모바일 장치에서 캐시 및 메모리 접근이 CNN 성능에 미치는 영향 분석

조근혜 O1, 홍경환 2, 신동군 1
1 성균관대학교 소프트웨어학과, 2 성균관대학교 전자전기컴퓨터공학과
{shurui, redc7328, dongkun}@skku.edu

Analysis on the Memory Access Impact of CNN on Mobile Devices

Geunhye Jo O1, Gyeonghwan Hong2, Dongkun Shin1
1Department of Software, Sungkyunkwan University
2Department of Electrical and Computer Engineering, Sungkyunkwan University

요 약

CNN은 높은 정확도를 얻기 위해서 층이 깊은 구조로 발전해왔다. CNN의 심층화는 연산량과 메모리 사용량을 크게 증가시켜 모바일과 같은 저성능 환경에서 추론 속도가 매우 저하되는 문제를 일으켰다. 이 문제를 해결하기 위해, MobileNet 등 모바일 환경에서도 연산량을 줄이면서 정확도 손실을 최소화하도록 구조를 개선한 CNN 모델이 제안되었다. 기존 연구에서는 부동소수점 연산량(FLOPs)을 줄이는 데 초점을 맞추었으나, FLOPs의 감소가 추론 시간 감소로 직결되지 않아 다른 요인이 있음을 시사하였다. 본 연구에서는 모바일 GPU에서 기존 CNN은 연산량 뿐만 아니라 메모리 접근량도 추론 시간을 결정함을 알아냈다. 이를 증명하기 위해, VGG, MobileNet, ResNet, GoogLeNet 등 널리 사용되는 CNN의 Width Multiplier를 조절하며, FLOPs, 메모리 접근 횟수, 그리고 캐시 접근 횟수 등이 추론 시간에 미치는 영향을 분석하였다.

1. 서 론

CNN(Convolutional Neural Network)은 Convolutional Layer를 주축으로 구성된 딥 뉴럴 네트워크의 일종이다. CNN은 주로 이미지 분류, 자율 주행 등에 응용되고 있으며, 이미지 분류 정확도 (Accuracy)가 모델 성능의 척도로 사용되고 있다. 따라서 정확도를 높이기 위해 신경망 모델들은 층이 깊어지고 학습되는 파라미터의 개수가 증가하는 방향으로 발전하였다.

CNN을 정확도에만 초점을 맞춰 모델의 규모를 크게 발전시키면 모델 복잡도가 높아지고 불필요한 연산이 늘어난다. 이에 따라 저사양의 임베디드 환경에서는 프로세싱 유닛의 성능이 일반 환경보다 떨어지기 때문에 유의미한 사용이 불가할 만큼 실행 속도가 느려진다. 따라서 CNN모델의 메모리 사용량과 실행 시간은 줄이면서 정확도는 최대한 유지하는 연구들이 등장했다.

CNN의 모델 규모를 줄이고 실행 속도를 증가시키는 연구는 모델 압축, 경량화 모델로 나뉜다. 모델 압축은 구조가 이미 정해진 모델을 변형하여 파라미터의 수나 bitwidth를 줄이는 방법이다. 모델 압축 연구에는 Quantization, Pruning[1], Tensor Decomposition 등이 있다. 경량화 모델은 신경망 구조를 메모리와 연산을 효율적으로 사용하도록 설계하는 방법이다. Convolutional Layer를 Bottleneck Module, Depthwise Separable Convolution[3], Group Convolution 등으로 대체하여 메모리와 연산량을 줄이는 연구가 이루어지고 있다.

