Multi-GPU 환경에서의 Convolution Layer 최적화 실험

하지원 [1], 테오도라 아두푸 [2], 김윤희 [2]
[1] 고려대학교 컴퓨터학과
[2] 숙명여자대학교 컴퓨터과학과

jwh0245@naver.com, theoadufu@sookmyung.ac.kr, yulan@sookmyung.ac.kr

Empirical Experiments for Convolution Layer Optimization on Multi-GPUs

Jiwon Ha[1], Theodora Adufu[2], Yoonhee Kim[2]
[1]Dept. of Computer Science and Engineering, Korea University
[2]Dept. of Computer Science, Sookmyung Women's University

요 약

GPGPU 환경에서의 ML 모델이 다양한 분야에 지속적으로 활용되면서, 이미지 분할(image segmentation) 연구가 활발하다. multi-GPU 환경에서 성능 최적화를 위하여 병렬화 기법들이 활용되고 있다. 본 연구에서는 multi-GPU 환경에서 U-Net 모델의 전체 수행 시간을 단축하기 위해 convolution 연산을 최적화하는 기법을 적용하는 실험을 진행하였고 shared memory, data parallelism 를 적용하여 82% 성능 향상을 보여주었다.

1. 서론

CPU 를 사용한 모델 학습은 학습 데이터의 크기가 클 경우 지나치게 많은 시간이 소모되어 실험을 진행하는 데에 현실적인 어려움이 있다. 특히 이미지 데이터의 경우, 데이터의 크기와 개수가 늘어날수록 학습에 필요한 연산이 크게 증가한다. Yadan et al. [1]의 "Multi-GPU Training of ConvNets"에 따르면, CPU 환경과 비교하여 multi-GPU 환경에서 CNN (Convolutional Neural Network) 모델에 data parallelism 와 model parallelism 를 적용하였을 때, 2 GPUs model parallelism 의 경우 약 37%, 2 GPUs data parallelism 의 경우 약 33%, 4 GPUs model parallelism, data parallelism 를 병행하는 경우 약 54%의 speedup 이 가능하다.

GPGPU(General-Purpose computing on Graphics Processing Units) 환경에서의 ML(Machine Learning) 모델이 다양한 분야에 지속적으로 활용되면서[2], 이미지로부터 객체를 추출하여 특정 클래스로 분류하는 이미지 분할(image segmentation) 연구가 활발하다. 의료 영상 분석, 자동차 자율주행 등 ML 모델이 사용되는 여러 task 들에서 물체를 구분하는 기능이 우선 수행되어야 하기 때문이다. 이미지 분할 모델 중에서도 2015 년 발표된 U-Net 모델은[3] 현재까지도 많은 모델의 base line 이 될 정도로 이미지 분할에서 우수한 성능을 보여준다.

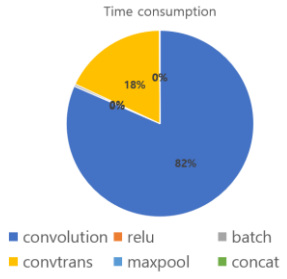
본 연구에서는 multi-GPU 환경에서 U-Net 모델 전체의 수행 시간을 단축하기 위해 convolution 연산을 최적화하는 기법을 설명한다. 2 장에서는 실험의 구조, 3 장에서는 실험 결과를 설명하고 4 장에서 결론을 맺는다.

2. 실험 내용

U-Net 은[4] 크게 Encoder, Decoder 2 가지 요소로 구성된다. 두 단계 모두 convolution 연산이 반복적으로 수행된다. convolution 연산은 Relu, batch normalization 등 다른 연산에 비해 계산 과정이 복잡하다.

각 함수 별 수행 시간 (초)

Convolution	16.688559
Relu	0.004277
Batch normalization	0.074462
Transposed convolution	3.669185
Maxpool	0.001810
Concat	0.016511



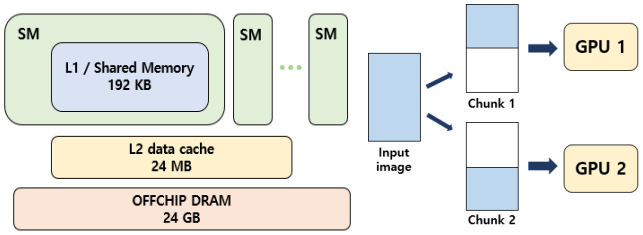
(표 1) 함수 수행 시간 (그림 1) 함수 수행 시간 비율

표 1, 그림 1 은 NVIDIA A30 머신 CPU(Intel® Xeon® Silver 4310 CPU @ 2.10GHz, 2699.998MHz, 12 CPU cores)에서 U-Net 으로 128*191 해상도의 3 channels 이미지 1 개를 처리하는 데에 함수 별 수행 시간을 정리한 것이다. U-Net 의 수행 시간의 82%가

convolution 수행에 소요되는 것을 확인할 수 있다. 실제로 convolution 연산은 VGG, ResNet 과 같은 CNN 모델 수행 시간의 70% 이상을 차지한다[5], [6]. 따라서 모델 전체의 성능 향상을 위해 convolution 연산의 최적화가 중요하다.

