ISSN : 1226-7244 (Print)
ISSN : 2288-243X (Online)
논문번호 20-03-02

j.inst.Korean.electr.electron.eng.Vol.24,No.3,685~690,September 2020
http://dx.doi.org/10.7471/ikeee.2020.24.3.685
9

저연산량의 효율적인 콘볼루션 신경망

Efficient Convolutional Neural Network with low Complexity

이 찬 호*★, 이 중 경*, 호 콩 안*

Chanho Lee*★, Joongkyung Lee*, Cong Ahn Ho*

Abstract

We propose an efficient convolutional neural network with much lower computational complexity and higher accuracy based on MobileNet V2 for mobile or edge devices. The proposed network consists of bottleneck layers with larger expansion factors and adjusted number of channels, and excludes a few layers, and therefore, the computational complexity is reduced by half. The performance the proposed network is verified by measuring the accuracy and execution times by CPU and GPU using ImageNet100 dataset. In addition, the execution time on GPU depends on the CNN architecture.

요 약

휴대용 기기나 에지 단말을 위한 CNN인 MobileNet V2를 기반으로 연산량을 크게 줄이면서도 정확도는 증가시킨 효율적인 인공신경망 네트워크 구조를 제안한다. 제안하는 구조는 Bottleneck 층 구조를 유지하면서 확장 계수를 증가시키고 일부 층을 제거하는 등의 변화를 통해 연산량을 절반 이하로 줄였다. 설계한 네트워크는 ImageNet100 데이터셋을 이용하여 분류 정확도와 CPU 및 GPU에서의 연산 시간을 측정하여 그 성능을 검증 하였다. 또한, 현재 딥러닝 가속기로 널리 이용하는 GPU에서 네트워크 구조에 따라 동작 성능이 달라짐도 보였다.

Key words : MobileNet, CNN, GPU, computation complexity, Accuracy

Ⅰ. 서론

최근 인공신경망에 대한 연구가 활발히 진행되고 있으며 특히 CNN(Convolutional Neural Network)은 영상인식 분야를 포함하여 다양한 분야에서 활용도가 높아 가장 널리 이용되고 있다. CNN은 초기에는 VGG, Inception, ResNet, Xception, DenseNet, SENet 등과 같이 정확도를 높이기 위해 연산량을 증가시킨 구조들이 발표되었으나[1-6] 최근에는 연산 능력이 부족한 휴대용 또는 에지 단말기에서도 추론이 가능한 가벼운 구조의 MobileNet이나 ShuffleNet 등이 발표되었다[7, 8]. MobileNet V2는 Xception에서 사용된 Depthwise-separable convolution (DSC)과 inverted residual block(IRB)을 이용하여 연산

* Dept. of Information and telecommunications Engineering, Soongsil University
★ Corresponding author
E-mail : chlee@ssu.ac.kr, Tel : +82-2-825-8108

※ Acknowledgment
This work was supported by the MOTIE (Ministry of Trade, Industry & Energy (10080568) and KSRC(Korea Semiconductor Research Consortium) support program for the development of the future semiconductor device, and by the National Research Foundation of Korea (NRF) grant (NRF-2016R1D1A1B01008846). This research was results of a study on the "HPC Support" Project, supported by the 'Ministry of Science and ICT' and NIPA.
Manuscript received May. 31, 2020; revised Aug. 11, 2020; accepted Aug. 18, 2020.

량을 획기적으로 줄이면서도 정확도를 VGG16 수준으로 유지하였다[9]. ShuffleNet은 group convolution과 shuffling을 이용하여 연산량을 MobileNet의 절반 정도로 줄였으나 분류 정확도가 상당히 감소하였으며 group convolution의 한계로 만족할 만한 정확도를 얻기 어렵다. 그런데 ShuffleNet의 저자들이 CNN 설계와 관련하여 비슷한 연산량을 갖는 네트워크에서 층간의 채널비 수의 비를 작게 유지하고 네트워크의 층의 수를 줄일 때 실제 연산 시간을 줄일 수 있다는 연구 결과를 발표하였다.

본 논문에서는 MobileNet V2에 기반하여 연산량을 줄이면서도 정확도를 유지하는 CNN 구조를 제안한다. MobileNet V2의 병목층(Bottleneck)의 일부를 제거하고 출력 채널의 수를 조절하여 연산량을 감소시키고 확장 계수를 증가시켜 정확도를 유지하거나 증가시켰다. 제안한 CNN 구조는 CPU와 GPU를 이용한 연산 시간을 측정하여 연산량과 실제 연산 시간을 비교하였다.

