

MobileNets: Efficient Convolutional Neural Networks for Mobile Vision

: Efficient Convolutional Neural Networks for Mobile Vision Applications 논문
링크: [MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications](#)

1. Abstract

모바일, embedded vision 앱에서 사용되는 것을 목적으로 한 MobileNet이라는 효율적인 모델을 제시한다. Depth-wise separable convolutions라는 구조에 기반하며 2개의 단순한 hyper-parameter를 가진다. 이 2가지는 사용되는 환경에 따라 적절히 선택하여 적당한 크기의 모델을 선택할 수 있게 한다. 수많은 실험을 통해 가장 좋은 성능을 보이는 설정을 찾으며 타 모델에 비해 성능이 거의 떨어지지 않으면서 모델 크기는 ~배까지 줄인 모델을 소개한다.

- 경량화된 효율적인 model을 제안
- 경량화 모델 구축을 위해 depthwise separable convolution 사용
- latency와 accuracy 사이의 효율적인 trade off를 하는 두 개의 파라미터

2. Introduction

ConvNet은 computer vision분야라면 어디서든 사용되었지만, 모델의 크기가 너무 커지고 가성비가 좋지 않다.

그리고 핸드폰이나 임베디드 시스템 같이 저용량 메모리 환경에 딥러닝을 적용하기 위해서는 **모델 경량화가 필요하다**. 그래서 이 논문에서는 모델의 크기와 성능을 적절히 선택할 수 있도록 하는 2개의 hyper-parameter를 갖는 효율적인 모델을 제시한다. 이 2개의 hyper-parameter는 latency(지연시간)과 accuracy(성능)의 균형을 조절한다.

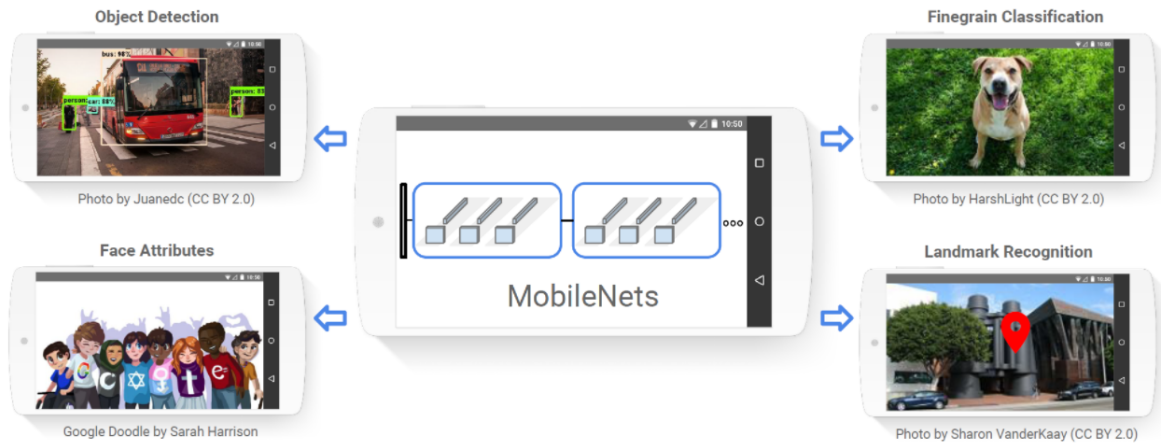


Figure 1. MobileNet models can be applied to various recognition tasks for efficient on device intelligence.

