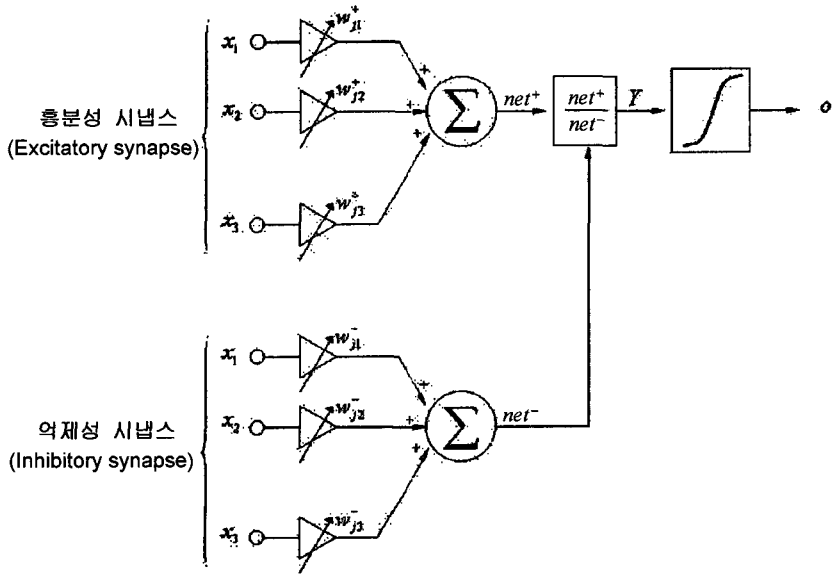

그림 7. 기본 뉴런모델
Fig. 7. The unit neuron model

입력 x_i는 뉴런의 전단계 층의 출력을 나타내고 이 입력들은 각각 가중치 W와 곱셈기를 통해 곱해지면서 가중치의 부호에 따라 흥분성인(excitatory) 것들과 억제성인(inhibitory) 것들로 따로 더해져서 각각 net^+와 net^-, 두 개의 덧셈기의 출력이 나온다. 나눗셈기는 두 덧셈의 결과를 나누어 시그모이드의 입력으로 넣어준다. 여기서 부호의 처리를 위해 흥분성과 억제성 뉴런을 따로 분리해 처리하게 된다. 이때 흥분성과 억제성의 분류 기준은 연결강도의 부호로부터 결정되는데 실제로 회로 내에서는 양의 수로 밖에 이루어져 있지 않다.

3.2 펄스 비를 이용한 on-chip 학습 알고리즘

확률연산을 이용한 신경회로망 구현에는 일반 신경회로망 구현과는 달리 수의 표현방법에 맞는 적절한 학습 알고리즘이 필요하다. 먼저 출력층의 오차의 제곱합 E를 정의하면 다음 식 (20)과 같다. p는 입력 패턴에 대한 index이고, k는 출력층의 각 뉴런들에 대한 index이다.

$$E_p = \frac{1}{2}\sum_k (t_{pk}-o_{pk})^2 \tag{20}$$

$$\frac{\partial E_p}{\partial w_{kj}} = -(t_{pk}-o_{pk})\frac{\partial f}{\partial net_{pk}}\frac{\partial net_{pk}}{\partial w_{kj}} \tag{21}$$

t_{pk}는 입력 패턴 p의 k번째 목표값이고, o_{pk}은 출력층의 k번째 출력값을 나타낸다. 각 연결강도는 오차의 제곱합을 음의 경사(negative gradient)방향으로 감소시키도록 조정된다. 이 식들을 이용하여 출력층의 오차(δ^+_{pk}, δ^-_{pk})를 구하면 다음과 같다.

$$\delta^+_{pk} = (t_{pk}-o_{pk})\frac{\partial o_{pk}}{\partial Y_{pk}} \tag{22}$$

$$\delta^-_{pk} = (t_{pk}-o_{pk})\frac{\partial o_{pk}}{\partial Y_{pk}} \tag{23}$$