II. 제안하는 네트워크 구조

1. MobileNet V2

MobileNet V2는 MobileNet의 특징과 함께 선형 병목층(linear bottleneck), ReLU6, IRB를 가지고 있다. 선형 병목층은 입력 채널을 확장하여 DSC을 진행함으로써 채널을 증가시키면서도 연산량을 줄이고 이를 압축하여 활성화 함수를 거치지 않고 출력하는 방식으로 출력값이 커지는 것을 방지하기 위해 ReLU6를 확장층과 DSC에 적용한다. IRB는 압축된 피쳐맵(feature map)에 대해 skip connection을 이용한 덧셈을 수행하여 입력 피쳐맵을 저장하는 공간을 줄일 수 있게 한다. MobileNet V2는 입력단의 3×3 2D convolution, 7개의 병목층, 그리고 1×1 convolution으로 이루어진 출력층으로 구성된다[9]. 병목층에서는 6배 확장과 압축을 수행하고 각 층마다 조금씩 다른 구성을 갖는다.

2. 제안하는 Simplified MobileNet V2

Simplified MobileNet V2(SM_V2)에서는 병목층의 확장 계수를 그 위치에 따라 줄이거나 늘려 연산량을 줄이면서 정확도를 최대한 유지하였다. 또한, 일부 병목층의 출력 채널의 수를 조금씩 감소시키고 내부의 반복층의 수를 2개로 제한하였으며 다섯 번째 병목 층을 제거하였다. 이는 ShuffleNet의 설계 방법에서 일부 영감을 얻은 것으로 전체적인 병목층의 수를 줄이면서 확장 계수를 조절하여 분류 정확도를 최소한으로 훼손하는 범위에서 연산량을 줄였다. 표 1에서 MobileNet V2와 SM_V2의 구조를 비교하였다. 표에서 t는 확장 계수, c는 출력 채널수, n은 bottleneck 내부의 층의 수, s는 stride이다. 굵은 글씨체는 각 네트워크 사이의 차이를 나타낸다. SM_V2_2_2(1.5)와 SM_V2_x2nr은 SM_V2의 변형 형태로 다음 절에서 설명한다.

SM_V2의 첫 번째 병목층은 가장 기본이 되는

Table 1. Network architecture of MobileNet and simplified MobileNet.
표 1. MobileNet와 simplified MobileNet의 구조

Operator/t/c/n/s			
MobileNet V2	SM_ V2	SM_V2_2_2(1.5)	SM_V2_x2nr
conv2D_3x3/-/32/1/2	conv2D_3x3/-/32/1/2	conv2D_3x3/-/32/1/2	conv2D_3x3/-/32/1/2
bottleneck/1/16/1/1	bottleneck/1/16/1/1	bottleneck/1/16/1/1	**bottleneck_1**/1/16/1/1
bottleneck/**6**/**24**/2/2	bottleneck/**4**/**16**/2/2	bottleneck/4/**32(24)**/2/2	**bottleneck_1**/4/**32**/2/2
bottleneck/**6**/32/**3**/2	bottleneck/**8**/32/**2**/2	bottleneck/8/**64(48)**/2/2	**bottleneck_1**/8/**64**/2/2
bottleneck/**6**/64/**4**/2	bottleneck/**8**/64/**2**/2	bottleneck/8/64/2/2	**bottleneck_1**/8/**128**/2/2
bottleneck/6/96/3/1			
bottleneck/**6**/**160**/**3**/2	bottleneck/**8**/**128**/**1**/2	bottleneck/8/128/1/2	**bottleneck_1**/8/**256**/1/2
bottleneck/**6**/**320**/1/1	bottleneck/**8**/**1024**/1/1	bottleneck/8/1024/1/1	**bottleneck_1**/8/1024/1/1
conv2D_1x1/-/1280/1/1	**ReLU6**	ReLU6	ReLU6
avgpool 7x7	avgpool 7x7	avgpool 7x7	avgpool 7x7
conv2D_1x1/-/k/1/	conv2D_1x1/-/k/1/	conv2D_1x1/-/k/1/	conv2D_1x1/-/k/1/

