Small CNN Models

성균관대학교 소프트웨어학과
이 지 형

Mobile Devices

- Data Centers (Clouds)
  - Rarely safety critical
  - Low power is nice to have
  - Real time is preferable
- Mobile Devices
  - Usually safety critical (especially for self-driving cars)
  - Low power is must have
  - Real time is required

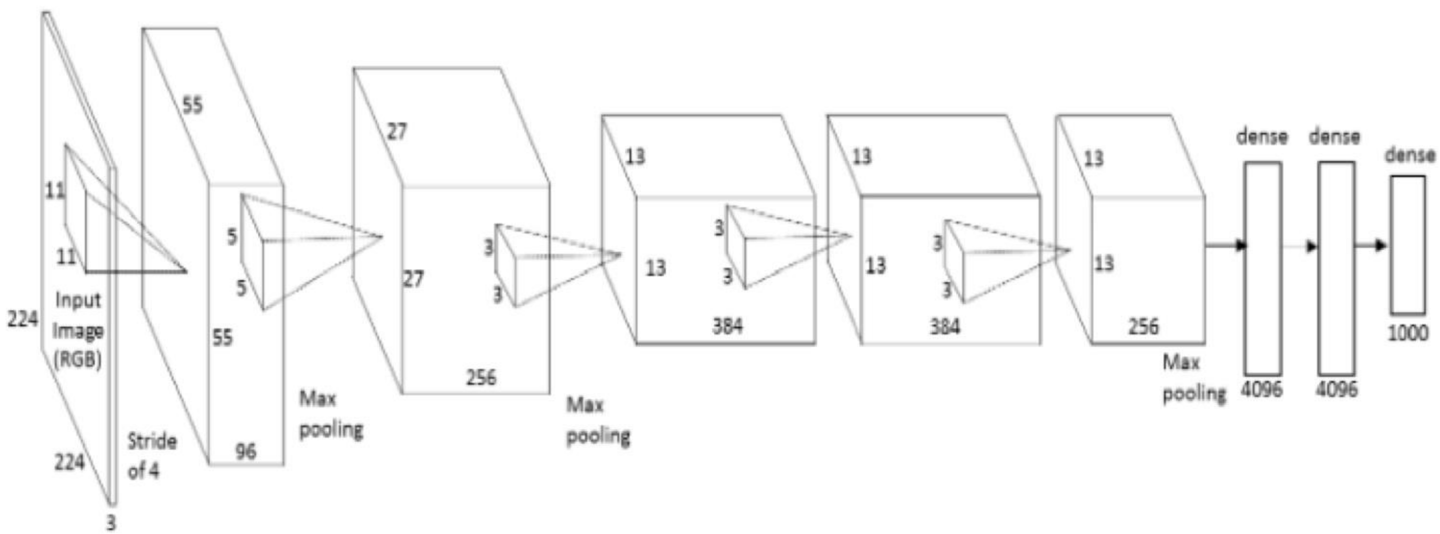
Mobile Devices

- **Deep Neural Networks for Mobile Devices**
  - Sufficiently high accuracy
  - Low computational complexity (Time)
  - Low energy usage
  - Small model size (Memory)

- **Merits of Small Deep Neural Networks**
  - Small DNNs train faster on distributed hardware
  - Small DNNs are more deployable on embedded processors
  - Small DNNs are easily updatable Over The Air(OTA)