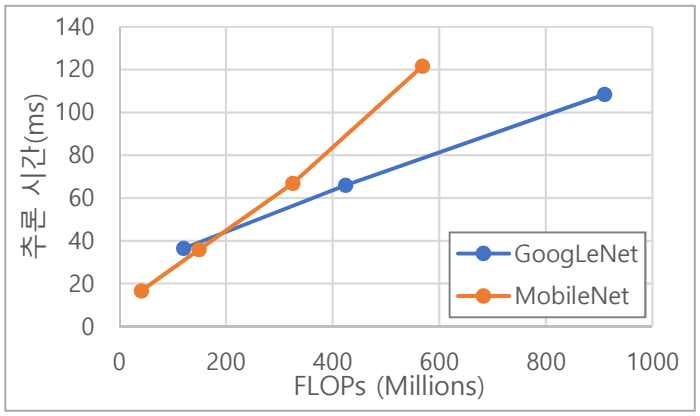

그림 1. GoogLeNet[2]과 MobileNet[3]에 Width Multiplier를 적용하여 측정한 결과. GoogLeNet은 아래쪽부터 {0.25, 0.5}를, MobileNet은 아래쪽부터 {0.25, 0.5, 0.75}를 적용하였다.

기존 연구에서는 주로 FLOPs(부동소수점 연산량) 감소에 주력하였으나, 실제 모바일 장치에서는 추론 시간은 다른 양상을 나타낸다. 모바일 AP를 탑재한 Odroid-XU4에서 실험한 결과, FLOPs는 추론 시간에 양의 상관관계를 보이지만 FLOPs의 감소율과 추론 시간의 감소율에는 차이가 크다는 점을 발견했다. 그림 1에서는 GoogLeNet-0.5와 MobileNet-

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW컴퓨팅산업원천기술개발사업(SW스타랩)의 연구결과로 수행되었음 (IITP-2017-0-00914)

0.75는 추론 시간이 유사하지만 FLOPs는 GoogLeNet-0.5가 99M 크다. GoogLeNet-0.75와 MobileNet-1.0에서는 차이가 더 크게 벌어져, MobileNet-1.0에서 시간은 14ms 더 오래 걸리지만 FLOPs는 GoogLeNet-0.5가 342M 더 크다.

이런 현상은 N. Ma 등[4]이 언급하였듯이 FLOPs 외에도 메모리 접근 비용이나 병렬 처리 수준 등 실행 속도에 영향을 미치는 다른 요인이 있기 때문이다. 이들 요인은 같은 모델에 대해서도 하드웨어 또는 소프트웨어 플랫폼에 따라 다른 양상을 보인다.

본 논문에서는 모바일 GPU에서 다양한 구조의 CNN 모델을 동작할 때 캐시 접근 횟수, 메모리 접근 횟수가 CNN의 추론시간에 미치는 영향을 실험을 통해 비교한다. 기존 CNN 모델로는 VGG[5], MobileNet, ResNet[6]에 다양한 Width Multiplier를 적용하여 각 Layer의 채널 수를 다양하게 변경한 네트워크들을 사용하였다. 해당 실험은 모바일 GPU이 탑재된 Odroid-XU4 상에서 임베디드 장치용 딥 러닝 플랫폼인 ARM Compute Library를 활용하여 진행하였다.

2. 관련 연구

ShuffleNet V2[4]에서 N. Ma 등은 대부분의 신경망 구조 설계가 연산 복잡도의 간접적 척도인 FLOPs 등에 의해 이끌어지는 점을 다음과 같이 비판한다. 연산 복잡도의 직접적 척도인 추론 속도 등은 메모리 접근 비용, 플랫폼 특성 등 여타 요인들에도 좌우되기 때문에 FLOPs는 불충분한 척도이고, FLOPs만 고려하여서는 최적인 아닌 네트워크 구조를 설계하는 방향으로 유도되기 쉽다.

N. Ma 등은 효율적인 네트워크 디자인을 위해서 직접적 척도를 사용하고, 그 척도는 그 네트워크를 사용할 플랫폼 상에서 측정해야 한다는 두 가지 원칙을 제시하였다. 이에 따라, ShuffleNet V2에서는 초당 이미지 처리 속도라는 직접적 척도를 서버급 GPU인 NVIDIA GeForce GTX 1080Ti와 ARM의 모바일 CPU 상에서 측정하였다.