뉴런 모델에서 설명하였듯이 가중치는 w^+와 w^- 두 가지로 나뉘어 있다. 따라서 다음과 같이 가중치의 변화량 역시 각기 Δw^+_{kj}와 Δw^-_{kj}로 있어야 한다. 이때 가중치의 변화는 현재 가중치의 부호와 같은 변화량만이 변화하게 된다. 즉 양의 가중치였다면 그것만이 변화량을 갖고, 음의 가중치였다면 양의 가중치가 아닌 음의 가중치만이 변화를 한다.

$$\Delta w^+_{kj} = \eta\delta_{pk}\frac{1}{net^-_{pk}}i_{pj} = \eta\delta_{pk}Y_{pk}\frac{1}{net^+_{pk}}i_{pj} \tag{24}$$

$$\Delta w^-_{kj} = \eta\delta_{pk}Y_{pk}\frac{1}{net^-_{pk}}i_{pj} \tag{25}$$

여기서 i_{pj}는 은닉층의 j번째 뉴런의 출력값이다. η은 뉴런의 학습률로써 상수이다. 다음은 같은 방법으로 유도된 은닉층의 학습 알고리즘이다.

$$E_p = \frac{1}{2}\sum_k (t_{pk}-o_{pk})^2 = \frac{1}{2}\sum_k (t_{pk}-f_k(\sum w_{kj}i_{pj}))^2 \tag{26}$$

$$\frac{\partial E_p}{\partial w_{ji}} = \frac{1}{2}\sum_k \frac{\partial}{\partial w_{ji}}(t_{pk}-o_{pk})^2 = -\sum_k\left[(t_{pk}-o_{pk})\times\frac{\partial o_{pk}}{\partial net_{pk}}\frac{\partial net_{pk}}{\partial i_{pj}}\frac{\partial i_{pj}}{\partial net_{pj}}\frac{\partial net_{pj}}{\partial w_{ji}}\right] \tag{27}$$

은닉층의 오차 역시 두 경우로 분류하여 다음과 같이 각각 계산한다.

$$\delta^+_{pj} = \frac{\partial o_{pj}}{\partial Y_{pj}}\sum_k \delta^+_{pk}Y_{pk}\left(\frac{1}{net^+_{pk}}w^+_{kj}+\frac{1}{net^-_{pk}}w^-_{kj}\right) \tag{28}$$

$$\delta^-_{pj} = \frac{\partial o_{pj}}{\partial Y_{pj}}\sum_k \delta^-_{pk}Y_{pk}\left(\frac{1}{net^+_{pk}}w^+_{kj}+\frac{1}{net^-_{pk}}w^-_{kj}\right) \tag{29}$$

식 (28)과 (29)의 오차를 이용한 은닉층의 가중치 변화량 역시와로 나뉘어져 있으며 아래에 각각의 변화량을 기술하였다.

$$\Delta w^+_{ji} = \eta\delta_{pj}\frac{1}{net^-_{pj}}x_{pi} = \eta\delta_{pj}Y_{pj}\frac{1}{net^+_{pi}}x_{pi} \tag{30}$$

$$\Delta w^-_{ji} = \eta\delta_{pj}Y_{pj}\frac{1}{net^-_{pj}}x_{pi} \tag{31}$$

이렇게 구해진 알고리즘을 이용해 학습기능을 내장한 뉴런의 구조도를 그림 8에 나타냈다. 그림은 두개의 시냅스와 1개의 뉴런으로 이루어져 있다. 이는 하드웨어 구현시에 방향을 제시해줄 것이다.

4. 모의실험 및 결과

위와 같이 유도된 on-chip 학습 알고리즘을 컴퓨터 모의실험을 통해 검증하였다. 모의 실험은 모두 세 가지로 한 개의 비선형 패턴 분류문제와 두 개의 십진수 인식문제에 대하여 실행하였다. 우선 Widrow가 제안한 까다로운 비선형 패턴 분리 문제[6]를 실험하여 알고리즘을 검증한 후, 간단한 활자체 숫자인식을 한다. 마지막으로 필기체 숫자 인식문제를 통해 다양한

