# Approximate computing 기법을 이용한 FPGA 기반 인공 신경망 가속기 최적화

박상우, 김한이, 서태원
고려대학교 컴퓨터학과
e-mail : psw0113@korea.ac.kr

# FPGA-based Artificial Neural Network Accelerator Optimization Using Approximate Computing

Sangwoo Park, Hanyee Kim, Taeweon Suh
Dept. of Computer Science and Engineering, Korea University

요 약

본 연구에서는 이미지를 분류하는 인공 신경망 가속기를 최적화했고, 이를 구현하여 기존 인공 신경망 가속기와 성능을 비교 분석했다. FPGA(Field Programmable Fate Array) 보드를 이용하여 가속기를 구현했으며, 해당 보드의 내부 메모리인 BRAM 을 FIFO(First In First Out)구조로 설계하여 메모리 시스템을 구현했다. Approximate computing 기법을 효율적으로 적용하기 위해 FWL(Fractional Word Length)최적점을 분석했고, 이를 기반으로 인공 신경망 가속기의 부동 소수점 연산을 고정 소수점 연산으로 변환했다. 구현된 인공 신경망 가속기는 기존의 인공 신경망에 비해, 약 7.4%더 효율적인 전력소모량을 보였다.

## 1. 서론

뉴럴 네트워크는 다양한 어플리케이션 도메인에서 좋은 성능을 보여주고 있으며 그 중에서도 특히 음성, 이미지, 텍스트의 패턴 인식과 분류에 특화되어있다 [1-3]. 뉴럴 네트워크는 인간의 신경망을 모방하여 설계되었으며, 다양한 트레이닝 데이터를 통해 학습하는 것이 가능한 알고리즘이다. 이러한 뉴럴 네트워크는 트레이닝 데이터가 많아지거나 모델의 파라미터 숫자가 증가할수록 분류 정확도에 있어서 좋은 성능을 보여 준다[4,5]. 그리고 이렇게 깊은 레이어 층을 구성하여 많은 파라미터를 가지는 뉴럴 네트워크를 '딥 뉴럴 네트워크'라고 부른다.

하지만 딥 뉴럴 네트워크는 모델의 크기가 커질 수록 많은 파라미터를 가지게 되고, 그만큼 많은 연산량을 가지기 때문에 처리시간이 오래 걸린다는 단점이 존재한다. 따라서 딥 뉴럴 네트워크에서는 속도가 중요한 요소였으며, 인공 신경망 연산에 대한 S/W 적인 다양한 가속화 연구가 진행되었다. 최근에는 S/W 적인 가속화 방법 뿐만 아니라, 인공 신경망 연산에 특화된 H/W 설계에 대해서도 활발한 연구가 진행되고 있다[6]. 본 논문에서는 고효율 인공지능 가속기 설계를 주제로, 정확도는 다소 하락하지만 성능을 향상시킬 수 있는 Approximate computing 기법 적용에 대해 설명한다.

이 논문은 2019 년도 정부의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임 (NRF-2017R1D1A1B03028926)

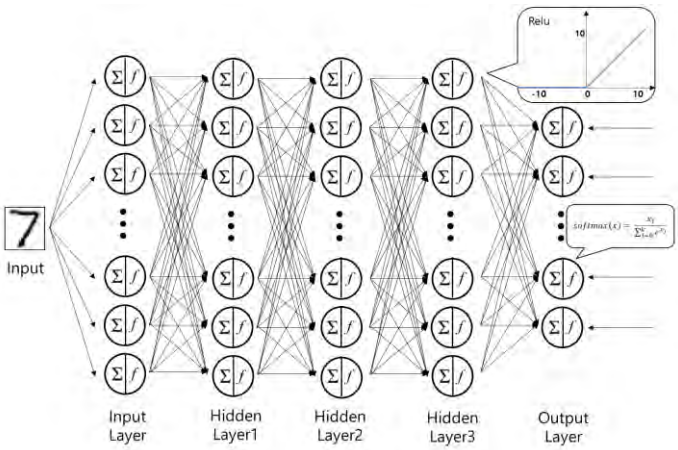
## 2. 선행연구

인공 신경망의 기본적인 구조는 Input 레이어, Hidden 레이어, Output 레이어로 구성되며 각 레이어들은 특정 개수의 뉴런들이 존재한다. 또한 각 뉴런들은 활성화 함수를 가지는데 비선형 활성화 함수들이 복잡한 문제를 해결하는데 적합하여 활성화 함수에는 주로 비선형 함수들이 사용되고 있다.

인공 신경망은 FP (Forward Phase), BP (Backward Phase), 가중치 갱신의 3 가지 작업을 순차적으로 진행하여 학습한다. FP 에서는 입력을 받아 레이어의 각 뉴런들의 가중치들과 곱셈 연산을 한 후, 그 결과들을 모두 더하고 활성화 함수를 거쳐 다음 레이어로 전파한다. 이 과정을 반복하여 Output 레이어까지 도달하면 FP 가 끝난다. BP 는 경사 하강법을 통해 모든 가중치들의 오차를 계산한다. 그리고 이 계산된 오차를 통해 가중치들을 갱신하여 인공신경망이 학습된다.

