1999年度 大韓電子工學會 秋季綜合學術大會 論文集 第 22 卷 第2號 99/11

재구성 가능한 신경망 프로세서의 설계

장 영 진(張 榮 眞), 이 현 수(李 顯 洙)
경희대학교 전자계산공학과
전화 : 0331-201-2947 팩스 : 0331-202-1723

A Design of Reconfigurable Neural Network Processor

Young Jin Jang and Hyon-Soo Lee
Dept. of Conputer Engineering, KyungHee University
E-mail : ddsmurf@cann.kyunghee.ac.kr/hslee@nms.kyunghee.ac.kr

Abstract

In this paper, we propose a neural network processor architecture with on-chip learning and with reconfigurability according to the data dependencies of the algorithm applied.

For the neural network model applied, the proposed architecture can be configured into either SIMD or SRA(Systolic Ring Array) without any changing of on-chip configuration so as to obtain a high throughput. However, changing of system configuration can be controlled by user program. To process activation function, which needs amount of cycles to get its value, we design it by using PWL(Piece-Wise Linear) function approximation method. This unit has only single latency and the processing ability of non-linear function such as sigmoid, gaussian function, etc. And we verified the processing mechanism with EBP(Error Back-Propagation) model.

Ⅰ. 서론

신경망의 현재 모델들은 벡터 컴퓨터, 웍스테이션, 전용의 코프로세서 또는 병렬 컴퓨터상에서 시뮬레이션에 의해 실행되고 실제 응용 문제를 풀기위해 사용되고 있다. 이 같은 뉴로 시뮬레이터들이 가지고 있는 문제점은 신경망의 본질적인 공간-시간적 병렬성(spatio-temporal parallelism)을 부분적 또는 전체적으로 충분히 반영하지 못한다는 점과 시뮬레이션되는 신경망의 크기가 증가함에 따라 비례하여 계산시간이 폭발적으로 증가한다는 것이다[1]. 따라서, 현재의 범용 컴퓨터 처리능력은 신경망을 이용한 시스템의 연구 및 실질적인 개발의 수행을 제한하는 중요한 요소로 고려된다. 범용 컴퓨터의 처리능력을 향상시키기 위한 방법으로 특정의 신경망 모델만을 풀 수 있는 전용의 ASIC 칩들이 개발되어 사용되고 있다[2 - 5]. 이들 전용 ASIC은 각 신경망 모델에 대해 비슷한 병렬성과 계산 처리방식을 갖는 단일 혹은 복수개의 모델만을 처리할 수 있으며, 병렬성에 따라 SIMD 또는 SRA(Systolic Ring Array)의 구조로 구현이 되었다. 그러나, 이러한 단일의 구조는 소수의 신경망 모델에만 사용할 수 있기 때문에 범용성이 부족하고, 모델을 구조에 맞도록 알고리즘의 변경이 필요하다.

따라서, 본 논문은 신경망의 알고리즘이 갖는 병렬성을 극대화시키기 위해 적용된 모델의 데이터 의존관계에 따라 사용자의 프로그램에 의해 동적으로 SIMD 또는 SRA의 형태로 시스템의 구성을 변경이 가능한 구조를 제안한다. 다음 절에서는 제안한 신경망 프로세서와 PE(Processing Element)의 구조를 설명하고, 3절에서는 EBP 알고리즘을 적용하여 동작 가능성을 검증한다. 4절은 제안한 구조의 성능을 평가한다.

Ⅱ. 제안한 시스템의 구조

범용의 신경망 알고리즘을 처리할 수 있는 신경망 프로세서의 구조를 설계하기 위해서 먼저 각각의 신경망 모델에 대한 구조적 분석이 필요하다. 이러한 고려사항을 토대로 본 논문에서 제안한 전체 시스템의 구조는 그림 1과 같다. 시스템은 기본적으로 SRA의 구조로 구성이 되며, 사용자의 프로그램에 의해 SIMD 구조로의 재구성이 가능하다. 시스템 상에서 데이터는 32-비트 고정 소수점으로 표현되며, 호스트와의 인터페이스를 통해 데이터와 프로그램을 교환한다.