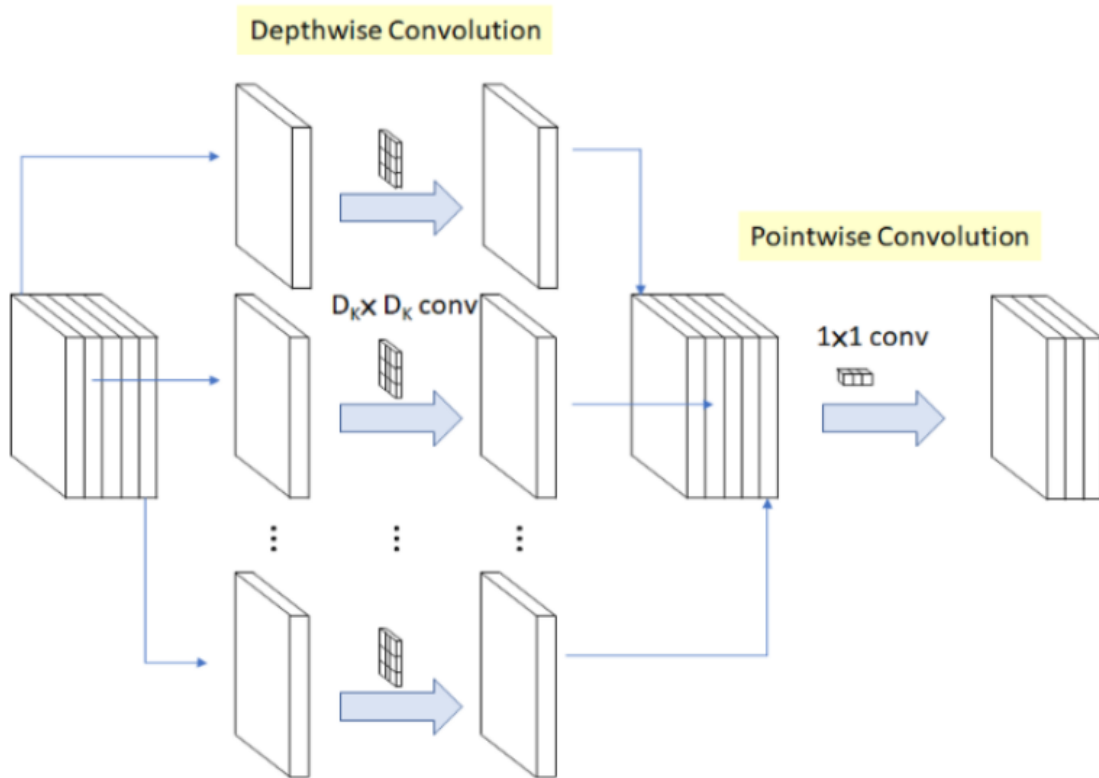
MobileNet은 기본적으로 작은 모델, 여기서 **Depthwise separable convolutions**을 사용한다.

3. Depthwise Separable Convolution

표준 convolution을

- Depthwise convolution과
- Pointwise convolution(1×1 convolution)

으로 쪼갠 것이다.



(1) Depthwise convolution

Depthwise convolution은 각 입력 채널에 대하여 3×3 conv 하나의 필터가 연산을 수행하여 하나의 피쳐맵을 생성한다. 입력 채널 수가 M 개이면 M 개의 피쳐맵을 생성하는 것이다. 각 채널마다 독립적으로 연산을 수행하여 spatial correlation을 계산하는 역할을 한다.

예를 들어, 5 채널의 입력값이 입력되었으면, 5개의 3×3 conv가 각 채널에 대하여 연산을 수행하고, 5개의 feature map을 생성한다.

Depthwise convolution의 연산량은 다음과 같다.

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F$$

M 은 input channel

N 은 output channel

D_K 는 filter size

D_F 는 output size

(2) Pointwise convolution

Pointwise convolution은 **Depthwise convolution**이 생성한 피쳐맵들을 **1x1conv**로 채널 수를 조정한다.

1x1conv는 모든 채널에 대하여 연산하므로 cross-channel correlation을 계산하는 역할을 한다.

Pointwise convolution의 연산량은 다음과 같다.