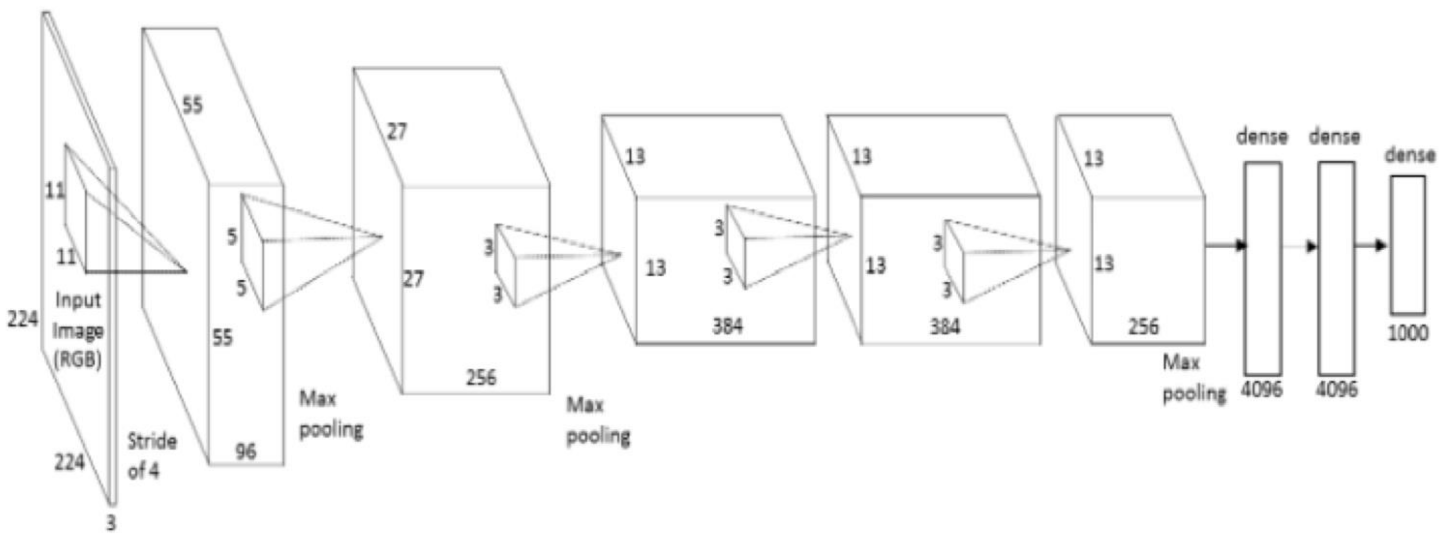
Huddles against Small Networks

- **Fully Connected Layers**
  - The FC7 layer in AlexNet has
    4096 input channels and 4096 filters -> 67MB of params



Huddles against Small Networks

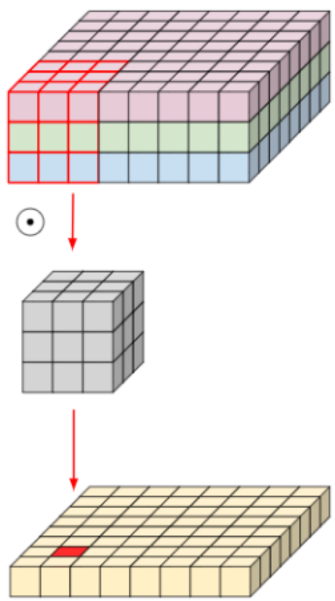
- **Filters**
  - **(Filter size) * (Filter size) * (Input channels) * (Output channels)**



