論文94-31B-7-23

펄스열에서 1인 펄스수와 0인 펄스수의 비를 이용하여 확률연산을 하는 신경회로망

(A Neural Network Based on Stochastic Computation Using the Ratio of the Number of Ones and Zeros in the Pulse Stream)

閔 勝 載, 蔡 洙 翊,

(Seung Jai Min and Soo-Ik Chae)

要 約

확률연산에서는 수를 표현하기 위해서 펄스열을 사용한다. 본 논문에서는 펄스열에서 0인 펄스수와 1인 펄스수의 비를 이용한 수 표현방법을 구현하기 위한 새로운 방법을 제안한다. 펄스열에서 펄스가 1이 될 확률을 P라고 하면 이 수 표현방법에 의한 수 X의 기대값은 P/(1-P)로 표현된다. 본 논문에서는 이러한 수 표현방법을 이용하여 덧셈, 곱셈, 그리고 시그모이드 함수와 같은 기본 연산의 회로구현 방법을 제안하고 확률연산에서의 이러한 기본연산들의 오차 특성을 연구한다. 본 논문에서는 뉴런(neuron) 모델을 제안하며 순방향 신경회로망을 위한 오차역전파 알고리즘(backpropagation algorithm)을 바탕으로 하여 제안한 뉴런 모델을 위한 학습알고리듬을 유도한다. 그리고 유도한 학습알고리듬을 이용하여 제안한 뉴런 모델을 숫자 인식 문제에 적용한다. 실험 결과를 해석하기 위하여 확률연산에서 펄스열의 길이에 의한 오차와 양자화 오차에 대하여 논한다.

Abstract

Stochastic computation employs random pulse streams to represent numbers. In this paper, we study a new method to implement the number system which uses the ratio of the numbers of ones and zeros in the pulse streams. In this number system, if P is the probability that a pulse is one in a pulse stream, then the number X represented by the pulse stream is defined as P/(1-P). We propose circuits to implement the basic operations such as addition, multiplication, and sigmoid function with this number system and examine the error characteristics of such operations in stochastic computation. We also propose a neuron model and derive a learning algorithm based on backpropagation for the 3-layered feedforward neural networks. We apply this learning algorithm to a digit recognition problem. To analyze the results, we discuss the errors due to the variance of the random pulse streams and the quantization noise of finite length register.

Ⅰ. 서론

신경회로망을 실시간 처리가 요구되는 응용분야에 적용하기 위해서는 신경회로망의 하드웨어 구현이 필

*正會員, 서울大學校 電子工學科
(Dept. of Elec. Eng., Seoul Nat'l Univ.)
接受日字 : 1994年 1月 3日

요하다. 현재 급속히 발전하고 있는 VLSI 기술을 이용하여 신경회로망을 집적화하려는 연구가 활발히 진행되고 있다. 신경회로망 칩의 구현방법에는 아날로그 회로를 이용한 방법과 디지탈 회로를 이용한 방법으로 크게 나눌 수 있다. 이 중 디지탈회로를 이용한 방법은 잡음에 덜 민감하고 잘 정립된 설계 방법론을 이용할 수 있다는 장점이 있는 반면에 신경회로망을 구현하는 데에 있어 기본적으로 단위 시간당 많은 덧셈과 곱셈의 연산이 필요하므로 어려움이 있다. 이러한 문제를 해결하기 위하여 여러가지 방법이 제안이 되고 있는데 그러한 방법중의 하나로서 펄스열을 이용한 확률연산 방법이 있다. 확률연산을 이용한 신경회로망은 디지탈 방법의 장점을 유지하면서 적은 면적으로 덧셈기나 곱셈기와 같은 기본연산을 구현할 수 있으므로 많은 수의 덧셈기와 곱셈기를 집적하여 이 문제를 극복할 수 있다.

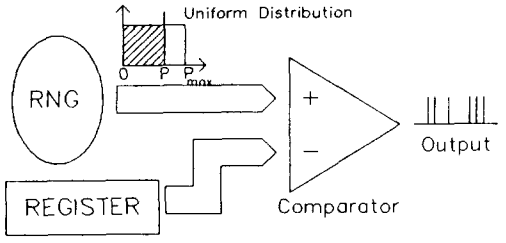


그림 1. 난수 발생기와 비교기를 이용하여 펄스열을 만드는 방법

Fig. 1. Method of generating a random pulse steam with a comparator and a pseudo-random number generator.