$$M \cdot N \cdot D_F \cdot D_F$$

M은 input channel

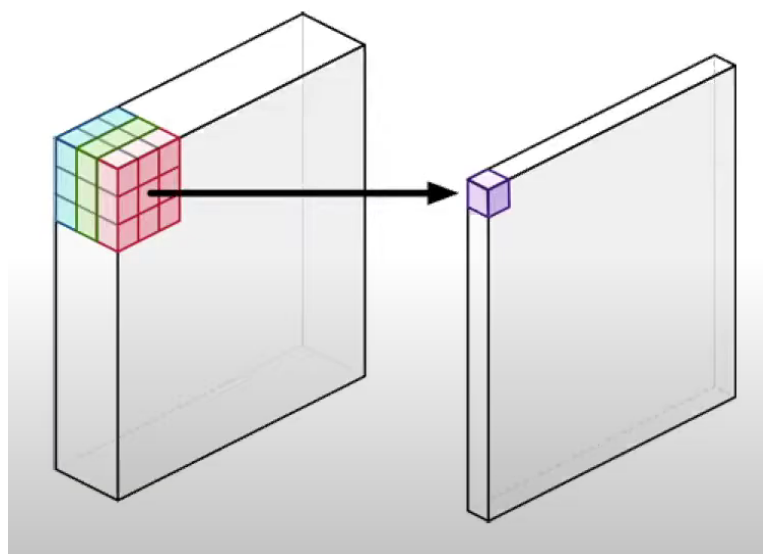
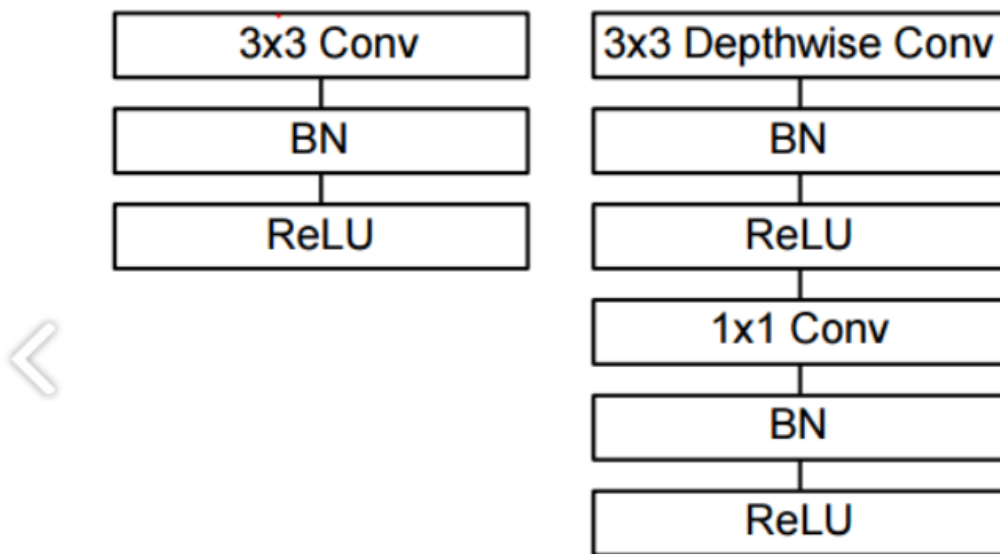
N은 output channel

D_F 는 output size

(3) Depthwise separable convolution

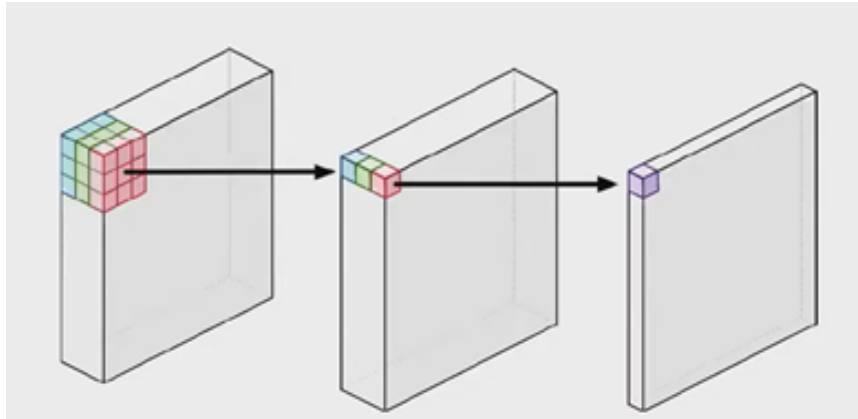
Depthwise separable convolution은 Depthwise convolution 이후에 Pointwise convolution을 적용한 것