시스템을 구성하는 PE의 내부구성은 그림 2와 같다. PE는 고정 소수점 가/감산기, 곱셈기로 구성된 데이터

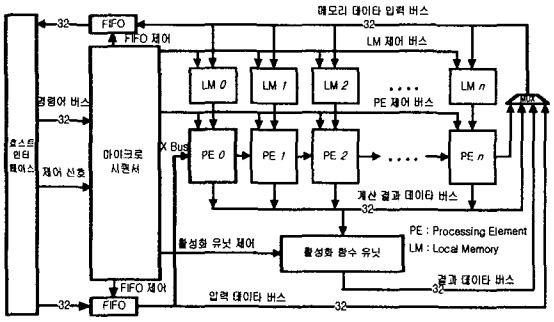

그림 1. 제안한 신경망 프로세서의 구조.

처리기와 중간 결과와 초기화 파라미터를 저장하기 위한 레지스터와 1KW의 로컬 메모리(LM)로 구성된다. 데이터 처리기의 기본 동작은 DSP와 같은 MAC (Multiplier-ACcumulator) 연산을 반복적으로 수행한다. 감산기는 신경망 알고리즘 상의 에러량의 계산과 종료 조건의 판단, 그리고 크기 비교를 위해 사용된다. PE 외부의 인터페이스는 데이터 입·출력, 메모리, 계산결과의 출력포트의 4개의 포트를 가지고 있으며, 인접 PE와의 쉬운 연결을 보장하여 보다 높은 확장성을 제공할 수 있도록 하였다. PE 상의 연산은 레지스터-레지스터 연산을 기본으로 수행하며 레지스터-메모리 이동 명령어를 통해 레지스터와 메모리간의 데이터를 교환한다. 이들 연산은 모두 1-싸이클에 수행할 수 있도록 설계하였다.

신경망 알고리즘의 기본적인 연산 중의 하나는 뉴런의 출력 값에 대한 활성화 함수의 계산이다. 이 함수는 전형적으로 비선형 함수이며 포화 함수(시그모이드 함수)와 종모양 함수(가우시안 함수)가 있다. 기존의 신경망 프로세서는 활성화 함수의 계산을 위해 look-up 테이블 방법과 함수 근사를 이용한 방법을 사용하였지만, 제안한 신경망 프로세서는 이들 두 방법의 장점을 최대로 활용하기 위해 적은 메모리와 정확한 근사값을 얻을 수 있는 PWL(Piece-Wise Linear) 함수 근사법[8]을 사용하였다. Look-up 테이블은 활성화 함수를 일정한 임의의 세그먼트로 분할한 경계 값을 저장하고 있으며, 이 값을 이용한 선형방정식을 데이터 처리기에서 계산함으로써 더 정밀하고 빠른 시간에 활성화 함수 값을 얻을 수 있다. 활성화 함수의 구성은 그림 3과 같다.

본 논문에서 제안한 데이터 병렬성에 따른 동적 재구성을 구현하기 위하여 그림 4와 같은 시스템 구조 변경 회로를 구성하였다. 데이터의 병렬성은 지역적 또는 전역적으로 표현되며, 신경망은 단일의 모델내에 지역적 또는 전역적 병렬성을 모두 가지므로 데이터의 의존관계를 고려한 시스템의 구현이 복잡하다. 그러나,

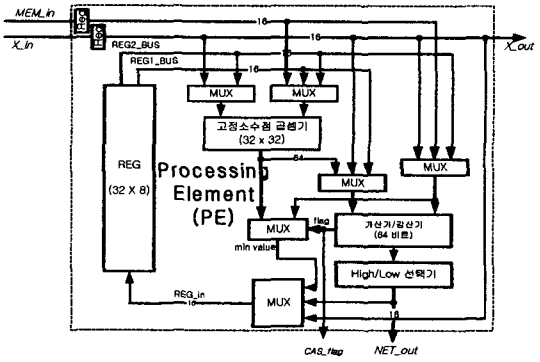

그림 2. 1-PE의 내부구조.