반면, 본 논문은 직접적 척도인 추론 시간에 주요한 영향을 주는 간접적 척도를 파악하기 위해 추론 시간, FLOPs, 메모리 접근 횟수, 캐시 접근 횟수 등을 측정하였다. 또한 임베디드 장치 상에서 CNN 모델을 하드웨어적으로 가속할 수 있는 모바일 GPU인 ARM Mali GPU 상에서 실험하여, 모바일 GPU의 메모리 구조와 딥 러닝 모델의 동작 간 관계를 규명하고자 하였다.

3. 실험 구성

3.1 실험 환경

ARM Mali-T628 MP6 가 장착된 Odroid XU4 보드에서 실험을 진행하였다. Ubuntu 16.04 환경에서 ARM CPU 와 GPU 에 최적화된 딥러닝 라이브러리 ArmCL 18.08 을 사용하였다.

ARM Mali-T628 MP6 는 Shader Core 6 개를 보유하며, 각 Shader Core 는 16KB 의 L1 Cache 를 갖추고 있다. L2 Cache 는 Shader Core 당 32KB 의 용량을 가지므로 총 192KB 의 용량을 갖는다. Mobile GPU 는 고성능 GPU 와 다르게 CPU 와 별도로 가지는 DRAM 이 없고, 외부의 메모리를 CPU 와 공유한다. 따라서 일반 GPU 처럼 파라미터를 CPU Memory 로 로드한 다음 GPU Memory 로 전송하는 시간이 사라지는 반면, CPU 와 함께 메모리를 공유하기 때문에 Contention 이나 Snooping 이 발생할 수도 있다.

3.2 CNN 모델

비교 모델으로는 VGG-16, ResNet-50, MobileNet 을 사용하였다. VGG 는 기본적인 3×3 Convolution 으로만 레이어를 쌓은 모델이다. ResNet 은 Bottleneck 모듈과 Shortcut Connection 이 특징인 네트워크이다. Bottleneck 모듈은 3×3 layer 에서 연산 될 채널 수를 그 앞의 1×1 layer 에서 줄여주고 그 뒤의 1×1 layer 에서 복구하는 형태로 네트워크를 구성하는 기본 블록이다. Shortcut Connection 은 모듈의 연산을 거쳐 나온 결과물에 모듈 이전의 피처맵을 더하는 방법이다. MobileNet 은 채널 내 연산을 Depthwise 레이어로, 채널 간 연산을 Pointwise(Separable) 레이어로 분리하여 파라미터 수와 연산량을 크게 줄인 Depthwise Separable Convolution 으로 구성된 모델이다.

입력 이미지의 크기는 224×224 를 사용하였다. 또한, 모델의 규모에 따른 특성을 규명하고, 모델의 구조에 따른 특성은 모델의 규모와는 무관함을 증명하기 위해 MobileNet 에서 제안된 Width Multiplier 를 다음과 같이 적용하였다.

- VGG-16: 0.125 간격으로 0.125 부터 1.0 까지
- ResNet-50: 0.125 간격으로 0.125 부터 1.5 까지
- MobileNet: 0.25 간격으로 0.25 부터 3.0 까지

4. 실험 결과

그림 2 를 보면, 한 모델에서 Width Multiplier 를 늘려감에 따라 FLOPs 와 추론 시간이 함께 증가한다. 그러나 추론 시간이 증가하는 양상은 세 모델에서 각각 다르게 나타났다. VGG 는 FLOPs 를 증가시킴에 따라 정비례에 가깝게 추론 시간이 증가하였지만 ResNet 과 MobileNet 은 그렇지 않다.