아래 그림은 **표준 Convolution**과 **MobileNet**에서 사용하는 **Depthwise separable convolution** 구조



Original Convolution

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$$



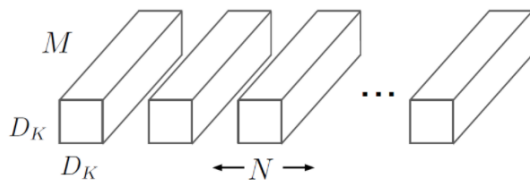
Depthwise Conv & Pointwise Conv

↑

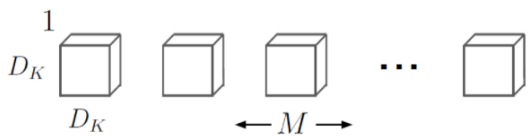
↑

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F$$

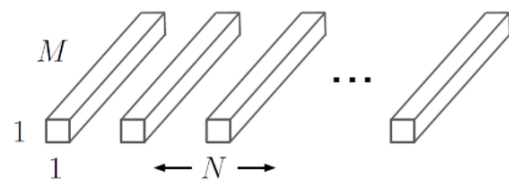
이 과정을 시각화하면 다음과 같다. 각 큐브는 3차원의 필터 모양(혹은 parameter의 개수)을 나타내며, 표준 conv는 딱 봐도 큐브의 부피 합이 커 보이지만 Depthwise convolution과 Pointwise convolution는 하나 또는 2개의 차원이 1이므로 그 부피가 작다(즉, parameter의 수가 많이 적다).



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Figure 2. The standard convolutional filters in (a) are replaced by two layers: depthwise convolution in (b) and pointwise convolution in (c) to build a depthwise separable filter.

이는 다르게 말해서 3차원적인 계산을 두 방향의 차원으로 먼저 계산한 후 나머지 한 차원을 그 다음에 계산하는 방식이라 생각해도 된다.

Depthwise separable convolution의 전체 연산량은 Depthwise Convolution과 Point Convolution 둘의 연산량을 더해준 것이 된다. **Depthwise separable convolution 연산량은 기존 conv 연산량보다 8~9배 더 적다.**

cf) Xception은 Depthwise separable convolution을 활용하여 감소한 파라미터 수 만큼 층을 쌓아 성능을 높이는데 집중했는데, MobileNet은 반대로 경량화에 집중한 것이다.

4. MobileNet Architecture

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32 \text{ dw}$	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64 \text{ dw}$	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128 \text{ dw}$	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128 \text{ dw}$	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256 \text{ dw}$	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256 \text{ dw}$	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5×	Conv dw / s1	$3 \times 3 \times 512 \text{ dw}$
	Conv / s1	$1 \times 1 \times 512 \times 512$
	Conv dw / s2	$3 \times 3 \times 512 \text{ dw}$
	Conv / s1	$1 \times 1 \times 512 \times 1024$
	Conv dw / s2	$3 \times 3 \times 1024 \text{ dw}$
	Conv / s1	$1 \times 1 \times 1024 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

MobileNet 모델에서 Pointwise convolution이 비교적 많은 연산량과 파라미터가 사용된다는 것을 볼 수 있다.

5. Hyper - parameter

MobileNet은 모델이 latency와 accuracy를 조절하는 두 개의 하이퍼파라미터가 존재한다.

(1) Width Multiplier Thinner Models

첫 번째 하이퍼파라미터 α 는 MobileNet의 두께를 결정한다. conv net에서 두께는 각 레이어에서 필터수를 의미한다.

이 width Multiplier α 는 더 얇은 모델이 필요할 때 사용한다. 입력 채널 M과 출력 채널 N에 적용하여 αM , αN 이 된다. 따라서 연산량은 다음과 같이 된다.

$$D_K \cdot D_K \cdot \alpha M \cdot D_F \cdot D_F + \alpha M \cdot \alpha N \cdot D_F \cdot D_F$$

α 는 0~1 범위이고 기본 MobileNet은 1을 사용한다. Width Multiplier를 낮추면 모델의 파라미터 수가 감소한다.