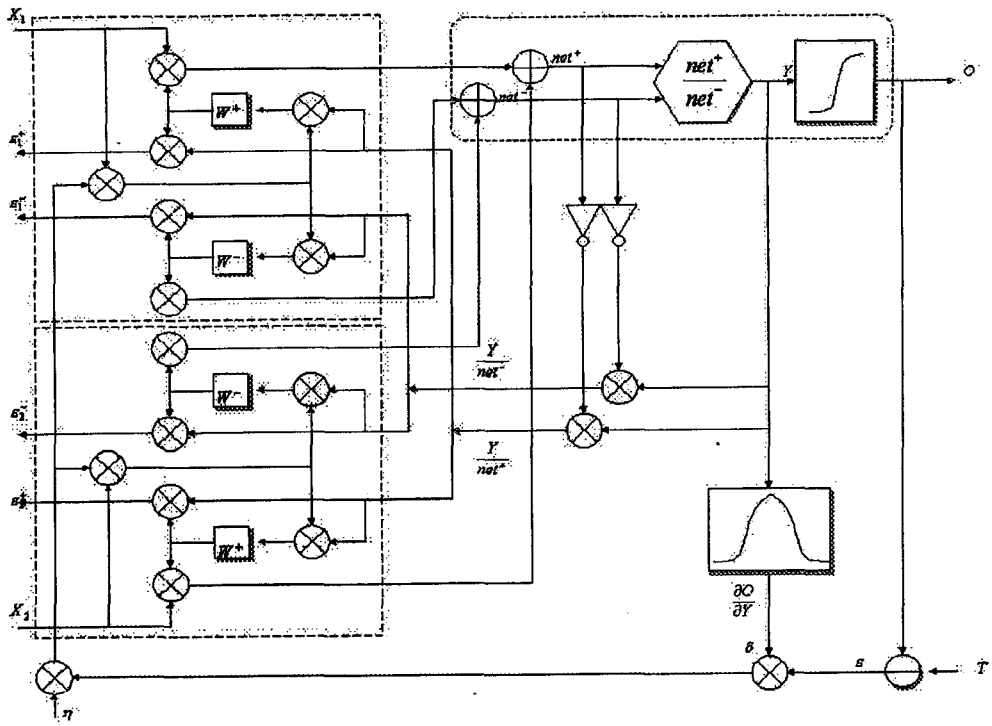

그림 8. 학습기능을 내장한 뉴런의 구조도
Fig. 8. An architecture of a neuron with on-chip learning

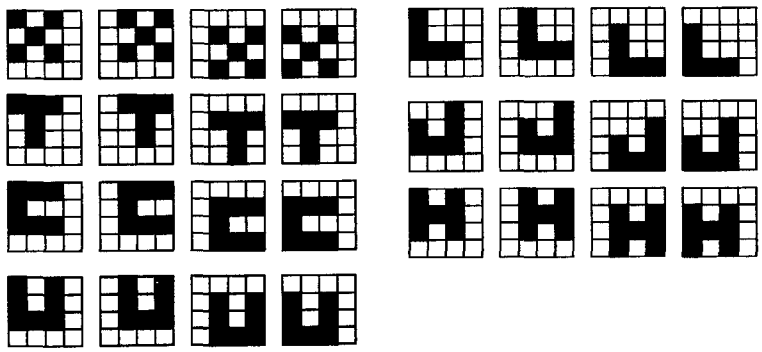

그림 9. Widrow의 학습패턴
Fig. 9. Widrow's patterns for training

응용분야에 적용될 수 있는지를 알아본다. 참고로 본 모의실험에서 사용된 네트워크는 모두 한 개의 은닉층을 갖는 것으로 하였다.

4.1 비선형 패턴 분류

본 실험에서 사용된 데이터는 모두 28개의 영문자로 Widrow가 제안한 것으로 하나의 문자는 4×4의 16개 픽셀로 되어 있다. 그림 9의 픽셀값은 1(흑색) 또는 0(백색)의 값을 가진다. 영문자 *X*, *T*, *C*, *U*를 한 그룹으로 분류하고, 나머지 *L*, *J*, *H*를 다른 그룹으로 분류하는 문제로써, 이 문제는 매우 심한 비선형 분리문제로 알려져 있다.