그림 1은 확률연산에서 펄스열을 만드는 방법을 나타낸 것으로 입력값 또는 레지스터(register)에 저장된 가중치(weight)가 나타내는 값을 P라고 할때 0과 P_{max}사이의 난수를 발생시키는 난수발생기(Random Number Generator)의 출력과 비교를 하여 발생된 난수가 P보다 작을 경우 1인 펄스를 비교기의 출력으로 내보내고 P보다 크면 0인 펄스를 내보낸다. 이러한 방법으로 발생하는 비교기의 출력 펄스열을 L 구간 동안 관측하면 이 펄스열이 1의 값을 가질 확률의 기대값 $E[P]$ 는 P/P_{max}이고 P_{max}가 1인 경우, 이 펄스열의 분산은 $P(1-P)/L$이 됨을 알 수 있다. 확률연산을 이용한 신경회로망에는 수를 표현하는 방법에 따라 세가지의 종류로 나누어 질 수 있다. 한가지는 기존의 대부분의 확률연산이 이용한 방법인 단극성(Unipolar) 표현 방법이다. [1] 이 표현 방법은 덧셈과 곱셈이 각각 AND 게이트와 OR 게이트로 하나로 구현이 된다는 장점이 있으나 OR게이트로 덧셈을 할 경우 덧셈의 입력값들이 작은 경우에만 정확한 덧셈이 이루어진다는 단점이 있고, 전체 펄스열 구간에서 펄스가 1이 될 확률로 수를 표현하므로 수의 표현 범위가 0에서 1까지로 제한이 된다. 두번째 방법은 양극성(Bipolar) 표현방법으로 펄스가 1일 확률에서 0일 확률의 차를 사용하며 이 방법으로는 -1에서 1까지의 수를 표현할 수가 있다. [2] 세번째 방법은 펄스가 1일 확률과 0일 확률의 비를 이용한 방법으로서 0과 양수를 나타낼 수가 있다. [3] 현재까지 연구되어온 대부분의 확률연산을 이용한 신경회로망들은 단극성 또는 양극성 수 표현방법을 주로 채택하여 왔는데 그 이유는 위에서 설명한대로 연산이 간단하게 구현되기 때문이다. 반면에 0과 1의 펄스수의 비를 이용한 방법이 거의 사용되어오지 않은 것은 이러한 수 표현 방법이 새로운 개념으로서 아직까지 이에 대한 해석이 부족하기 때문이다. 본 연구에서는 펄스수의 비를 이용한 확률연산에 대한 해석을 시도하고 이를 신경회로망에 적용하기 위한 기본 연산방법들의 회로구현 방법을 설명한다.

논문의 구성은 Ⅱ장에서 0과 1의 펄스수의 비를 이용한 확률연산으로 덧셈기, 곱셈기와 나눗셈기 그리고 시그모이드 출력 함수를 구현하는 방법에 대하여 설명한다. Ⅲ장에서는 Ⅱ장에서 설명된 방법들을 이용하여 인공신경회로망을 구현하는 방법을 설명하고 Ⅳ장에서는 제안한 신경회로망 구조를 숫자인식문제에 대해 적용한 실험 결과를 제시하고 실험 결과에 대한 오차에 대해 해석하며 Ⅴ장에서는 제안한 방법에 대한 결론을 기술하였다.

Ⅱ. 0과 1의 펄스비를 이용한 확률연산

펄스열에서 1인 펄스수와 0인 펄스수의 비를 이용한 방법으로 표현된 수를 X라 하고 펄스열에서 펄스가 1이 될 확률을 P라고 하면 둘 사이의 관계는 다음과 같다.

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

펄스열에서 펄스가 1이 될 확률 P는 0과 1사이에 있으므로 X는 0에서 무한대까지의 수 표현범위를 가진다. 관측하는 펄스열의 길이가 L일때, 확률연산에서 펄스열이 나타내는 수 P가 가지는 기대값과 오차의 분산은 다음과 같다.

$$E[P] = P, \quad Var[P] = \sigma_p^2 = \frac{P(1-P)}{L} \tag{2}$$

펄스수의 비를 이용한 확률연산에서 표현하는 수 X는 식 (1)과 (2), 그리고 $f(P)=P/(1-P)$의 관계식으로부터 다음과 같은 분포를 가지게 됨을 알 수 있다.

$$E[X] \cong f(P)+\frac{\partial f^2(P)}{\partial P^2}\frac{\sigma_p^2}{2}=\frac{P}{1-P}+\frac{\sigma_p^2}{(1-P)^3}=X+\frac{(1+X)X}{L} \quad (3)$$

$$Var[X] \cong \left|\frac{\partial f(P)}{\partial P}\right|^2 \sigma_p^2=\frac{P}{(1-P)^3 N}=\frac{X(1+X)^2}{L} \quad (4)$$