이들 모델에서 FLOPs 외의 어떤 요인이 추론 시간에 밀접한 영향을 가지고 있는지 알기 위하여 각각 Parameter 수, L1 캐시 접근 횟수, 메모리 접근 횟수를 FLOPs 에 대하여 나타내어 비교하였다. 메모리 접근 횟수는 L2 캐시 미스 횟수를 측정하여 이와 동일하다고 가정하였다.

Parameter 수는 MobileNet 이 VGG 의 약 4 분의 1 이고 ResNet 은 VGG 의 절반 정도로 나타나 추론 시간과 전혀 다른 경향을 보였다. L1 캐시 접근 횟수와 추론 시간은 양의 상관관계를 보이나 MobileNet 보다 ResNet 에서 대체적으로 더 높은 수치를 가져 해당 두 모델간 추론 시간 차이를 설명하지 못한다.

그림 3 에 나타낸 메모리 접근량이 그림 2 의 추론 시간과 가장 유사한 증가 추세를 보인다. 또한 다른 지표들에서는 나타나지 않았던 MobileNet-3.0 에서 ResNet-1.5 보다 높은 수치를 보여주므로 결정적인 요인임을 확인할 수 있었다.

5. 분 석

MobileNet 과 ResNet 이 VGG 에 비해서 FLOPs 는 낮으면서 추론 시간은 오래 걸리는 현상은 Depthwise 레이어와

Pointwise 레이어의 Arithmetic Intensity 가 낮기 때문에 발생한다. Arithmetic Intensity 는 하나의 Layer 에서 발생하는 FLOPs 를 해당 연산을 위해 접근해야 하는 메모리의 양으로 나눈 것이다. Arithmetic Intensity 가 낮으면 FLOPs 가 떨어져도 메모리 접근은 높아 추론 시간이 길다. 표 1 에서 보듯이, Depthwise Convolution 의 경우, 일반 Convolution 에 비하여 메모리 접근 중 작은 양을 차지하는 필터만 M 분의 1 으로 줄어드는 반면, 연산량은 전체가 M 분의 1 로 줄어 Arithmetic Intensity 가 작아진다. Pointwise Convolution, 즉 1×1 Convolution 또한 메모리 접근량에서는 필터 부분만 D_K^2 분의 1 으로 줄고 연산량은 전체가 D_K^2 분의 1 으로 줄어 Arithmetic Intensity 가 악화된다.

ResNet 은 또한 Shortcut Connection 기법이 적용되어 있어 메모리 접근량이 높다. Shortcut Connection 은 Bottleneck 모듈의 연산을 마친 이후, 모듈 연산 이전의 피처맵에 다시 접근하여야 하기 때문이다.