Table 6. MobileNet Width Multiplier

Width Multiplier	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
0.75 MobileNet-224	68.4%	325	2.6
0.5 MobileNet-224	63.7%	149	1.3
0.25 MobileNet-224	50.6%	41	0.5

(2) Resolution Multiplier: Reduced Representation

두 번째 하이퍼파라미터는 Resolution Multiplier ρ 이다.

모델의 연산량을 감소시키기 위해 사용한다. ρ 는 입력 이미지에 적용하여 해상도를 낮춘다.

범위는 0~1이고, 논문에서는 입력 이미지 크기가 224, 192, 169, 128 일때 비교를 한다. 기본 MobileNet은 $\rho=1$ 을 사용한다.

Layer/Modification	Million Mult-Adds	Million Parameters
Convolution	462	2.36
Depthwise Separable Conv	52.3	0.27
$\alpha = 0.75$	29.6	0.15
$\rho = 0.714$	15.1	0.15

두 가지 파라미터를 조정했을 때의 변화

6. Experiments

MobileNet을 여러 multiplier 등 여러 세팅을 바꿔가면서 실험한 결과인데, 주로 성능 하락은 크지 않으면서도 모델 크기나 계산량이 줄었음을 보여준다. 혹은 정확도는 낮아도 크기가 많이 작기 때문에 여러 embedded 환경에서 쓸 만하다는 주장을 한다.

Table 4. Depthwise Separable vs Full Convolution MobileNet

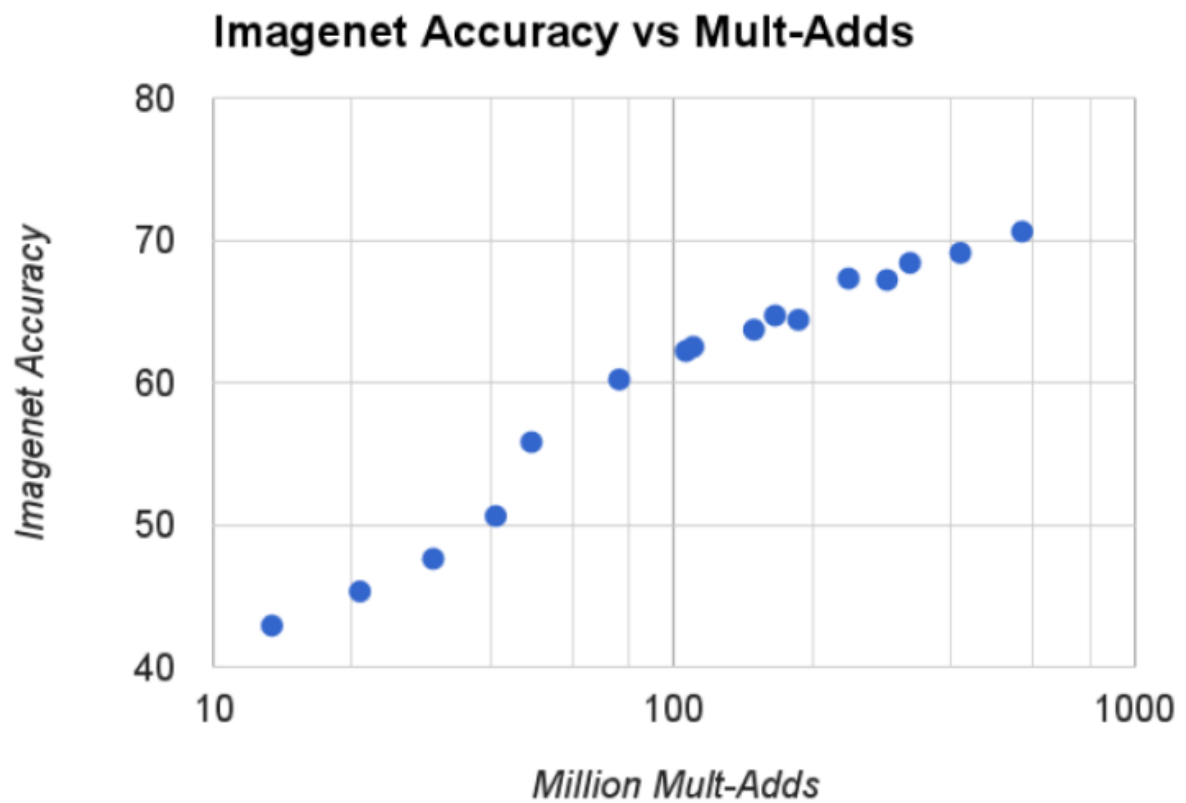
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
Conv MobileNet	71.7%	4866	29.3
MobileNet	70.6%	569	4.2