본 연구에서는 Input layer, 4 Hidden layer, Output layer 로 구성된 완전연결 뉴럴네트워크 모델을 구현하였으며, 각 Hidden layer 에는 활성화 함수로 ReLU(Rectified Linear Unit)를, Output 레이어에서는

Softmax 함수를 사용했다. 모든 Hidden layer 의 뉴런의 개수는 각각 512 개이며 완전연결 뉴럴네트워크의 구조에 따라 layer 마다 512*512 만큼 Weight 가 존재한다. [그림 1]은 본 논문에서 실험한 인공 신경망구조를 보여준다.



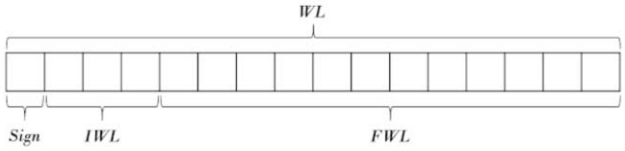

[그림 1] 인공 신경망 구조

## 3. Approximate computing 기법 적용

본 연구에서는 [그림 2]와 같이 32bit 의 부동소수점 변수를 16bit 고정소수점 방식으로 변환하는 Approximate Computing 전처리기법을 적용했다. 부동소수점에서 고정소수점으로 변환 시 소수점 이하 부분을 표현하는 Fractional Word Length (FWL, [그림 3]) 의 비트수에 따라 정확도에 차이가 발생한다. 이에 따라 적절한 FWL 을 찾기 위해 비트 수를 조절하며 실험한 결과, 16bit 부동소수점에서 FWL 을 9 로 설정하는 것이 적절하였다 (<표 1> 참고). 32bit 부동소수점에서 16bit 고정소수점으로 변환 시 이전보다 정확도가 약 0.27% 감소하지만, 고정소수점 방식은 실수의 사칙연산을 간단한 정수의 사칙연산으로 처리하기 때문에 별도의 Floating Point Unit (FPU)가 요구되지 않아 하드웨어 구조가 단순해지고, 전력소비가 감소하는 장점을 가진다.

```
int float2fix(float f, int wl, int iwl) {
    ieee754_float standard;
    standard.f = f;

    int ret = standard.ieee.mantissa | (1 << MANTISSA);
    int exp = standard.ieee.exponent - EXP_BIAS;
    int fwl = wl - iwl - 1;
    int fraction = MANTISSA - exp;
    int filter = (1 << wl) - 1;

    if(fraction > INT_SIZE) {
        ret = ret >> fraction - INT_SIZE;
        fraction = INT_SIZE;
    }

    ret = ret >> fraction - fwl;

    if(standard.ieee.negative) {
        ret = ~ret + 1;
    }

    return ret & filter;
}
```

[그림 2] 고정 소수점으로 변환하는 코드



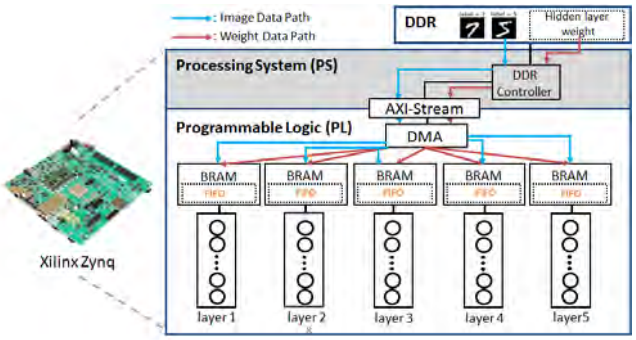

[그림 3] 고정 소수점 구조

[표 1] FWL 변화에 따른 고정소수점 정확도

| Data Type | FWL | Accuracy |
|---|---|---|
| float32 | - | 98.41% |
| fix16 | 10 | 92.40% |
| fix16 | 9 | 98.14% |
| fix16 | 8 | 97.88% |

## 4. FPGA 기반 인공 신경망 가속기 메모리 시스템 최적화

Xilinx 사의 Zynq Ultrascale+ 보드에 구현된 인공 신경망 가속기의 구조는 [그림 4]와 같다. 인공 신경망 연산 엔진은 Programmable Logic (PL)에 포팅되며, 해당 하드웨어가 연산을 진행할 때는 Processing system (PS)의 DDR Controller 를 거쳐 DRAM 로부터 데이터를 공급 받는다. 이때 인공 신경망 엔진의 효율적인 연산을 위해, BRAM 이라는 PL 영역 내부 메모리에 데이터가 블럭단위로 저장된다. 하지만 BRAM 은 그 크기가 제한적이기 때문에, 해당 인공 신경망 연산에 필요한 가중치 및 입력 데이터를 모두 저장할 수 없다. 따라서 내부 매모리 시스템을 FIFO(First In First Out) 구조로 설계했고, 입력 데이터들을 DDR 로부터 순차적으로 최대 16Kbits 만큼 읽어와 BRAM 에 저장하도록 구현했다. 이에 따라 인공 신경망 연산 유닛도 BRAM 데이터를 순차적으로 읽어 연산하도록 설계하여 가속기의 메모리 시스템을 최적화했다.