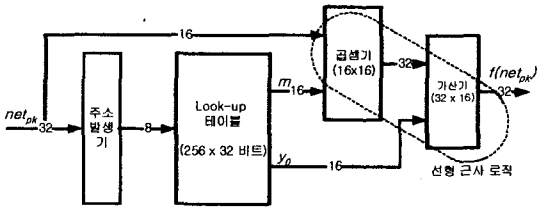

그림 3. 활성화 함수 유닛의 구조.

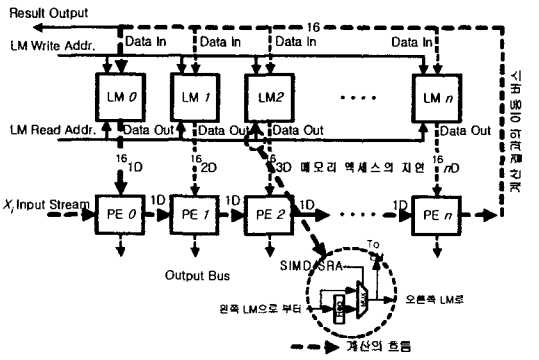

그림 4. 시스템 구조의 변경 - SRA의 경우.

제안한 구조상에서는 데이터의 의존관계가지역적인 특성을 가지는 알고리즘의 영역에서는 SRA의 구조로 프로세서가 동작하고, 전역적인 영역에서는 SIMD의 형태로 재구성되어 동작한다. 이러한 프로세서의 재구성은 적용한 모델에 대한 사용자의 프로그램에 의해 제어된다.

Ⅲ. EBP 모델의 처리 과정

에러 역전파(EBP) 모델은 가장 널리 쓰이는 교사 신호에 의한 신경망 학습 모델로써 모멘텀을 가진 gradient descent 학습 알고리즘을 사용한다. EBP는 그림 5에 보인 것처럼 다층의 구조를 가지며 feed-forward 와 feed-backward 과정을 통해 학습을 수행

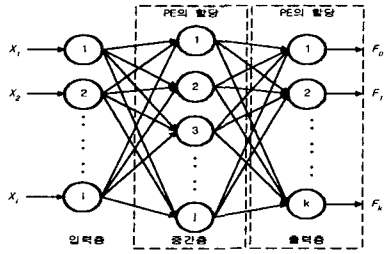

그림 5. EBP 모델의 네트워크 구조.

한다.

노드 병렬성을 이용하여 레이어상의 노드는 제안한 프로세서상의 PE에 순차적으로 1:1 맵핑된다. 만일, PE의 수 j가 레이어상의 노드의 수 N보다 적은 경우 PE의 수만큼 분할하여 $\lceil j/N \rceil$ 만큼 반복 수행한다.

제안한 구조상에서 EBP 모델의 동작을 검증하기 위하여, 먼저 EBP 모델 중 feed-forward 알고리즘의 처리과정을 설명한다. Feed-forward 알고리즘은 수식 (1), (2)와 같다.

$$net_{pj} = \sum_i w_{ji} \cdot O_x \qquad (1)$$

$$O_{pj} = f(net_{pj}) \quad f(x) = 1/(1+\exp^{-x}) \qquad (2)$$

이전 레이어의 출력은 수식 (1)에 의하여 가중치와의 곱의 합으로 j 노드의 $net(j)$값이 결정된다. 수식 (1)을 제안한 구조에 맵핑하기 위하여 recurrsion 알고리즘으로 변환하면, 수식 (3)과 같다. 이러한 recursion 알고리즘은 지역적 병렬성을 가지므로 SRA 구조를 사용하여 구현한다(그림 6).

$$net_{pj}^t = net_{pj}^{t-1} + w_{ji}^t \cdot O_i^t \qquad (3)$$

net_{pj}^t : t 시간에서의 j 노드의 $net(j)$ 값

j 노드의 $net(j)$ 값이 구해지면, 노드의 출력을 구하기 위하여 수식(2)의 시그모이드 함수를 통하여 얻을 수 있다. 시그모이드 함수는 $f(x)=1/(1+\exp^{-x})$를 사용하였고, [-1, +1] 구간에 대해 256개의 세그먼트로 분할하여 각각의 기울기와 y 절편을 look-up 테이블에 저장하여 함수 근사를 하였다. 따라서 시그모이드 함수는 데이터 처리기를 이용하므로 1D의 latency를 가진다. 출력 층에 대해서도 수식(2)와 (3)의 방법이 동일하게 적용이 된다.