식 (3)에서 $L>>1$ 인 경우에는 $E[X] \simeq X$ 로 표현할 수 있다. 식 (4)로부터 펄스열에서 0과 1의 수의 비를 이용한 확률연산에서는 표현하는 수 X가 커질수록 오차가 커지게 됨을 알 수 있다. 이 수 표현방법을 이용하면 AND, OR와 같은 간단한 논리 소자들과 플립플롭(f/f)같은 지연 소자만으로도 부동소수점 곱셈과 덧셈, 그리고 나눗셈을 구현할 수 있다.

1. 덧셈기의 구현

덧셈기에 있어서 단극성 수 표현방법을 사용하는 기존의 확률연산은 OR 게이트를 사용하여 덧셈을 수행하기 때문에 덧셈의 결과가 비선형적인 형태를 나타낸다. 0과 1의 비를 이용한 확률연산에서는 그림 2의 회로를 이용하여 선형 덧셈을 구현한다.

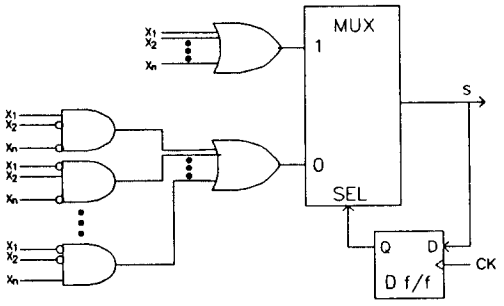


그림 2. 입력이 n개인 덧셈기의 회로
Fig. 2. An adder circuit with n inputs.

그림 2에서 덧셈기의 i-번째 입력을 X_i, 펄스열에서 펄스가 1이 될 확률을 P_i라고 하면 $P_i=X_i/(1+X_i)$이고, 연산결과를 g라고 하면 덧셈기의 출력 s는 $s=g/(1-g)$로 부터 다음과 같이 표현된다.

$$g=g[1-\prod_i(1-P_i)]+(1-g)[\sum_{i=1}^{n}(P_i\prod_{j\neq i}(1-P_j))] \quad (5)$$

$$s=\frac{g}{1-g}=\frac{\sum_i^n[P_i\prod_{j\neq i}(1-P_j)]}{\prod_i(1-P_j)}=\sum_i^n\frac{P_i}{1-P_i}=\sum_i^n X_i \quad (6)$$

이러한 덧셈기 회로를 이용한 덧셈의 기대값과 오차의 분산은 덧셈기 입력으로 들어오는 펄스열들이 서로 독립적인 경우, 다음과 같이 구할 수 있다.

$$E[s]=\eta_s=\sum_i\eta_i=\sum_i X_i,\ Var[s]=\sigma_s^2=\sum_i\sigma_i^2=\frac{1}{L}\sum_i X_i \quad (7)$$

2. 곱셈기의 구현

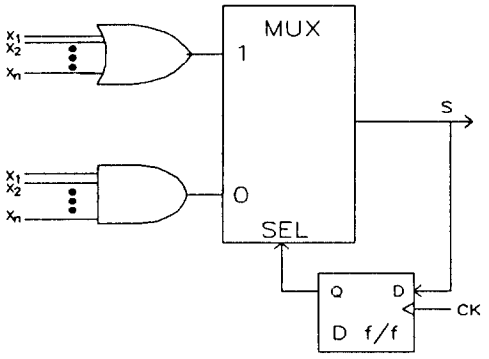


그림 3. 곱셈기의 회로
Fig. 3. A multiplier circuit with n inputs.

그림 3은 입력이 n개인 곱셈기의 회로 구현 방법을 나타낸 것으로 덧셈기의 경우와 마찬가지로,

$$g=g(1-\prod_i(1-P_i))+(1-g)\prod_i P_i \quad (8)$$

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

본 논문에서 사용하는 곱셈기는 입력으로 뉴런(neuron)의 입력값과 가중치(weight)를 사용한다. 이때 입력값과 가중치의 펄스열이 서로 독립적이고 각각의 기대값과 분산을 η_x, η_w, σ_x^2 그리고 σ_w^2라고 하면 입력이 두개인 곱셈의 기대값과 분산은 다음과 같다.

