The Journal of the Convergence on Culture Technology (JCCT)
Vol. 3, No. 4, pp.165-169, November 30, 2017. pISSN 2384-0358, eISSN 2384-0366

http://dx.doi.org/10.17703/JCCT.2017.3.4.165
JCCT 2017-11-21

GPGPU와 Combined Layer를 이용한 필기체 숫자인식 CNN구조 구현

Implementation of handwritten digit recognition CNN structure using GPGPU and Combined Layer

이상일*, 남기훈**, 정준모***

Sangil Lee*, Kihun Nam, Jun Mo Jung*****

요 약 CNN(Convolutional Nerual Network)는 기계학습 알고리즘 중에서도 이미지의 인식과 분류에 뛰어난 성능을 보이는 알고리즘 중 하나이다. CNN의 경우 간단하지만 많은 연산량을 가지고 있어 많은 시간이 소요된다. 따라서 본 논문에서는 CNN 수행과정에서 많은 처리시간이 소모되는 convolution layer와 pooling layer, fully connected layer의 연산수행을 SIMT(Single Instruction Multiple Thread)구조의 GPGPU(General-Purpose computing on Graphics Processing Units)를 통하여 병렬로 연산처리를 수행했다. 또한 convolution layer의 출력을 저장하지 않고 pooling layer의 입력으로 바로 사용함으로 메모리 접근횟수를 줄여 성능 향상을 기대했다. 본 논문에서는 이 실험검증을 위하여 MNIST 데이터 셋을 사용하였고 이를 통하여 제안하는 CNN 구조가 기존의 구조보다 12.38% 더 좋은 성능을 보임을 확인했다.

주요어 : 기계학습, CNN, GPGPU, 스레드, 필기체인식, MNIST

Abstract CNN(Convolutional Nerual Network) is one of the algorithms that show superior performance in image recognition and classification among machine learning algorithms. CNN is simple, but it has a large amount of computation and it takes a lot of time. Consequently, in this paper we performed an parallel processing unit for the convolution layer, pooling layer and the fully connected layer , which consumes a lot of handling time in the process of CNN, through the SIMT(Single Instruction Multiple Thread)'s structure of GPGPU(General-Purpose computing on Graphics Processing Units).And we also expect to improve performance by reducing the number of memory accesses and directly using the output of convolution layer not storing it in pooling layer. In this paper, we use MNIST dataset to verify this experiment and confirm that the proposed CNN structure is 12.38% better than existing structure.

Key Words : machine learning, CNN, GPGPU, thread, Handwriting Recognition, MNIST

*준회원 , 서경대학교 전자컴퓨터공학과
**정회원, 서경대학교 컴퓨터공학과
***정회원 , 서경대학교 전자공학과(교신저자)
접수일: 2017년 9월 13일, 수정완료일: 2017년 9월 25일
게재확정일: 2017년 10월 10일

Received: 13 September, 2017 / Revised: 25 September, 2017
Accepted: 10 October, 2017
***Corresponding Author: sang1l@skuniv.ac.kr
Dept. of Electronics Engineering, SeoKyeong University

Ⅰ. 서 론

사람이 사물을 바라보고 인지하는 능력을 컴퓨터에 부여하고자하는 연구가 오래전부터 폭넓게 이루어져 왔다. 하드웨어 기술의 급격한 발달은 많은 응용 영역에서 실용화되어 실생활에 여러 분야에서 이용되고 있으며 컴퓨터가 직접 사물을 인지하고 판독할 수 있는 능력의 기반을 가져왔다.[1]

과거 딥러닝은 많은 연산량 때문에 처리 시간이 오래 걸려 침체기에 빠져있었으나 컴퓨터의 연산처리능력이 향상되고 GPGPU를 이용한 고속화가 가능해졌기 때문에 연산 처리 시간적인 문제가 해결되면서 다시 화제로 떠올랐다.

기계학습 분야 중 인공 신경 회로망의 일종인 CNN은 동물의 두뇌 신경이 동작하는 방법에 기안하여 설계되었다.[2] 본 논문에서는 연산의 가속화를 위하여 GPU를 사용하였다. GPU를 이용한 고속화연구는 CPU 성능의 한계극복을 위해서 시작되었으며

GPU가 연산을 수행하는 ALU의 개수가 많기 때문에 대규모 연산에 CPU를 이용하는 것 보다 유리하다.[3] 본 논문에서는 이러한 GPGPU의 특징을 딥러닝의 기술 중 하나인 CNN의 연산을 GPGPU를 통 하여 병렬로 수행하고 Combined Layer라는 구조를 사용하여 속도 향상을 기대했다.