Convolution 최적화를 위해 실험에 사용할 최적화 기법은 크게 2 가지다. 첫 번째는 GPU 의 메모리 구조 중 shared memory 를 이용한 tiling 기법이다. GPU 의 SM(Streaming Multiprocessor)은 SM 내부에 고유한 shared memory 를 가진다(그림 2 의 shared memory block 참조). shared memory 로의 접근은 off-chip 메모리인 글로벌 메모리에 접근하는 것보다 약 100 배 빠르므로 최적화에 사용할 수 있다. 실험에서는 convolution 연산에 사용되는 input 과 filter weight 를 shared memory 에 올린다.

두 번째 최적화 기법은 multi-GPUs, multi-streams 을 이용한 data parallelism 이다. 그림 3 은 전체 데이터를 2 개의 chunk 로 분할한 data parallelism 예시이다. convolution 연산의 경우 각 데이터 간의 의존성(dependency)이 존재하지 않으므로 데이터 분할이 자유롭다.



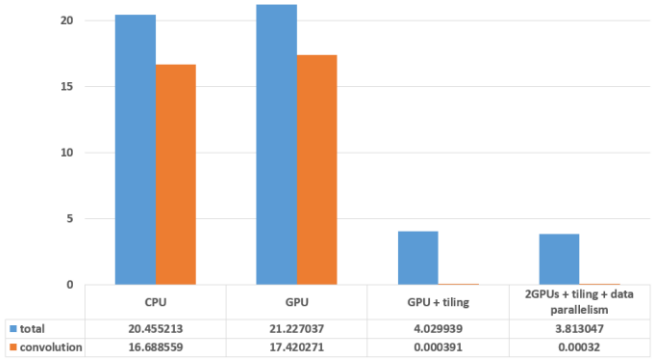
(그림 2) GPU 메모리 구조[2]

(그림 3) data parallelism

3. 실험 결과

학습 데이터는 128*191 해상도의 3 channels 이미지 데이터를 사용한다. 우선 최적화하지 않은 기본 모델을 각각 CPU, NVIDIA A30 머신(Compute capability 8.0, Device memory 24GB) GPU 에서 학습시킨다. 이어서 tiling 기법을 적용한 모델을 1 GPU 에서 학습시키고 tiling 과 data parallelism 기법을 적용한 모델을 2 GPUs 환경에서 학습시킨다.

실험 결과, 수행 시간은 그림 4 와 같다. 최적화하지 않은 모델을 GPU 에서 훈련시켰을 때 convolution 연산에 소요되는 시간은 CPU 에 비해 약 0.74 초 증가하였다. CPU 에서 GPU 로 데이터를 transfer 하는 데에 소요되는 시간이 영향을 미치기 때문으로 추정된다. 최적화한 모델에서 convolution 연산에 소요되는 시간은 CPU 16.6 초에서 1 GPU 0.00039 초, 2 GPUs 0.00032 초로 크게 감소하였다. U-Net 의 전체 학습 시간 또한 20.4 초에서 1 GPU 4.0 초로 약 80%, 2 GPUs 3.8 초로 약 81% 속도가 향상되었다. 최적화한 모델이 1 GPU 와 2 GPUs 에서 큰 성능 차이가 나지 않는 이유는 데이터셋의 크기가 충분히 크지 않기 때문으로 추정된다.



(그림 4) 실행 환경에 따른 실행 시간 비교

4. 결론

Multi GPUs 환경에서 convolution 연산을 최적화하는 기법(shared memory, data parallelism)을 통하여 U-Net 의 성능을 향상하였다. 다른 최적화 기법들을 추가로 적용하여 convolution layer 를 포함하는 VGG, ResNet 등 여러 CNN 모델의 수행 시간 성능 향상을 실험할 예정이다.

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(No. 2021R1A2C1003379)과 2023 년도 정부재원(과학기술정보통신부 여대학원생 공학연구팀제 지원사업)으로 과학기술정보통신부와 한국여성과학기술인육성재단의 지원을 받아 연구되었습니다.

참고문헌

[1] Omry Yadan et al., “Multi-GPU Training of ConvNets”, arXiv:1312.5853, pp 1, 2014

[2] Jieun Kim et al., “Empirical experiments of profiled data locality for memory-divergent workloads on GPU”, KNOM Review Vol.25, No.01, Jan 2022

[3] Olaf Ronneberger et al., "U-Net: Convolutional Networks for Biomedical Image Segmentation", MICCAI 2015, pp 234-241, May 2015

[4] Kyung-Min Gwak et al., “Tracking Method of Dynamic Smoke based on U-net”, IIBC Vol. 21, No.4, pp 81-87, Aug 31, 2021

[5] X. Li, et al. “Performance Analysis of GPU-Based Convolutional Neural Networks,” 2016 45th International Conference on Parallel Processing (ICPP), Philadelphia, PA, pp. 67-76, 2016

[6] Xu QQ, An H, Wu Z, Jin X. “Hardware Design and Performance Analysis of Mainstream Convolutional Neural Networks”. Computer Systems and Applications, 29(2): 49-57(in Chinese), 2020