Feed-forward 과정이 끝나며, 교사신호와의 오차 계산이 필요하다. 에러량 e_k^t는 출력층의 출력값 O_k와 목표값 T와의 차에 의하여 구할 수 있다.

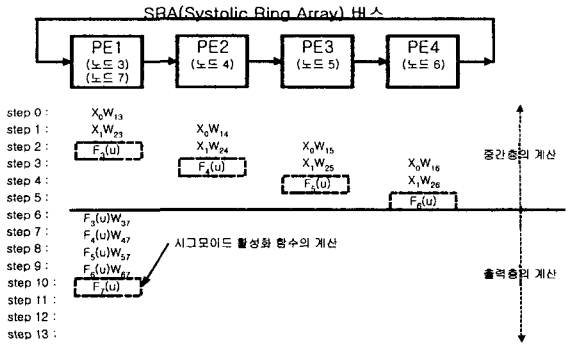

그림 6. Feed-forward 처리과정(PE가 4개인 경우).

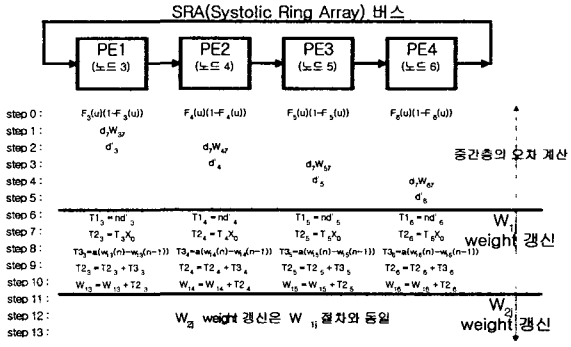

그림 7. 중간층의 weight 갱신 과정(PE가 4개인 경우).

두 번째로, feed-backward 과정에 대한 데이터 흐름을 고려해 본다. Feed-backward 과정은 출력 층과 중간층에서 발생하고 각 입력에 대한 연결강도를 갱신하기 위한 것이다. 먼저 각 층의 연결 강도를 조절하기 위한 출력층의 뉴런 k와 중간층의 뉴런 j에 대한 에러량은 각각 수식 (4)와 (5)를 이용하여 구할 수 있다.

$$\delta_k^t = e_k^t \cdot O_k \cdot [1-O_k] \qquad (4)$$

$$\delta_j^t = O_j^t \cdot [1-O_j^t] \cdot \sum_k \delta_k^{t+1} \cdot w_{kj}^{t+1} \qquad (5)$$

오차 δ 항이 계산된 다음 수식 (6)을 이용하여 각 레이어에서의 네트워크 연결 강도를 조절한다.

$$w_{ji}^{t+1} = w_{ji}^t + \alpha \cdot [w_{ji}^t - w_{ji}^{t-1}] + \eta \cdot \delta_j^t \cdot O_i^{t-1} \qquad (6)$$

Feed-backward 과정은 LM에 저장된 상수항(η, α), weight, feed-forward 과정의 결과 값에 의하여 연산이 이루어지므로 각 PE에서 동시에 연결강도를 갱신할 수 있다. 따라서 feed-backward 과정은 SIMD 방식으로 시스템을 구성한다(그림 7).

Ⅳ. 성능평가 및 고찰

재구성 가능한 구조상에서 한 레이어를 계산하기 위하여 필요한 계산시간은 SIMD, SRA, 그리고 두 개의 모델이 혼합적으로 사용되는 경우에 대하여 각각 구할 수 있다.

표 1. 제안한 신경망 프로세서의 계산시간.