미세 특징을 추출하므로 정확도에 큰 영향을 미치므로 그대로 유지하였다. 두 번째 층은 첫 번째 층과 동일한 피쳐맵 크기를 가지고 있어 확장 계수와 피쳐맵 수가 연산량에 가장 큰 영향을 미친다. 따라서 확장 계수를 4로 줄이고 출력 피쳐맵의 수도 24에서 16으로 줄였다. 세 번째와 네 번째 층은 정확도 유지를 위해 확장 계수를 8로 증가시키고 대신 반복층을 3개에서 2개로 줄였다. MobileNet V2의 다섯 번째 층은 정확도에 큰 영향을 미치지 않아 제거하였다. 따라서, 다음 층은 피쳐맵 수를 네 번째 층의 2배인 128로 증가시키고 반복층은 하나로 줄였다. 전체적으로 Bottleneck 층으로 구성된 구조를 최대한 유지하면서 각 bottleneck 층 내부의 반복 회수를 2개 이내로 줄였다. MobileNet V2는 마지막 병목층 이후 1×1 convolution을 이용하여 피쳐맵을 1280개로 늘리고 전역 평균 풀링(global average pooling)층을 통과시켜 출력층과 연결한다. SM V2는 마지막 병목층에서 확장이후 축소를 생략하고 ReLU6 활성화이후 전역 평균 풀링(global average pooling)층을 통과시켜 출력층과 연결한다.

3. 실험 결과 및 분석

표 2에 다양한 CNN의 특성을 비교한 결과가 나타나 있다. 학습에 사용한 하드웨어는 CPU Quadcore 3.4GHz, 메모리 16GB, GPU Nvidia GTX1060이고 Ubuntu 16.04 OS와 Tensorflow 프레임워크를 이용하였다. 분류 정확도(Accuracy)는 ImageNet 데이터셋에 대한 Top-1 정확도이고 GPU1과 GPU2는 각각 하나와 두 개의 GPU 카드를 이용했을 때 하나의 이미지 데이터가 학습시 네트워크를 통과하는 시간이다. CPU는 GPU를 이용하지 않고 CPU만으로 실행시킨 결과이며 GPU 이용시 발생하는 현상을 분석하기 위해 측정하였다. 정확도는 각 네트워크를 발표한 문헌에서 참조하였고 나머지 값들은 실험을 통해 얻었다. 각 네트워크의 연산량은 구조에 따라 큰 차이를 보이고 CPU 실행시간은 연산량에 선형적으로 비례하지는 않지만, 실제 연산량을 가장 잘 반영하는 값이다. 이는 연산량(Computation)이 덧셈/곱하기 연산을 위주로 계산된 값이어서 메모리 접근 시간이나 네트워크와 관련한 순수 연산을 제외한 동작을 위한 시간을 고려하지 않았기 때문이다. GPU를 이용한 연산의 경우에는 실행 시간의 차이가 연산량의 차이를 거의 반영하지 못한다는 것을 알 수 있다. VGG16 이후 CNN은 convolution 연산과 파라미터 메모리 크기를 줄이기 위해 노력하여 단순 직렬 구조를 탈피하여 동일한 층에서 다양한 연산을 병렬적으로 진행하고 skip connection 등을 도입하였는데 이러한 구조가 한 층의 연산에 다양한 파라미터를 필요로 하여 메모리 접근 빈도가 높아진 것으로 추정된다. GPU2의 경우에도 층 내부의 병렬 연산에는 효과가 있으나 이를 다시 모으거나 더하는 과정이 자주 반복되면서 병렬 처리의 효과가 감소하여 50% 정도의 성능 개선이 이루어지고 있다. 즉, GPU를 딥러닝에 효과적으로 활용하기 위해서는 3D 연산과는 다른 딥러닝에 최적화된 구조가 필요하다. 또한, Depthwise-separable convolution이 연산량이 줄어든 것에 비해 GPU 연산 시간에서 큰 효과가 나타나지 않는데 이는 cuDNN

Table 2. Performance of various CNNs.
표 2. 다양한 CNN의 특성 비교

CNN	Accuracy	Computation [MFLOS]	Parameters [M]	GPU1 [ms]	GPU2 [ms]	CPU [ms]
VGG16	0.71	15618	138.4	6.16	4.39	265
Inception v3	0.78	5749	23.8	5.63	3.88	108
ResNet 101	0.76	7623	44.6	7.68	5.43	148
SENet	0.806	20915	115.3	25	18.55	583
DenseNet 121	0.75	2843	7.98	3.94	2.78	70.8
Xception	0.78	8386	22.9	9.75	7.49	177
MobileNet v1	0.71	491	3.31	1.94	1.35	17.3
MobileNet v2	0.72	369	2.35	2.22	1.53	14.6
ShuffleNet v2	0.68	146	2.3	1.81	1.48	7.3

라이브러리의 개선이 필요한 것으로 추정된다.