이번 실험에서 입력층은 16개의 노드로 구성하였으며, 은닉층에는 10개의 뉴런, 출력층에는 1개의 뉴런으로 각각 구성하였다. 학습률은 0.08로 하였다. 이것 역시 학습된 네트워크는 학습에 사용된 원래의 패턴

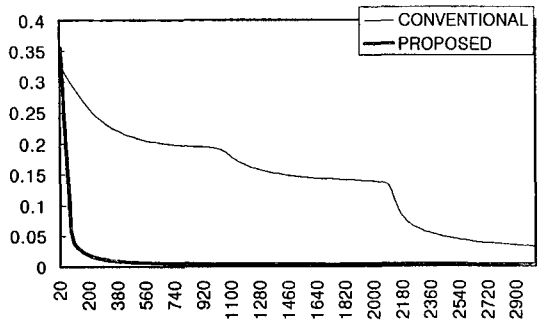

그림 10. 학습시 평균제곱오차
Fig. 10. Mean Square Error

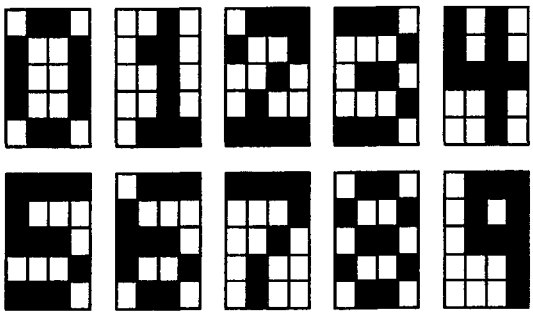

그림 11. 십진수 활자패턴
Fig. 11. Printed decimal patterns

들에 한 픽셀의 잡음을 첨가하여 분류 실험을 하였다. 실험결과는 그림 10와 같으며, 기존의 역전파 네트워크보다 훨씬 빠르게 학습되는 것을 알 수 있고, 정확도 역시 매우 좋은 것을 볼 수 있다.

4.2 활자체 인식

십진수의 활자체를 4×5의 픽셀로 입력을 받는다. 따라서 입력층의 노드수는 20개이고, 은닉층의 뉴런은 22개로 하였다. 그리고 출력층은 4개의 뉴런으로 구성하였다. 본 실험에서의 학습률은 0.4로 하였다. 학습은 그림 11과 같은 0에서 9까지의 열개의 패턴으로 이루어졌으며, 학습종료후 실제 적용시에는 각 패턴에 1 비트의 노이즈를 랜덤하게 첨가하여 실시하였다. 이에 대한 결과는 표 2에 나타나 있다.

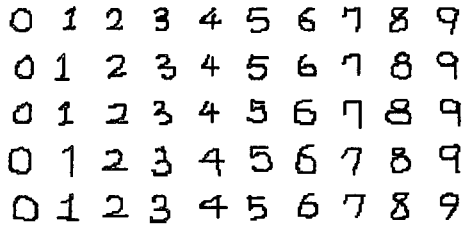

그림 12. 필기체 데이터 샘플
Fig. 12. Some samples of the handwritten data

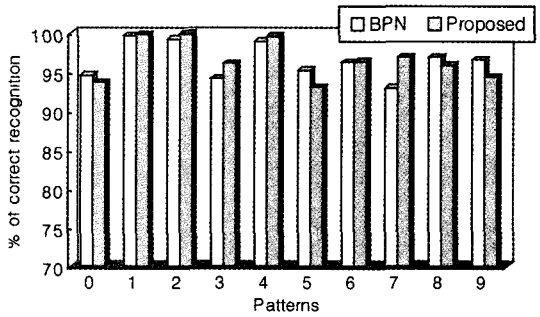

그림 13. 숫자 인식률
Fig. 13. Recognition rate

결과에서 보면 알 수 있듯이 기존의 역전파 네트워크보다 인식능력이 향상됨을 알 수 있으며, 수치적으로 0.13%정도의 향상을 보이고 있다.