$$E[s]=\eta_s=f+\frac{1}{2}\left(\frac{\partial^2 f}{\partial x^2}\sigma_x^2+\frac{\partial^2 f}{\partial w^2}\sigma_w^2\right)=\eta_x\eta_w \quad (10)$$

$$\sigma_s^2=\frac{\partial f^2}{\partial x}\sigma_x^2+\frac{\partial f^2}{\partial w}\sigma_w^2=\eta_w^2\sigma_x^2+\eta_x^2\sigma_w^2=\frac{1}{N}(\eta_w^2 X_i+\eta_x^2 W) \quad (11)$$

3. 나눗셈기의 구현

비를 이용한 수 표현방법에서 펄스열이 인버터(inverter)를 거치면 펄스열이 나타내는 수의 역수

(reciprocal)가 되므로 나눗셈은 입력이 두개인 곱셈기회로에서 하나의 입력을 인버터를 거쳐서 들어가게 하면 쉽게 구현이 된다. 역수를 취한 젯수가 y이고 피젯수가 x일 때, 나눗셈기의 기대값과 분산은 곱셈의 경우과 마찬가지로 구할 수 있다.

$$E[s]=\eta_s=f+\frac{1}{2}\left(\frac{\partial^2 f}{\partial x^2}\sigma_x^2+\frac{\partial^2 f}{\partial y^2}\sigma_y^2\right)=\frac{\eta_x}{\eta_y}+\frac{\eta_x}{\eta_y^4}\sigma_y^2 \tag{12}$$

$$\sigma_s^2=\left(\frac{\partial f}{\partial x}\right)^2\sigma_x^2+\left(\frac{\partial f}{\partial y}\right)^2\sigma_y^2=\frac{1}{\eta_y^2}\sigma_x^2+\frac{\eta_x^2}{\eta_y^4}\sigma_y^2 \tag{13}$$

4. 시그모이드 출력함수의 구현

펄스열에서 1인 펄스수와 0인 펄스수의 비를 이용한 확률연산 방법으로 신경회로망을 구현하기 위해서는 덧셈기, 곱셈기 같은 기본 연산외에도 출력함수(Output function)가 필요하다. 본 논문의 신경회로망에 사용된 출력 함수는 오차역전달(error back-propagation)방법을 이용한 학습알고리듬을 유도하기 위해 시그모이드(Sigmoid) 형태의 출력함수를 사용하였다. 그림 4는 이를 구현하기 위한 것으로 입력으로 들어오는 수를 X라고 하고 이 입력을 곱셈기를 거쳐서 X의 m승으로 만들어 주어 되먹임(Feedback)이 있는 AND 게이트를 거치면 0과 1사이의 출력을 내는 출력함수를 얻을 수 있다. 이를 식으로 나타내면 다음과 같다.

$$\frac{X^m}{1+X^m}*\left(1-\frac{Y}{1+Y}\right)=\frac{Y}{1+Y} \tag{14}$$

$$Y=\frac{X^m}{(1+X^m)} \tag{15}$$

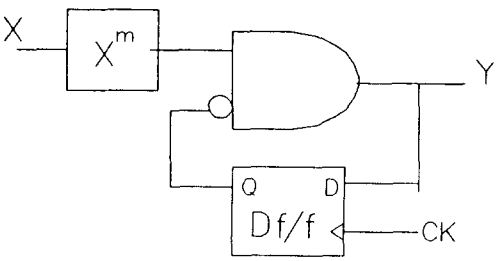


그림 4. 시그모이드 출력 함수의 회로도
Fig. 4. A circuit for the sigmoid function.

그림 5는 m의 값에 따른 시그모이드 함수의 입력에 대한 출력의 기대값을 부동소수점으로 계산한 것이고 그림 6은 펄스열을 이용한 확률연산으로 계산한 것이다. 그림 6에서 사용한 펄스열의 길이는 4096이다. 그림 5와 6에서 볼 수 있듯이 m의 값이 커질수록 시그모이드 출력의 기울기가 급격해짐을 알 수 있다.

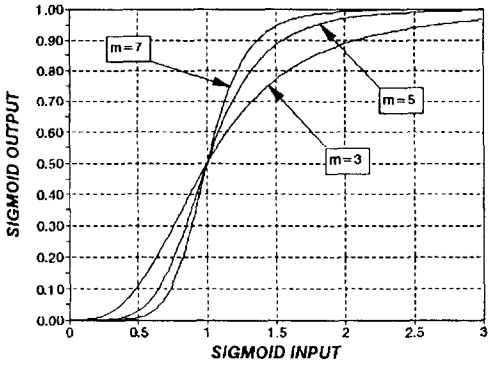


그림 5. 시그모이드 출력함수의 입출력함수의 기대값
Fig. 5. Expectation values of input-output transfer curves for the sigmoid function.