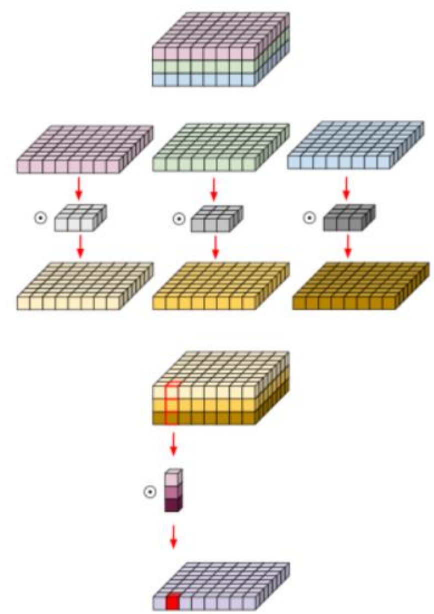
MobileNet-v1

Depthwise Separable Convolution

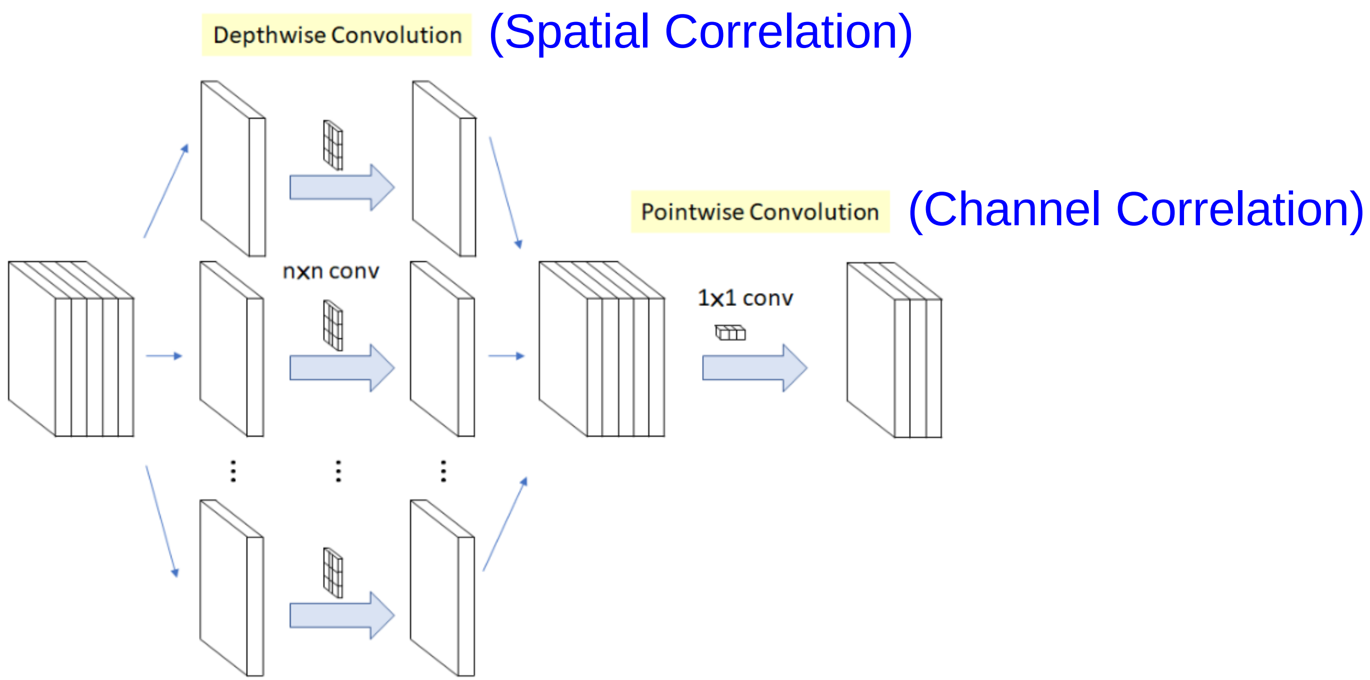
Regular Conv



Depthwise Separable Conv

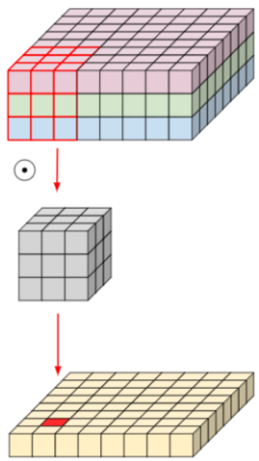


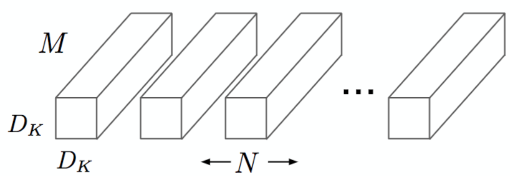
Depthwise Separable Convolution



Depthwise Separable Convolution

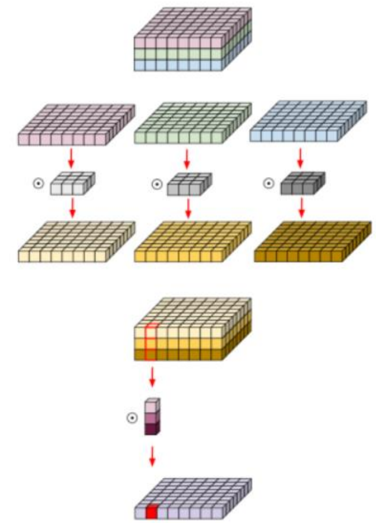
Regular Conv

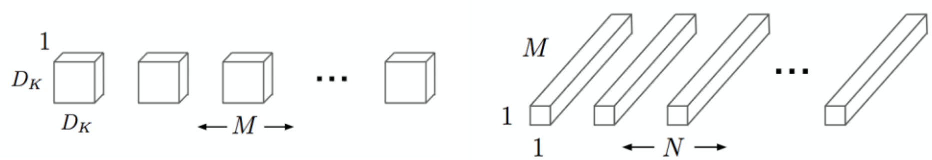




$$D_K \times D_K \times M \times N \times D_F \times D_F$$

Depthwise Separable Conv





$$D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F$$

Depthwise Separable Convolution