본 논문에서는 본론 2장에서는 연산을 가속화하는 GPU와 그 구조에 대한 설명하며 3장에서는 CNN에대한 설명과 제안하는 CNN의 구조를 설명하고 4장에서는 실험 결과 그리고 5장에서 결론으로 마무리된다.

Ⅱ. 연산을 가속화하는 GPU구조

GPU는 본래 3D 그래픽 처리를 위해 사용되었던 GPU의 우수한 계산능력을 이용하여 그래픽연산뿐만 아니라 일반 컴퓨터에서도 GPU의 역할을 수행하는 연구가 시작되었다. 수천개의 코어로 구성되며 CPU보다 많은 ALU를 가지는 GPU는 병렬처리로 인하여 빠른 연산 처리가 가능해졌다.[4]

GPU의 내부는 다수의 처리장치가 동시에 동작하여 빠른 속도를 낼 수 있게 설계되었으며, 하나의 처리장치에서 실행되는 논리적인 프로그램의 단위를 thread라 부른다.[5]

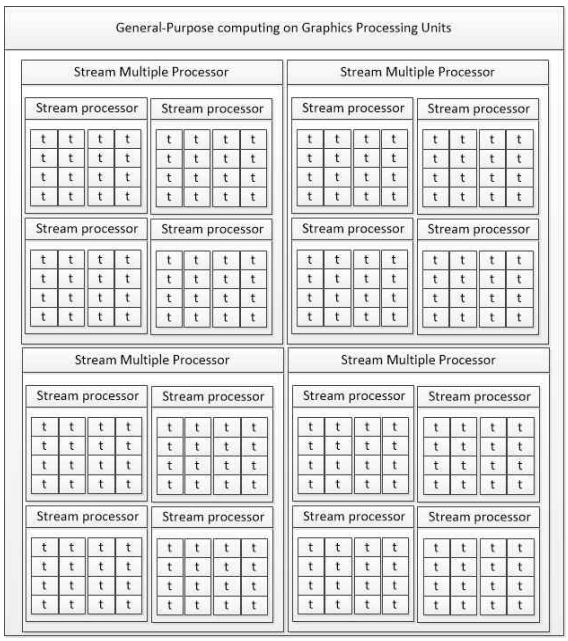

그림 1. GPGPU 구조
Figure 1. Structure of GPGPU

GPU가 많은 ALU를 이용해 연산의 병렬처리가 가능한 이유는 여러 개의 thread가 연산을 병렬로 처리하기 때문에 속도 측면에서 매우 뛰어난 성능을 발휘한다. 본 논문의 실험에서 사용한 GPGPU의 구조는 그림 1과 같다. GPGPU는 Stream Multiple Processor를 가지는데 본 논문의 실험에서 사용한 GPGPU의 경우 4개의 stream multiple processor를 가진다. 하나의 stream multiple processor는 4개의 stream processor로 구성되며 stream processor는 16개의 thread로 이루어져 있다. 따라서 실험에서는 256개의 thread의 stream multiple processor를 통하여 제어하며 이를 통하여 연산처리를 수행하였다. thread는 SIMD의 특징을 이용하여 각각의 data에 대하여 같은 연산을 각 thread마다 진행하므로 하나의 thread를 이용하여 연산처리를 수행하는 것보다. multi thread를 이용할 경우 보다 빠른 속도를 낼 수 있다.

GPGPU의 memory는 그림 2와 같이 세 부분으로 나눌 수 있다. processor에서 실행되는 바이너리가 저장되는 code memory와 thread에서 선언한 지역, 전역 변수들이 저장되는 thread memory 그리고 모든 thread가 접근할 수 있는 shared memory가 있다.

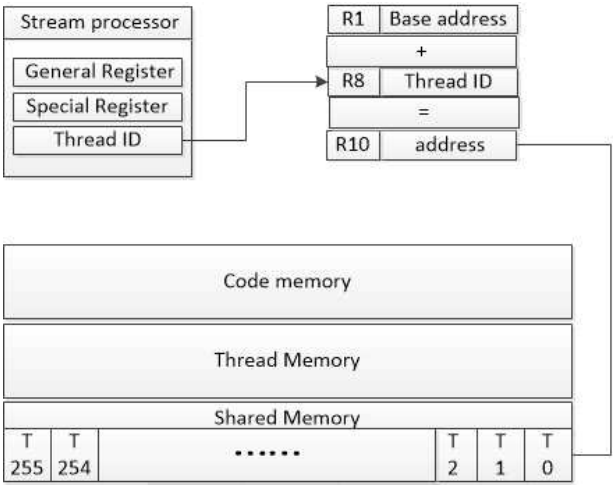

그림 2. stream processor의 shared Memory 접근방법과 메모리구조

Figure 2. Stream processor's shared memory approach and memory structure

그림 2처럼 stream processor이 가지고 있는 thread ID와 base address를 더하여 메모리에 대한 주소를 얻는다. 이 주소 값을 이용하여 어떤 thread에 접근할지 정할 수 있으며, 각 thread들은 shared memory를 통해 데이터를 PCIe driver를 이용해 Host PC와 공유할 수 있으며 이를 통하여 Read/Write작업을 수행하여 공동작업이 가능하다.