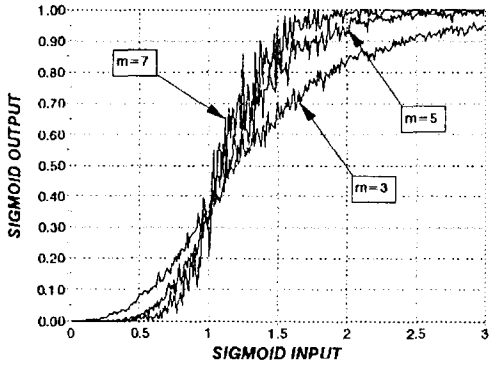


그림 6. 확률연산에 의한 시그모이드 출력함수의 입출력 곡선
Fig. 6. Input-output transfer curves for the sigmoid function in stochastic computation.

Ⅲ. 신경회로망의 구현

제안한 확률연산 방법을 이용하여 입력층, 은닉층, 그리고 출력층으로 구성된 다층 신경회로망을 구현하였다.[7-9] 그림 7은 이를 위해 Ⅱ장에서 설명한 기본 연산들을 구현하는 방법들로 하나의 뉴런 모델(neuron model)을 구성하는 방법을 나타낸 것이다. O_j는 뉴런의 전 단계층(layer)에서 넘어오는 입력들

을 나타내고 이 입력들은 각각 가중치 w와 곱셈기를 통해 곱해지면서 가중치의 부호가 양수이냐 음수이냐에 따라 흥분성인 것들과 억제성인 것들로 따로 더해져서 각각 net$^+$와 net$^-$, 두개의 덧셈기의 출력이 나온다. 나눗셈기는 이 두 덧셈 결과를 결합하여 흥분성 신호(net$^+$)가 억제성 신호(net$^-$)보다 크면 1보다 큰 값을 나타내고 반대의 경우에는 1보다 작은 작은 수를 나눗셈기의 출력으로 나타내어 시그모이드 출력함수의 입력으로 넣어준다. 시그모이드 출력함수는 그림 6에서처럼 0 또는 1에 가까운 값으로 O_k를 결정한다. 이러한 뉴런 모델에 평균기를 적용하는 경우에는 각 뉴런들의 출력함수의 결과를 평균기의 입력으로 넣어준다.

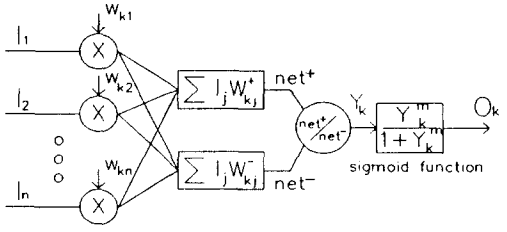


그림 7. 입력이 n개인 뉴런의 수학적 모델
Fig. 7. Mathematical model for a neuron with n inputs.

위의 구조에서 가중치의 부호에 따라 net$^+$와 net$^-$로 나누어 주는 것은 펄스열을 이용한 확률연산에서는 하나의 펄스열이 동시에 양수와 음수를 표현할 수가 없기 때문에 양수와 음수를 따로 분리하여 표현해야 하기 때문이다.

그림 7의 구조를 이용하여 3층 순방향 신경회로망(3-layered feedforward neural network)을 구성하고 가중치들은 호스트 컴퓨터(host computer)에서 유도한 오차역전파(error backpropagation) 알고리즘으로 학습을 시키고 시뮬레이션은 학습된 가중치들을 사용하여 제안한 신경회로망 구조에서 펄스열을 이용하여 실험하였다. 식 (16)과 (17)은 제안한 신경회로망 구조에서 은닉층과 출력층간의 가중치를 구하기 위해 오차역전파 알고리듬을 구한 것이다.[5] (O_j는 은닉층의 출력값, O_k는 출력층의 출력값, Y_k는 시그모이드 출력함수의 입력, T_k는 목표값(target value), η 는 학습률이다.)

$$\Delta w_{kj}^{+} = \eta\,\delta_k \frac{\partial O_k}{\partial Y_k} Y_k \frac{1}{net_k^{+}} O_j,\quad \Delta w_{kj}^{-} = \eta\,\delta_k \frac{\partial O_k}{\partial Y_k} Y_k \frac{1}{net_k^{-}} O_j \quad (16)$$

$$\Delta w_{ji}^{+} = \eta\,\delta_j \frac{\partial O_j}{\partial Y_j} Y_j \frac{1}{net_j^{+}} O_i,\quad \Delta w_{ji}^{+} = \eta\,\delta_j \frac{\partial O_j}{\partial Y_j} Y_j \frac{1}{net_j^{-}} O_i, \quad (17)$$