시스템 구조	계산 시간(Computation Time)
SRA	$\{n_{sra}+(n_{sra}-1)\}\cdot\lceil j/N\rceil$
SIMD	$n_{simd}\cdot\lceil j/N\rceil$
SRA+SIMD	$[n_{simd}+\{n_{sra}+(n_{sra}-1)\}]\cdot\lceil j/N\rceil$

n_{simd} : 한 노드의 처리를 위한 SIMD 명령어의 수

j : 레이어 상의 노드의 수, N : 시스템 안의 PE의 수

n_{sra} : SRA 명령어의 수

EBP 모델의 성능을 기존의 병렬 프로세서와 비교하기 위하여 64x64x1과 16x5x1의 구조를 가진 모델을 고려하였다. 이들 네트워크의 계산을 위해 필요로 되는 계산 시간과 입력에서 출력까지의 latency를 비교하면 표 2와 같다. 본 논문에서 제안한 구조는 계산 시간에서 네트워크의 크기가 클수록 뛰어난 성능을 보였으며, latency time에 대해서도 다른 구현에 비해 3~8배정도 빠른 결과를 얻을 수 있었다.

표 2. MA-16, CNAPS, 제안한 시스템과의 Feed-forward 계산시간의 비교.

네트워크 구조	시스템	시스템 구성	계산 시간	Latency 시간
64x64x1	CNAPS[5]	8-bit/20MHz	8μs	unknown
	SAND[7]	16-bit/40MHz	5.1μs	27μs
	Proposed	16-bit/50MHz	2.8μs	8.26μs
16x5x1	MA-16[5]	16-bit/8MHz	5.5μs	8μs
	SAND[7]	16-bit/40MHz	0.5μs	3.6μs
	Proposed	16-bit/50MHz	0.7μs	1.04μs

제안한 신경망 프로세서는 프로그램 가능성에 의한 범용성과 확장성, 재구성 가능한 병렬성을 효과적으로 얻을 수 있다. 그러나, PE와 LM간의 연결은 높은 bandwidth(N x 16 bit)를 요구하므로, 이를 극복할 수 있는 구현 기술이 필요로 된다.

Ⅴ. 결론

본 논문에서는 데이터의 병렬성에 따라 동적으로 프로세서의 재구성이 가능한 범용의 신경망 프로세서의 구조를 제안하고 성능을 평가하였다. 제안한 구조는 범용의 알고리즘을 고속 실행할 수 있는 범용성과 적용하고자 하는 신경망 모델의 데이터 병렬성을 지역적 또는 전역적 병렬 알고리즘으로의 변환하는 과정을 제거함으로써 사용자에게 편리한 개발 환경을 제공한다. 또한, PE의 구조를 단순화시킴으로써 신경망 네트워크 구조에 유연하게 대응할 수 있는 높은 확장성을 제공하였다.

앞으로의 연구방향은 현재 설계된 신경망 프로세서의 구조를 VLSI로 구현하고, IRAM(Intelligent RAM)과 같은 Embedded 메모리 ASIC의 구현을 위한 구조를 개선하는 것이다.

[참고문헌]

[1] Ramacher, "VLSI Design of Neural networks", Kluwer Academic Publisher, 1991.

[2] Takayaki Morishita, et al., "Neural Network Multiprocessors Applied with Dynamically Reconfigurable Pipeline Architecture", IEICE Trans. Electron.,Vol. E-77-C, No. 12, Dec. 1994.

[3] Shinji Komori, et al., "A 3.2 GFLOPS Neural Network Accelerator", IEICE Trans. Electron., Vol. E80-C, No. 7, Jul.1997.

[4] IBM, "ZISC 036 Neurons User's Manual", Jan 9, 1997.

[5] Hammerstrom, "A VLSI architecture for high performance, low-cost, on-chip learning", IJCNN, Vol. 2, Sandiego, 1990.

[6] Fadi N. Sibai, et al., "A Time-Multiplexed Reconfigurable Neuroprocessor", IEEE Micro, Vol. 17, No. 1, Jan/Feb. 1997

[7] Thomas Becher, et al., "The MiND-project: building, applying and speeding-up neural networks using te SAND-neuroprocessor", EUFIT, 1997.

[8] Paolo, et al., "Programmable VLSI Systolic Processors for Neural Network and Matrix Computations", PHD Thesis, Ecole Polytechnique Federale de Lausanne-LAMI, Lausanne , 1996.