Table 5. Narrow vs Shallow MobileNet

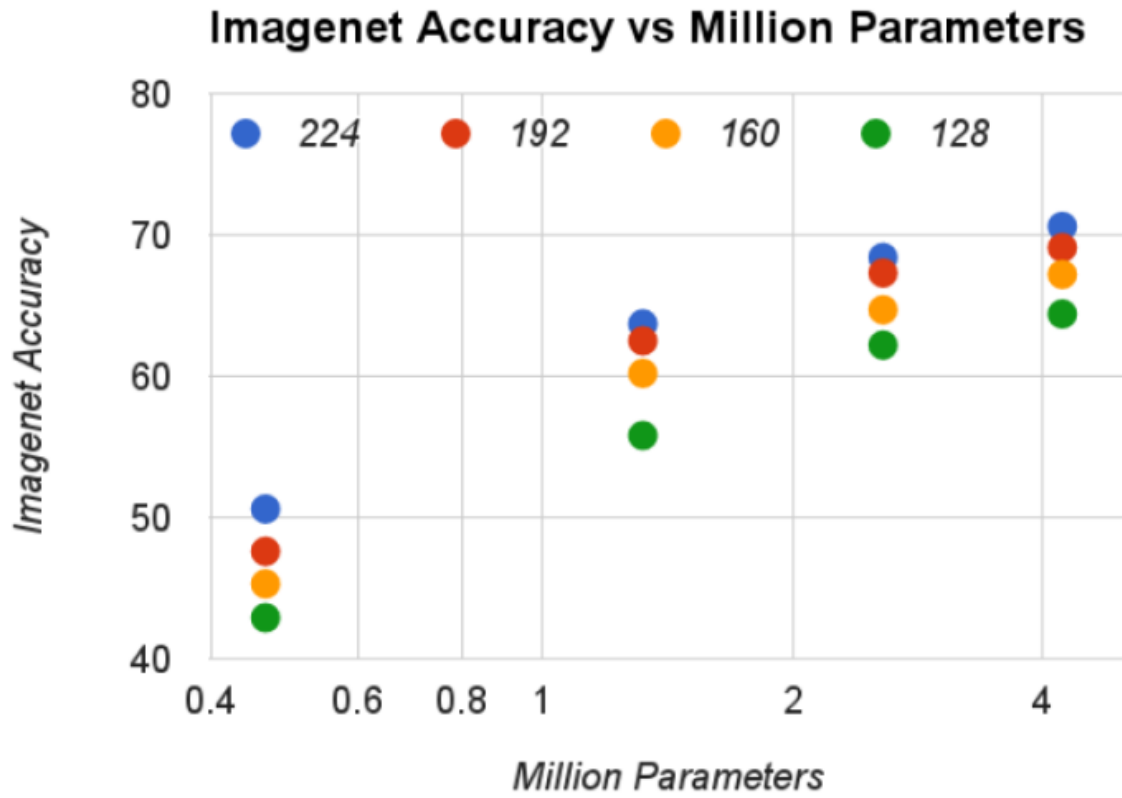
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
0.75 MobileNet	68.4%	325	2.6
Shallow MobileNet	65.3%	307	2.9

Depthwise Separable과 Full Convolution의 차이는 명확하다. 정확도는 1% 낮지만, **모델 크기는 7배 이상 작다**.

또 Narrow와 Shallow MobileNet을 비교하면 아래와 같다. (깊고 얇은 모델 vs 얇고 두꺼운 모델)



계산량과 성능 사이의 trade-off는 위처럼 나타난다. 계산량이 지수적으로 늘어나면, 정확도는 거의 선형적으로 늘어난다.