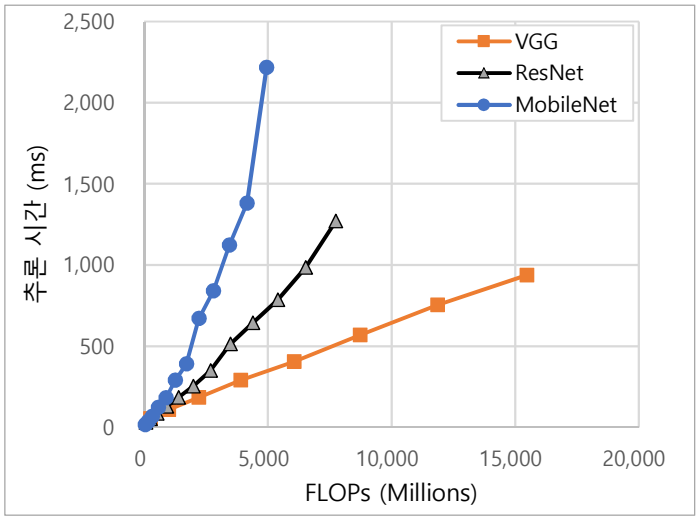

그림 2. 전체 네트워크에서 측정한 FLOPs-추론시간 그래프

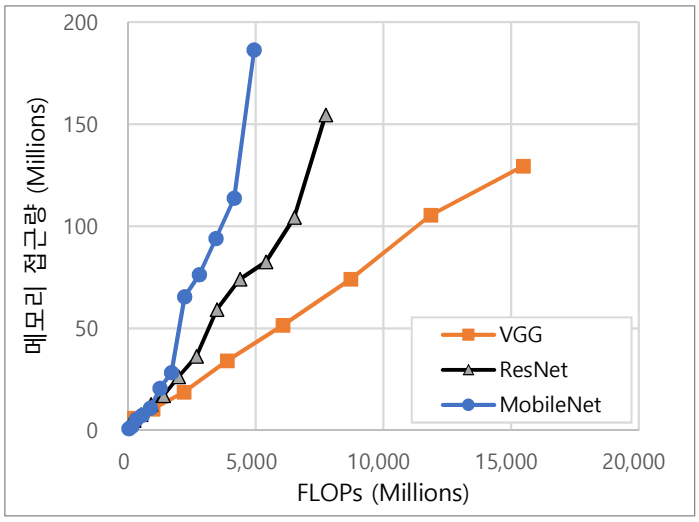

그림 3. 전체 네트워크에서 측정한 FLOPs-메모리 접근량그래프

표 1. Convolution 종류에 따른 FLOPs와 메모리 접근량.

D_F: Feature Map의 한 변의 길이. M: Input 및 Output 채널 수. D_K: Convolution Filter의 한 변의 길이

일반 Convolution	FLOPs	$D_F^2 \times M^2 \times D_K^2$
	메모리 접근	$D_F^2 \times 2M + D_K^2 \times M^2$
Depthwise Convolution	FLOPs	$D_F^2 \times M \times D_K^2$
	메모리 접근	$D_F^2 \times 2M + D_K^2 \times M$
Pointwise Convolution	FLOPs	$D_F^2 \times M^2$
	메모리 접근	$D_F^2 \times 2M + M^2$

6. 결 론

본 논문에서는 CNN 의 부동소수점 연산량 만으로는 실제 추론 시간의 변화를 충분히 설명할 수 없음을 지적하고, 추론 시간에 영향을 미치는 캐시 및 메모리 관련 요인을 찾는 실험을 진행하였다. VGG 와 ResNet, MobileNet 모델을 사용하고 또한 Width Multiplier 를 적용하여 서로 다른 구조와 규모의 네트워크에서 부동소수점 연산량과 파라미터 수, 캐시 및 메모리 접근 횟수와 실행 시간을 비교하였다. 비교 결과, FLOPs 와 함께 메모리 접근 횟수의 증가가 추론 시간을 결정함을 알아냈다. 따라서 모바일 GPU 환경에서 효율적으로 작동하는 모델을 선택할 때에는 FLOPs 과 메모리 접근량을 함께 고려하여야 한다.

본 연구의 결과는 신경망의 시간적 복잡도를 추정할 때 사용하는 간접 측정 수단(부동소수점 연산량, 메모리 접근 횟수 등)과 직접 측정 수단(실행 시간, 처리 속도 등)에 대한 연구에 참고가 될 것이다. 더 나아가 저성능 컴퓨팅 환경에서 효율적으로 작동하는 신경망을 설계하는 데에 도움을 줄 것으로 기대된다.

참고 문헌

[1] W. WEN, et al. “Learning structured sparsity in deep neural networks,” In: Advances in Neural Information Processing Systems, p. 2074-2082, 2016.

[2] Szegedy, Christian, et al., "Going deeper with convolutions," Proceedings of the IEEE conference on computer vision and pattern recognition, 2015.

[3] A. G. Howard, et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[4] N. Ma, et al. "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design," arXiv preprint arXiv:1807.11164, 2018.

[5] Simonyan, et al. "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[6] K. He, et al. “Deep residual learning for image recognition,” In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770-778, 2016