III. 제안하는 CNN 구조

기계학습이란 컴퓨터가 학습을 통하여 입력영상에 대하여 분석하는 것이다. 이는 매우 발전되어 왔으며 알고리즘에 따라서 결과물을 토대로 예측하는 것도 가능하다. 이를 이용하면 사람마다 다른 필기체 인식도 성공할 수 있다. 기계학습은 여러 종류가 있으며 본 논문에서는 CNN이라는 영상 인식과 추론에 특화된 학습 기법을 사용하며 이 구조는 많은 분야에서 적용이 가능 하다. CNN은 생물학적인 시각에서 뉴런과 시냅스들의 조합으로 사물을 인식하는 것이다. 뉴런이라는 신경세포가 정보를 저장하고, 시냅스들은 그 정보를 저장하고 이 정보가 뇌에 닿았을 경우 물체를 판단단하는 과정을 컴퓨터로 구현한 것으로 영상의 인식과 추론에 뛰어난 성능을 보이는 기계학습 구조이다. 이 구조는 많은 분야에서 적용이 가능하다. CNN은 convolution layer, pooling layer, connected layer로 구성되며, 입력영상의 픽셀들로부터 객체의 특징을 추출할 수 있게 학습된다. 이렇게 학습된 CNN의 파라미터들을 이용하면 입력영상에 대한 판독결과를 얻을 수 있다. [6] 또한 필요한 픽셀의 주변 픽셀만을 이용하여 연산이 가능하기 때문에 설계자가 원하는 크기의 커널 혹은 필터크기를 설정할 수 있다. convolution layer는 convolution연산을 통하여 특징들을 추출해주는 layer이며 일정한 패턴을 가지는 곱의 합으로 이루어진다. pooling layer는 convolution layer의 feature map을 줄여주는 역할을 한다. 마지막으로 fully connected layer는 여러 개의 뉴런으로 구성된 layer와 활성화 함수로 구성되어서 분류 또는 회귀 기능을 수행한다. WX+B의 간단한 연산으로 이루어져있지만 뉴런과 레이어 수가 많아짐에 따라 연산량이 증가하며 입력 이미지의 채널을 늘려 더욱 많은 파라미터들을 학습할 수 있으나 이 또한 연산량 증가를 가져온다. 따라서 이를 수행하는데 많은 시간을 필요로 한다. 그러나 이 문제는 GPU로 도움 받을 수 있다. GPU는 원래 고속의 그래픽 처리를 위해 사용되는 장치이지만, 다수의 병렬 처리 연산 장치들을 포함하고 있다.[7] 따라서 많지만 간단한 연산들로 이루어지는 CNN구조에 매우 적합하다. 본 논문에서는 그림 3과 같은 구조의 convolution layer와 pooling layer를 결합한 layer와 fully connected layer를 GPGPU를 이용하여 수행했다.

그림 3을 비교해보면 위의 CNN구조보다 본 논문에서 제안하는 아래의 CNN의 구조의 경우 더 간단한 구조를 가지고 있는 것을 확인할 수 있다. 첫 번째 convolution layer와 pooling layer를 합친 combined layer는 12x12의 8개의 feature map를 가지는 layer이며 두 번째 combined layer는 4x4개의 feature map을 가지는 layer이다. 나머지 fully connected layer는 각 층마다 256, 128, 64, 10개의 뉴런으로 이루어져 있다.