정확도, 계산량, 모델 크기를 종합적으로 비교

Table 8. MobileNet Comparison to Popular Models

Model	ImageNet	Million	Million
	Accuracy	Mult-Adds	Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

Table 9. Smaller MobileNet Comparison to Popular Models

Model	ImageNet	Million	Million
	Accuracy	Mult-Adds	Parameters
0.50 MobileNet-160	60.2%	76	1.32
Squeezenet	57.5%	1700	1.25
AlexNet	57.2%	720	60

다른 모델들과의 성능 비교

웹에서 얻은 대량이지만 noisy한 데이터를 사용하여 학습한 다음 Stanford Dogs dataset에서 테스트해보았다.

Table 10. MobileNet for Stanford Dogs

Model	Top-1	Million	Million
	Accuracy	Mult-Adds	Parameters
Inception V3 [18]	84%	5000	23.2
1.0 MobileNet-224	83.3%	569	3.3
0.75 MobileNet-224	81.9%	325	1.9
1.0 MobileNet-192	81.9%	418	3.3
0.75 MobileNet-192	80.5%	239	1.9

MobileNet의 또 다른 쓸모 있는 점은 전혀 또는 거의 알려져 있지 않은(unknown or esoteric) 학습 과정을 가진 큰 모델을 압축할 수 있다는 것

MobileNet 구조를 사용하여 얼굴 특징 분류기에서 distillation을 수행했는데, 이는 분류기가 GT label 대신에 더 큰 모델의 출력값을 모방하도록 학습하는 방식으로 작동한다. 기본 모델에 비해 최대 99%까지 연산량을 줄이면서도 성능 하락은 별로 없는 것을 볼 수 있다.

Width Multiplier / Resolution	Mean AP	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	88.7%	568	3.2
0.5 MobileNet-224	88.1%	149	0.8
0.25 MobileNet-224	87.2%	45	0.2
1.0 MobileNet-128	88.1%	185	3.2
0.5 MobileNet-128	87.7%	48	0.8
0.25 MobileNet-128	86.4%	15	0.2
Baseline	86.9%	1600	7.5

MobileNet을 물체 인식에도 적용시켜서 Faster-RCNN 등과 비교해 보았다. 이 결과 역시 모델 크기나 연산량에 비해 성능이 좋다는 것을 보여주고 있다.

Framework Resolution	Model	mAP	Billion Mult-Adds	Million Parameters
SSD 300	deeplab-VGG	21.1%	34.9	33.1
	Inception V2	22.0%	3.8	13.7
	MobileNet	19.3%	1.2	6.8
Faster-RCNN 300	VGG	22.9%	64.3	138.5
	Inception V2	15.4%	118.2	13.3
	MobileNet	16.4%	25.2	6.1
Faster-RCNN 600	VGG	25.7%	149.6	138.5
	Inception V2	21.9%	129.6	13.3
	Mobilenet	19.8%	30.5	6.1

얼굴인식 모델에서 FaceNet은 SOTA 모델인데, 적절히 distillation을 수행한 결과, 성능은 조금 낮으나 연산량을 고려하면 만족할 만한 수준인 것 같다.

Table 14. MobileNet Distilled from FaceNet

Model	1e-4	Million	Million
	Accuracy	Mult-Adds	Parameters
FaceNet [25]	83%	1600	7.5
1.0 MobileNet-160	79.4%	286	4.9
1.0 MobileNet-128	78.3%	185	5.5
0.75 MobileNet-128	75.2%	166	3.4
0.75 MobileNet-128	72.5%	108	3.8

7. Conclusion

Depthwise Separable Convolutions을 사용한 경량화된 모델 MobileNet을 제안하였다. 모델 크기나 연산량에 비해 성능은 크게 떨어지지 않고, 시스템의 환경에 따라 적절한 크기의 모델을 선택할 수 있도록 하는 여러 옵션(multiplier)를 제공하였다.

참고자료

블로그

<https://deep-learning-study.tistory.com/532>

https://greeksharifa.github.io/computer_vision/2022/02/01/MobileNetV1/

유튜브

<https://www.youtube.com/watch?v=GyQUBLDQEJI>