식 (16)과 (17)에서 가중치의 변화량에 $\triangle w^+$와 $\triangle w^-$가 각기 있는 것은 양수와 음수를 따로 표현하므로 가중치를 보정(update)하기 이전의 부호에 따라 다르게 가중치를 보정하기 때문이다. 위의 식에서 사용된 δ_j와 δ_K는 다음과 같다.

$$\delta_k = T_k - O_k \quad (18)$$

$$\delta_j = \sum_k [\delta_k \frac{\partial O_k}{\partial Y_k} Y_k (\frac{1}{net_k^{+}} w_{kj}^{+} + \frac{1}{net_k^{-}} w_{kj}^{-})] \quad (19)$$

Ⅳ. 신경회로망 실험 결과 및 오차 해석

펄스열에서 0과 1인 펄스수의 비를 이용한 신경회로망을 검증하기 위해 그림 8과 같은 5×4의 입력 패턴에 대해 0에서 9까지의 숫자를 인식하는 문제에 적용을 하였다. 신경회로망은 입력층에 20개의 입력단위와 은닉층에는 18개의 은닉단위, 그리고 출력층에는 4개의 출력단위로 구성하였다. 학습은 오차역전파 알고리듬을 이용하여 Ⅲ절에서 제안한 모델을 가지고 학습알고리듬을 유도하여 사용하였다.

제안된 그림 7과 같은 신경회로망 구조로 그림 8의 입력 패턴에 대해 숫자인식을 하는데에는 난수발생기 부분을 제외하고 AND, OR 게이트 660여개와 D f/f 120여개, 그리고 105개 정도의 멀티플렉서를 사용하여 구현한다.

그림 8. 숫자 인식을 위한 5 4 입력 패턴
Fig. 8. 5×4 Input pattern for digit recognition.

본 논문에서 제안한 방법에 의한 신경회로망 구조에서는 세가지 종류의 오차라 존재한다. 첫번째 오차는 어떤 수를 펄스열로 바꾸어 표현하기 위해서 사용하는 난수발생기(random number generator)에 의한 오차이다. 난수발생기에 의한 오차는 두가지의 형태를 가지는데 하나는 난수발생기의 난수도(randomness)가 이상적이지 않은 경우 나타내고자 하는 수와 난수발생기에 의해 펄스열로 바꾸어져 표현된 수와의 차이가 생기는 경우이다. 다른 한가지는

난수발생기간에 상관도(correlation)가 있는 경우에 생기는 오차로서 확률연산에 사용되는 펄스열간에 상관도가 있는 경우, 예를 들면 곱셈을 하는 경우 두 입력 펄스열간에 상관도가 존재하면 정확한 연산이 이루어지지 않는다. 본 논문에서는 난수발생기의 난수도(randomness)는 충분하다고 가정하여 그에 따른 오차는 제외하였으며, 펄스열간에 존재하는 상관도에 의한 오차의 정도가 가중치를 저장하는 레지스터의 길이에 의한 양자화 간격(△)보다 작은 경우 펄스열간의 상관도가 신경회로망에 미치는 영향은 무시할 수 있다. 두번째 오차는 확률연산에 내재하는 펄스열의 길이에 따른 오차이다. 이 오차는 관측하는 펄스열 구간의 길이가 늘어날수록 줄어들게 된다. 그림 9는 그림 8의 입력패턴에 대해 숫자인식을 실험한 것으로 펄스열 구간의 길이를 변화시켜가며 오차의 평균 제곱오차(mean squared error)를 구하였다. 그림 9에서 가운데의 곡선이 이를 나타내고 위에 있는 곡선은목표값이 '1'인 것에 의한 평균 제곱 오차이고 아래의 곡선은 목표값이 '0'인 것에 의한 평균 제곱 오차를 나타낸 것이다. 이렇게 목표값이 '1'인 것에 의한 오차가 '0'인 것에 의한 오차보다 크게 되는 것은 단극성 수 표현방법을 사용하는 확률연산 방법에서는 오차의 분산이 $X(1-X)/L$이므로 X가 0.5에서 오차가 최대이고 0또는 1에서 최소가 되지만 0과 1의 펄스수의 비를 이용한 확률연산 방법에서는 $X(1-X)^2/L$ 이 되므로 X가 클수록 오차는 커지게 된다.