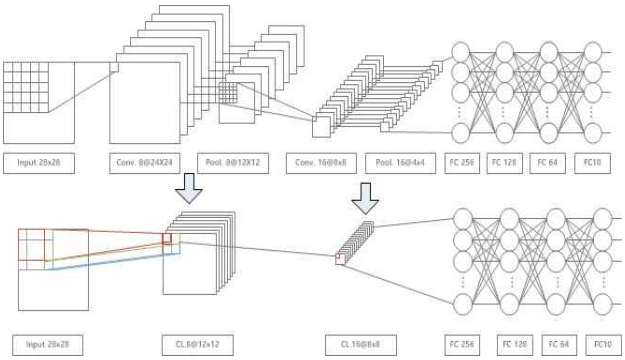

그림 3. 기존의 CNN구조와 제안한 CNN구조 비교
Figure 3. Comparison of existing CNN structure and proposed CNN structure

본 논문에서 제안하는 combined layer의 경우 convolution layer의 연산 결과를 메모리에 저장하지 않고 바로 polling layer의 입력으로 사용한다. 결과적으로 메모리에 저장하는 횟수가 감소함으로써 메모리 접근이 줄어들기 때문에 처리속도가 줄어들며, thread하나가 combined layer에서 4개의 convolution 연산과 1개의 pooling연산을 수행한다. 하지만 fully connected layer에서는 여러 개의 thread가 동일한 데이터를 저장할 경우 문제가 발생하기 때문에 이를 막기 위해 각 thread당 시간 할당 등의 제어가 필요하며, fully connected layer에서 하나의 thread는 하나의 뉴런을 위한 곱의 합 과정을 수행하며 활성화 함수를 거친다. 이러한 과정들을 통하여 CNN의 Feedforward 과정이 진행된다.

Ⅳ. 실험 및 결과

표 1. 기존의 CNN구조의 처리시간과 제안한 구조의 CNN 처리시간 비교
Table 1. Comparison of the processing time of the existing CNN structure and the CNN processing time of the proposed structure

	Processing Time
Existing CNN Structure	12.28ms
Proposed CNN structure	10.76ms

실험에는 MNIST라는 손 글씨 데이터셋을 사용하였다. 이를 Xilnix사의 VC707 FPGA보드를 이용하여 256개의 Multi thread로 동작하는 GPGPU기술을 이용하여 CNN의 feedfoward 연산처리과정을 수행하였다. 그림 3의 CNN구조의 feed forward과정을 수행하였을 때 표1과 같은 결과를 보였다. 제안한 구조의 CNN은 기존의 CNN구조 보다 12.38.% 빠른 속도를 보여 제안하는 구조가 feed forward 과정에서 기존의 구조보다 더 좋은 성능을 보이는 것을 확인하였다.

Ⅴ. 결 론

최근 딥러닝은 개발 빅데이터의 활성화, 하드웨어의 발전에 힘입어 처리가능한 문제의 범위가 급속도로 넓어지면서 여러 가지 분야에 접목되고 있는 중이다. 특히 영상처리 분야의 경우 영상의 인시고가 판독에 한계가 분명했던 분야 등 예전에는 기계학습이 많이 사용되지 않았던 분야라 하더라도 기계학습을 도입하려는 움직임이 나타났고, 딥러닝은 매우 범용적인 기술이 되었다. 또한 많은 연구가 진행되면서 다양한 알고리즘이 나오고 있다. 이는 목적에 따라 학습 구조를 사용하느냐에 따라, 알고리즘 개선을 통한 성능개선의 여지가 있으며, 본 연구에서 보이듯 각 layer에서 사용되는 연산을 GPU에서 수행하고 combined layer라는 구조를 사용함으로써 성능이 개선됨을 확인하였다. 이를 통해 연산이 많이 필요한 복잡한 알고리즘이라 하더라도 GPGPU기술 등 하드웨어 성능에 따라 연산처리시간이 가속화될 가능성이 많음을 알 수 있다. 따라서 앞으로도 많은 연구가 필요하다.

References

[1] Lee Wan-Joo, "A Study on Implementation of Vehicle Number Plate Recognition Moduler" BUll Nat. Sci, Yong-In Univ, vol.8, no.2, pp.153-159, Feb. 2004.
[2] KimYeonGyu, "Improvement of Korea characters Recognition Performance Using CNN and Feature Extraction", Master thesis, Busan University, 2017
[3] Lee Seonguk, Byeon Gibeom, Kim Kisu, Hong Jiman, "Research of accelerating method of video quality measurement program using GPGPU", Smart Media Journal, vol.5, no4, pp. 66-74, Dec. 2016.
[4] Jeon Sanghui ,Development of multi-pedestrain gender classification algorithm using GPGPGU, Master thesis, KyungSung University, 2017.
[5] Heewon Kye and Junho Kim, "Acceleration techniques for GPGPGU-based Maximum Intensity Projection", Journal of Korea Multimedia Society, vol.1, no.8, pp981-991, 2011.
[6] Ye, Jaehyung, "Scale-invariant CNN for building and road detection from aerial images", Master thesis, Korea University, 2017.
[7] John Nickolls, William J Dally, "The GPU Computing Era", IEEE, vol.30, no.2, pp. 59-69, March-Aprill, 2010.

※ This work was supported by Seokyeong University in 2017