4.3 필기체 인식

이번에는 제안된 알고리즘을 이용한 네트워크를 10진수 필기체에 적용시켜 보겠다. 입력은 PC의 마우스를 이용해 0부터 9까지의 숫자를 입력 받는다. 이렇게 입력 받은 데이터를 제안된 네트워크에 사용하기 전에 간단한 화상처리를 통해 16×16픽셀의 0과 1로 이루어진 데이터로 변환한다. 필기체 데이터는 11명으로부터 모두 110개를 받았다. 여기서 80개를 학습패턴으로 사용하였고 나머지 30개는 데이터로 적용을 하여 인식률을 알아보았다. 네트워크의 학습률은 0.4로 하였다.

그림 12이 입력 받은 필기체 데이터의 예제이고,

표 2. 숫자 인식률
Table 2. Recognition rate (comparing with the proposed MLP and a conventional BP network)

Patterns	0	1	2	3	4	5	6	7	8	9
BPN	98.55	99.24	99.12	98.62	99.17	97.54	97.99	98.70	99.15	99.25
Proposed	98.90	99.24	99.21	99.00	99.58	97.00	98.54	98.78	99.22	99.20

학습 종료 후 각 숫자들에 대한 인식률을 그림 13에 나타내었다.

5. 결론 및 향후과제

본 연구에서는 역전파 알고리즘을 이용해 확률펄스비를 이용한 다층 퍼셉트론의 on-chip 학습 알고리즘을 유도하였다. 확률연산을 사용하는 신경회로망은 기존의 디지털 회로에 비해 간단한 회로로 기본 연산기를 구현할 수 있다는 장점이 있다. 또한 역전파 학습 알고리즘을 이용하여 구현한 on-chip 학습 알고리즘은 확률 신경회로망의 특성으로 인해 지역최소값에 빠지는 확률이 기존의 알고리즘에 비해 훨씬 적은 것을 확인할 수 있었다.

제안된 연산기들과 학습 알고리즘을 이용해 다층 퍼셉트론 네트워크를 구성하여 십진수 인식과 문자분류 실험을 통해 그 성능이 기존의 역전파 네트워크 보다 훨씬 향상된 성능을 가짐을 확인할 수 있었다. 앞으로 연구해야 할 과제는 좀더 실제적인 문제에 적용하려는 시도와 오차의 해석에 대한 연구가 진행되어야 할 것으로 본다. 또한 신경회로망의 학습과 일반화 능력에 확률연산의 불확실성이 어떠한 영향을 미치는가에 대한 고찰도 필요하다. 그리고 현재 이렇게 제안된 신경회로망을 집적회로로 구현하는 연구를 진행하고 있다.

감사의 글

본 연구는 대학기초연구 및 뇌의약학 연구과제의 지원을 받아 수행되었습니다.

참고문헌

[1] B. R. Gaines, "Stochastic computing system", Advances in Information Systems Science, New York, 1969.

[2] Hiroomi Hikawa, "Frequency-Based Multilayer Neural Network with On-Chip Learning and Enhanced Neuron Characteristics", *IEEE Trans. On Neural Networks*, vol. 10, no. 3, May 1999.

[3] Dan Hammerstrom, "Digital VLSI for Neural Networks", Part III : Articles, pp. 304-309.

[4] Robert W. Newcomb and Jason D. Lohn, "Analog VLSI for Neural Networks", Part III : Articles, pp. 86-90.

[5] Y. Kondo and Y. Sawada, "Functional Abilities of a Stochastic Logic Neural Network", *IEEE Trans. on Neural Networks*, vol. 3, no. 3, pp. 434-443, 1992.

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation", Parallel Distributed Processing, vol. 1, ch.8, MIT press, Cambridge, MA, 1986.

[7] A. Cichoki, R. Unbehauen, "Neural Networks for Optimization and Signal Processing", John Wiley & Sons, pp. 38-167, 1998.

[8] G. S. Eisman, "Frequency based computation in neural networks", *Proc. Of IJCNN,* vol. 2, pp. 577-582, 1991.

[9] DARPA Neural Network Study Final Report, Lincoln Lab., MIT, Lexington, MA, 1988.

[10] A. Murray, L. Tarassenko, "Analogue neural VLSI – a pulse stream approach", Chapman and Hall, London, 1994.

[11] J. E. Tomberg, K. K. K. Kaski, "Pulse-density modulation technique in VLSI implementations of neural network algorithms", *IEEE J. Solid-State Circuits*, vol. 25, pp. 1277-1286, 1990.