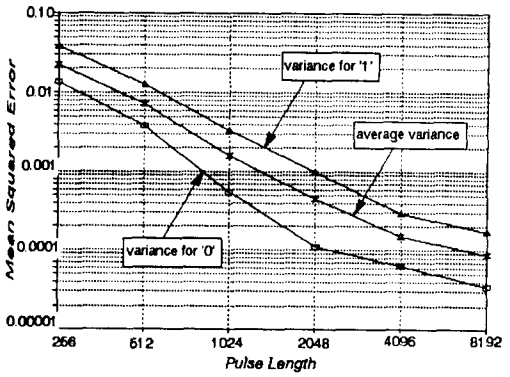


그림 9. 숫자 인식 문제에서의 펄스열의 길이에 따른 평균 제곱오차

Fig. 9. Mean squared error with various pulse lengths in the digit recognition problem.

그림 9에서 보면 펄스열의 길이가 두배로 늘어나면 출력의 평균 제곱오차는 25%씩 줄어드는 것을 알 수 있다. 그러나 펄스열의 길이가 4096이상이 되면서부터는 오차가 줄어드는 경향이 한계값에 가까와지면서 누그러드는 것을 알 수 있다. 이러한 한계값이 존재하는 것은 신경회로망을 목표값인 '0' 또는 '1'이 되도록 학습을 할때 무한히 학습을 하여 오차가 0이 되도록 학습을 하지 않고 목표값에서 각각 δ_0, δ_1이내의 범위로 오차가 들어오면 수렴을 한 것으로 하기 때문이며, 이때의 신경회로망의 출력의 기대값은 $0+\delta_0$, $1-\delta_1$이 된다. 세번째 오차는 가중치를 저장하는 레지스터의 유한한 길이에 따른 양자화 오차이다. 양자화 오차에 의한 오차는 가중치를 저장하는 레지스터의 길이가 가중치를 표현하기 위한 최소한의 길이보다 작을 경우 생긴다. 그림 9의 실험에서 사용한 가중치는 대부분 0.5에서 -0.5사이에 분포하며 정밀도는 소숫점이하 4자리로서 시뮬레이션을 할 때에는 16bit의 레지스터에 저장함을 가정하여 정수로 바꾸어 표현하였으므로 65536단계로 양자화한 것으로 충분히 가중치를 표현하였다. 그러나 모든 가중치를 16bit의 레지스터에 저장하기에는 하드웨어의 구현을 고려할 때 문제가 있으므로 되도록 최소한의 레지스터 길이로 가중치를 표현할 수 있도록 해야 한다. 그림 10은 펄스열의 길이가 128부터 2048까지의 각각에 대해 가중치를 저장하는 레지스터의 길이를 변화시켜가면서 평균 제곱오차를 구한 것이다. 그림 10으로부터 펄스열의 길이가 2^r일 경우 레지스터의 길이가 r 비트이상이 되면 더 이상 오차는 줄어들지 않음을 알 수 있다. 이러한 실험결과로부터 펄스열의 길이가 2^r일 경우 필요한 최소한의 레지스터의 길이는 r비트임을 알 수 있다.

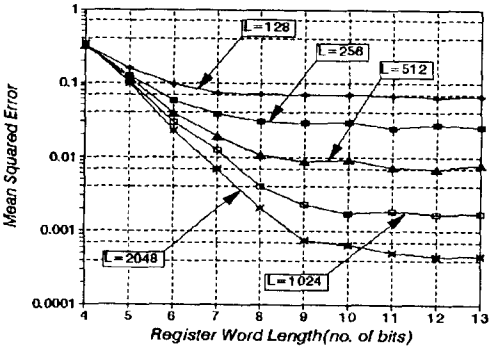


그림 10. 레지스터의 길이에 따른 양자화 오차의 영향에 의한 평균 제곱오차의 변화

Fig. 10. Mean squared error with the effect of quantization error of finite register length.

V. 결론

본 연구에서는 펄스열에서 펄스수의 비를 이용한 확률연산의 기본 연산기의 회로들을 제안하고 이러한 기본 연산기들의 특성과 오차를 실험하였다. 그리고 이러한 방법들을 기초로 하여 신경회로망을 구현하는 방법을 제안하고 이를 숫자인식문제에 적용하여 펄스열에서 0과 1의 펄스수의 비에 의한 확률연산을 이용한 신경회로망이 유용함을 보였다.