위에서 제안한 SM_2의 기본 구조에 기반해서 병목층의 출력 채널 수를 조절하며 다양한 구조에 대해 학습을 진행하여 그 결과를 비교하였다. 학습은 ImageNet100 데이터셋에 대해 진행하여 정확도(Acc.)를 측정하였고 각 네트워크의 연산량(Comp.) 및 파라미터의 크기(Param.)를 계산하였다. 또한, 하나의 데이터에 대한 네트워크 통과 시간을 CPU만을 이용한 경우(CPU)와 CPU와 GPU를 이용한 경우(GPU)에 대해 측정하였다. 모든 측정값은 Tensorflow가 제공하는 값을 이용하였다. 표 2와 표 3에서 측정 결과를 통해 알 수 있는 바와 같이 MobileNet V2는 MobileNet V1에 비해 연산량을 줄이고 CPU에서 실행시간을 줄였으면서도 정확도를 증가시켰다. 그러나 GPU를 이용한 연산 시간은 오히려 증가하는 현상을 보인다. 이는 표 2의 VGG16과 ResNet의 결과에서와 마찬가지로 skip connection과 관련이 있는 것으로 판단된다. ResNet의 경우 연산량이 VGG16에 비해 절반 정도로 줄었으나 GPU 실행 시간은 오히려 증가하였다. MobileNet V2의 경우도 MobileNet에 비해 convolution 연산은 줄었으나 skip connection이 추가되었다. skip connection이 존재하는 경우 이전 층의 피쳐맵을 불러와야 하는데 내부 메모리에 저장하지 않은 경우 외부 메모리에서 다시 불러와야 한다. 이 작업으로 인해 연산 시간이 증가한 것으로 보인다. 이러한 문제를 해결하기 위해서는 딥러닝을 위한 GPU 구조와 관련 소프트웨어의 개선이 필요하다.

SM_v2의 경우 주어진 연산 환경에서 MobileNet v2에 비해 연산량을 1/2 이상 줄였으나 정확도는 0.8%p 감소하고 CPU 연산시간은 30%, GPU 연산 시간은 28% 감소하였다. 정확도의 경우 MobileNet에 비해 0.6%p 증가하였다. SM_V2_x2nr은 GPU 연산 시간 성능을 개선하기 위해 SM_v2의 bottleneck에서 skip connection을 제거한 bottleneck_1을 적용하고 대신 피쳐맵 수를 2배 증가시킨 네트워크인데 연산량이 SM_v2보다 20% 증가하였으나 CPU 연산시간은 7% 증가하고 GPU 연산 시간은 4% 감소하였다. 이를 통해 skip connection이 GPU 연산 시간을 증가시킨다는 것을 알 수 있다. 한편, SM_V2_x2nr의 정확도가 SM_V2에 비해 0.3%p 감소하여 GPU 연산 시간외에는 장점이 없어 skip connection이 필요함을 알 수 있다. 따라서, SM_V2에 대해 GPU의 구조와 관련 소프트웨어를 개선하거나 위의 문제가 없는 딥러닝 전용 가속기를 이용하여 GPU 연산 시간을 개선하면 연산 시간이나 정확도에서 모두 SM_V2_x2nr보다 좋은 결과를 보여줄 것이다.

SM_V2는 MobileNet에 비해 연산 시간을 30% 정도 줄였으나 정확도가 0.8%p 감소하였다. 정확도를 개선하기 위해 SM_V2에 대한 최적화를 진행하였다. 먼저 6개의 병목층중 첫 번째와 마지막 층을 제외한 나머지 4개의 층에 대해 일부를 선택하여 출력 채널의 수를 2배로 증가시켰다. 표 4에서 SM_V_2_2는 4개의 층에서 앞의 2개의 층에 대해 출력 채널 수를 2배로 증가시킨 구조이고 SM_V2_3_2는 가운데 2개의 층에 대해 2배 증가시킨 구조이다. SM_V2_2_2의 경우 SM_V2의 기본 구조에 비해 연산량이 50% 정도 증가하였으나 정확도는 1.6%p 증가하였다. 이는 MobileNet V2에 비해 정확도가 0.8%p 만큼 더 높으면서도 연산량은 40% 감소한 것이다. SM_V2_2_1.5는 SM_V2_2_2에서 출력 채널을 2배가 아닌 1.5배 증가시킨 구조로 MobileNet V2에 비해 정확도는 0.6%p 증가하고 연산량은 45% 감소한 구조이다. 따라서, 여전히 MobileNet V2보다 더 높은 정확도를 가지면서도