[그림 4] Xilinx Zynq 보드에 구현된 인공 신경망 가속기 구조

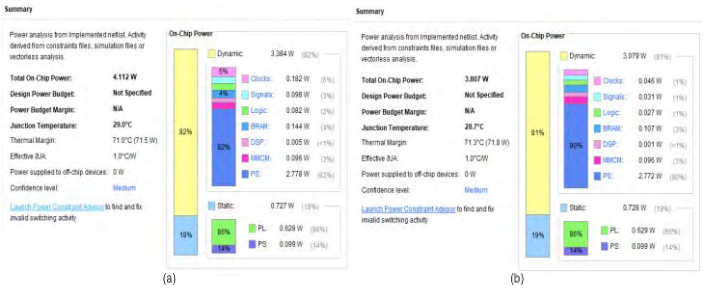
## 5. 실험 환경 및 결과분석

우리는 인공 신경망의 가중치를 학습시키기 위해 MNIST 손 글씨 데이터 셋을 데스크톱 PC 환경에서 학습시켰다. 학습이 끝난 인공 신경망의 가중치들은 98.41%의 정확도를 가졌고, 이를 [그림 1]과 같이 5 개의 레이어를 가지는 인공 신경망 가속기를 FPGA 보드에 구현했다.

우리는 구현된 인공 신경망 가속기의 성능을 검증하기 위해 MNIST 손 글씨 테스트 데이터셋 10,000 장을 추론하는데 소비되는 전력량, 그리고 추론 정확도를 비교했다. [표 2]는 기존 인공 신경망 가속기와 본 논문에서 최적화한 인공 신경망 가속기를 비교한 결과이다. 최적화된 인공 신경망 가속기는 기존 인공 신경망 가속기보다 정확도가 0.27%감소하지만, 7.4% 더 효율적인 전력 소비량을 보였다[그림 5].

| 인공 신경망 가속기 | 정확도 | 전력소모 |
|---|---|---|
| 기존의 인공 신경망 가속기 | 98.41 % | 4.112 W |
| 최적화된 인공 신경망 가속기 | 98.14 % | 3.807 W |

<표 2> 인공 신경망 가속기 성능차이




[그림 5] 32bit 부동소수점 기반 인지컴퓨팅 엔진 전력소비량(a)과, 16bit 고정소수점 기반 인지컴퓨팅 엔진 전력소비량(b)

## 6. 결론

본 논문에서는 FPGA 보드에 이미지를 분류하는 인공 신경망 가속기를 구현했고, 기존 인공 신경망 가속기와 성능을 비교 분석하였다. 인공 신경망 연산 엔진이 보드의 내부 메모리인 BRAM 을 효율적으로 사용하도록 하기 위해 내부 메모리 구조를 FIFO 로 설계했으며, 에너지 효율적인 인공 신경망 가속기를 구현하기 위하여 Approximate computing 기법을 적용했다. Approximate computing 기법을 적용할때는 정확도 손실을 최소화하기 위해, FWL 의 최적점을 찾는 실험을 진행했다. 그 결과 FWL 을 9 로 설정한 16 비트의 부동소수점 구조가 정확도 손실 대비 전력 소모가 효율적인 것을 알아냈다. 최적화된 인공 신경망 가속기는 32 비트 부동 소수점 연산을 16 비트 고정 소수점으로 변환하여 기존 인공 신경망 가속기보다 정확도가 0.27%로 감소했지만, 7.4% 더 효율적인 전력 소비량을 보였다.

## 참고문헌

[1] G. Dahl, D. Yu, L. Deng, and *A. Acero. Context-dependent pre-trained deep neural networks for large vocabulary speech recognition*. IEEE Transactions on Audio, Speech, and Language Processing, 2012.

[2] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber. *Deep big simple neural nets excel on handwritten digit recognition*. CoRR, 2010.

[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. *A neural probabilistic language model*. Journal of Machine Learning Research, 3:1137–1155, 2003.

[4] A. Coates, H. Lee, and A. Y. Ng. *An analysis of single-layer networks in unsupervised feature learning*. In AISTATS 14, 2011.

[5] Q.V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A.Y. Ng. *On optimization methods for deep learning*. In ICML, 2011.

[6] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao and J. Cong, *Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks*, FPGA'2015, 2015.