[12] P. D. Hortensius, R. D. McLeod, and H. C. Card, "Parallel Random Number Generator for VLSI Systems Using Cellular Automata", *IEEE Trans. On Computers*, vol. 38, no. 10, pp. 1466-1473, Oct. 1989.

[13] P. D. Hortensius, R. D. McLeod, Werner Pries, D. Michael Miller, and H. C. Card, "Cellular Automata-Based Pseudorandom Number Generators for Built-In Self-Test", *IEEE Trans. On CAD*, vol. 8, no. 8, pp. 842-859, Aug. 1989.

[14] J. Alspector, J. W. Gannett, S. Haber, Michael B. Parker, and Robert Chu, "A VLSI-Efficient Technique for Generating Multiple Uncorrelated Noise Sources and Its Application to Stochastic Neural Networks", *IEEE Trans. On Circuits and Systems*, vol. 38, no. 1, pp. 109-123, Jan. 1991.

[15] H. Eguchi, T. Furuta, H. Horigushi, S. Oteki, and T. Kitabuchi, "Neural Network LSI Chip with On-chip Learning", *Proc. Of IJCNN*, vol. 1, pp. 453-456, 1991.

[16] E. E. Pesulima, A. S. Pandya, and R. Shankar, "digital implementation issues of stochastic neural networks", *Proc. Of IJCNN*, vol. 2, pp. 187-190, 1989.

[17] H. Guo and S. B. Gelfand, "Analysis of Gradient Descent Learning Algorithm for Multilayer Feedforward Neural Networks", *IEEE Trans. On Circuits and Systems*, vol. 38, pp. 883-894, 1991.

[18] Y. Q. Chen, T. Yin and H. A. Babri, "A Stochastic Backpropagation Algorithm for Training Neural Networks", *Int'l Conf. On Info., Comm. and Signal Processing*, pp. 703-707, 1997.

[19] C. L. Janer, J. M. Quero, J. G. Ortega, and L. G. Franquelo, "Fully Parallel Stochastic Computation Architecture", *IEEE Trans. On Signal Processing,* vol. 44, no. 8, pp. 2110-2117, Aug. 1996.

[20] 서정원, "Stochastic Pulse Coding을 이용한 신경회로망의 구현", 서울대학교 전자공학과 석사학위 논문, 1993.

[21] 민승재, 이일완, 채수익, "펄스열에서 1인 펄스수와 0인 펄스수의 비를 이용한 확률연산에 관한 연구", 제 3회 인공지능, 신경망 및 퍼지 시스템 종합 학술대회/전시회 논문집, pp. 489-493, 1993.

김 응 수 (Eung-Soo Kim)

1977년 : 부산대학교 공과대학 전자공학과 졸업
1979년 : 부산대학교 대학원 전자공학과 졸업(석사)
1993년 : 日本 東北大學校 공학연구과 전자공학과 졸업(박사)
1986년~1987년 : 日本 Waseda University 객원연구원
1984년~1994년 : 한국전자통신연구원 책임연구원
1994년~2000년 : 선문대학교 전자정보통신공학부 교수
2000년~현재 : 대전대학교 공과대학 전자공학과 교수
관심분야 : 신경회로망, 브레인 컴퓨팅, 카오스와 프랙탈 이론 Neural Network and Brain Computing, Statistical Physics of Neural Network, Chaos and Fractal theory in Neural Network, Modeling biological function with Neural Net., Learning and Generalization Algorithm

박 태 진 (Tae-Jin Park)

1999년 2월 : 선문대학교 공과대학 전자정보통신공학부 졸업(학사)
1999년 현재 : 선문대학교 대학원 전자공학과
관심분야 : 신경회로망, 하드웨어 구현, 비선형 동역학, 패턴 인식

조 덕 연 (Duk-Yun Cho)

2000년 2월 : 선문대학교 공과대학 전자정보통신공학부 졸업(학사)
2000년~현재 : 선문대학교 대학원 기계제어공학부
관심분야 : 신경회로망, 비선형 동역학, 패턴인식, 정보이론 Embedded Systems, Real-Time Systems