Table 3. Comparison of MobileNet and SM.
표 3. MobileNet과 SM의 특성 비교

Model	Acc. [%]	CPU [ms]	GPU [ms]	Comp. [FLOPs]	Param.
MobileNet V1	78.9	17.3	1.94	491.3M	3.31M
MobileNet V2	80.3	14.6	2.22	368.5M	2.35M
ShuffleNet V2	76.4	7.3	1.41	146M	2.3M
SM_V2_x2nr	79.2	10.9	1.47	186.92M	1.64M
SM_V2	79.5	10.2	1.53	155.3M	1.54M

Table 4. Results of SM optimization.
표 4. SM의 최적화 결과

Model	Acc. [%]	CPU [ms]	GPU [ms]	Comp. [FLOPs]	Param.
SM_V2_0	79.5	10.2	1.53	155.3M	1.54M
SM_V2_2_2	81.1	13.4	1.88	231.7M	1.6M
SM_V2_3_2	80.8	13.1	1.73	231.3M	1.77M
SM_V2_2_1.5	80.9	12.1	1.72	201.9M	1.57M
MobileNet V2	80.3	14.6	2.22	368.5M	2.35M

연산량을 더 감소시킬 수 있어 필요에 따라 채널 수 증가비를 2 또는 1.5로 선택하여 사용할 수 있다.

Ⅲ. 결론

휴대용 또는 에지 기기를 위한 MobileNet V2를 개선한 CNN 구조를 제안하였다. MobileNet V2의 출력 채널 수를 조정하고 일부 층을 제거하는 대신 확장 계수를 증가시켜 MobileNet V2에 비해 정확도는 80.3%에서 81.1%로 증가하였고 연산량은 최대 45% 감소하였다. 또한, skip connection 구조를 현재의 GPU 구조가 제대로 지원하지 못한다는 것과 이를 위해 GPU 구조와 관련 소프트웨어의 개선이 필요함을 보였다.

References

[1] K. Simonyan, A. Zisserman, “Very Deep Convolutional Network for Large-scale image Recognition,” *2015 International Conference on Learning Representations (ICLR)*, San Diego, USA, pp.7-9, 2015.

[2] C. Szegedy, X. Zhang, S. Ren, J. Sun, “Going Deeper with Convolutions,” *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, Massachusetts, USA, pp.1-9, 2015. DOI: 10.1109/CVPR.2015.7298594

[3] K. He, “Deep residual Learning for Image Recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, Nevada, pp.770-778, 2016.
DOI: 10.1109/CVPR.2016.90

[4] François Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.1801-1807, 2016.

[5] “Densely Connected Convolutional Networks,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.2261-2269, 2016.
DOI: 10.1109/CVPR.2017.243

[6] J. Hu, L. Shen and G. Sun, “Squeeze-and-Excitation Networks,” *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.7132-7141, 2018.
DOI: 10.1109/CVPR.2018.00745

[7] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” arXiv: 1704.04861, 2017

[8] N. Ma, X. Zhang, H. T. Zheng, J. Sun, “ShuffleNet v2: Pratical Guidelines for Efficient CNN Architectur Design,” *The 16th European Conference on Computer Vision (ECCV)*, 8-14, pp.116-131, 2018.

[9] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L. C. Chen, “Mobilenet v2: Inverted residuals and Linear Bottlenecks,” *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 18-22, pp.4510-4520, 2018.
DOI: 10.1109/CVPR.2018.00474

BIOGRAPHY

Chanho Lee (Member)

1987 : BS in Electronic Engineering, Seoul National University.
1989 : MS in Electronic Engineering, Seoul National University.
1994 : Ph.D in Electrical Engineering, UCLA

Joongkyung Lee (Member)

2018 : BS degree in Electronic Engineering, Soongsil University.
2020 : MS degree in Electronic Engineering, Soongsil University.

Cong Ahn Ho (Member)

2000 : BS degree in Electronic and Communication Engineering, FPT University, Vietnam.
2020 : MS student in Information and telecommunications Engineering, Soongsil University.