확률연산의 장점은 대량의 병렬처리를 위한 신경회로망의 하드웨어 구현이 적은 면적에 가능하다는 점이다. 이는 신경회로망의 연산에서 주로 수행하는 덧셈과 곱셈과 같은 기본 연산을 적은 수의 게이트로 구현할 수 있기 때문이다. 기존의 확률연산 방법은 덧셈과 곱셈을 본 논문에서 제안한 방법보다 좀 더 적은 게이트로 구현할 수 있지만 덧셈이 비선형 덧셈이므로 입력의 수가 늘어나면 정확한 덧셈이 이루어지지 않고 수의 표현범위가 제한된다는 단점을 가진다. 제안한 방법에서의 덧셈은 선형 덧셈을 할 뿐만 아니라, 나타내고자 하는 수의 역수를 인버터 한개로 표현할 수 있기 때문에 나눗셈이 곱셈기를 이용하여 쉽게 구현된다. 그리고 제안한 방법은 수의 표현범위가 넓다는 특성이 있다. 앞으로 연구해야할 과제는 제안한 방법의 특성을 더 파악하기 위해 입력 패턴에 잡음(noise)을 넣어주는 실험결과와 같이 보다 실제적이고 다양한 문제에 이를 적용하려는 시도와 오차의 해석에 대한 연구가 진행되어야 하겠다. 그리고 현재 이러한 제안한 신경회로망을 집적회로로 구현하는 연구를 진행하고 있다.

參 考 文 獻

[1] M. S. Tomlinson Jr., D. J. Walker, and M. A. Sivilotti, "A Digital Neural Network Architecture for VLSI" *Proceedings of IJCNN*, vol.2, pp. 545-550, 1990.

[2] E. W. Lee, and S. I. Chae, "Implementation of the Boltzmann Machine with Stochastic Computation," *Proceedings of World Congress on Neural Networks*, vol. 4, pp. 831-834, July, 1993.

[3] G. S. Eisman, "Frequency Based Computation in Neural Networks", *Proceedings of the IJCNN., Seattle*, Washington, vol.2, pp. 577-582, 1991.

[4] P. Mars, and W.J. Poppelbaum, "Stochastic and Deterministic Averaging Processors" Stochastic and Deterministic Averaging Processors, Peter Peregrinus, pp. 20-53, 1981.

[5] D. E. Rummelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, D. E. Rummelhart and J. L. McClelland, Ed. Cambridge, MA: MIT Press, 1988.

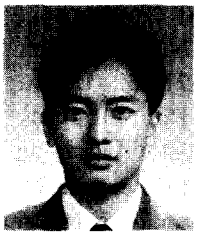
[6] Yoshikazu Kondo and Yasuji Sawada, "Functional Abilities of a Stochastic Logic Neural Network", IEEE. Trans. on Neural Networks, vol.3, no.3, pp. 434-443, 1992.

[7] 민 승재, 이 일완, 채 수익, "펄스열에서 1인 펄스수와 0인 펄스수의 비를 이용한 확률연산에 관한 연구," 제 3회 인공지능, 신경망 및 퍼지 시스팀 종합 학술대회/전시회 논문 집, pp. 489-493, Oct., 1993.

[8] S. J. Min, E. W. Lee, S. I. Chae, "A Study on the Stochastic Computation Using the Ratio of Zero Pulses and One Pulses," *Proceedings of International Symposium on Circuits and Systems*, London, pp. 471-474, 1994.

[9] 민 승재, 채 수익, "새로운 펄스연산방법을 이용한 Hopfield Network의 구현," 94 인공지능, 신경망 및 퍼지시스팀 춘계종합학술대회 논문집, pp. 385-390.

著者紹介



閔 勝 載(準會員)

1970年 11月 12日生. 1993年 2月 서강대학교 전자공학과(학사). 1993年 3月 - 현재 서울대학교 대학원 전자공학과 석사과정 재학중. 서강 알파 시그마 누 회원. 주관심 분야는 집적회로 설계와 신경회로망 등임.

蔡 洙 翊(正會員)

1952年 11月 2日生. 1976年 2月 서울대학교 전기공학과(학사). 1978年 2月 서울대학교 대학원 전기공학과(석사). 1979年 8月 - 1982年 3月 공군사관학교 교수부교관. 1987年 (美)Stanford 대학교 전기공학과(박사). 1978年 11月 ~ 1988年 7月 ZyMos Corp. Design Manager. 1988年 8月 - 1990年 5月 대우통신 근무. 1990年 7月 ~ 현재 서울대학교 반도체 공동연구소 및 전자공학과 조교수. 주관심 분야는 신경회로망과 집적회로 설계 등임.