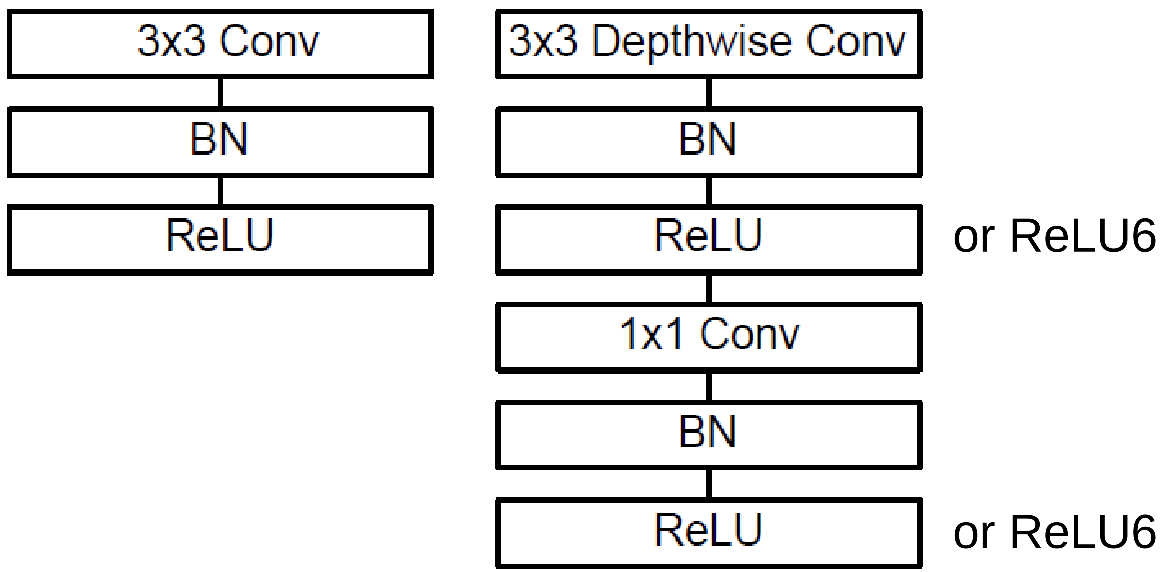


Figure 3. Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

Model Structure

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	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
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

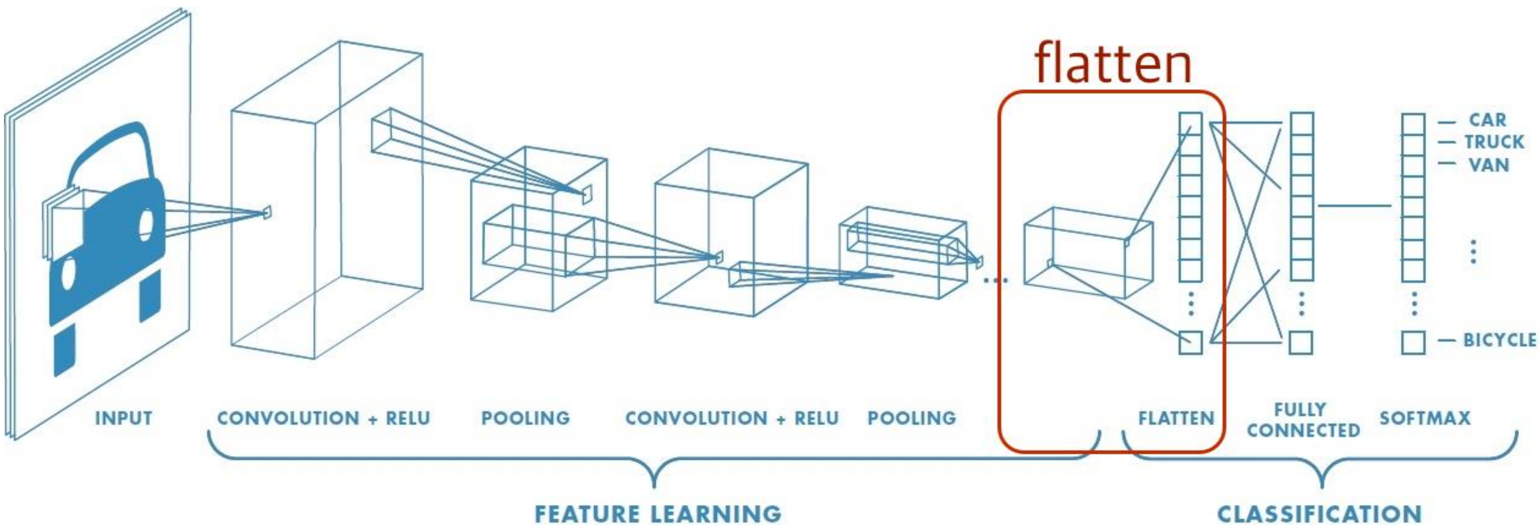
Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

Additional Feature

- **Width Multiplier Thinner Models**
  - For a given layer and width multiplier α, the number of input channels M becomes αM and the number of output channels N becomes αN
  - α with typical settings of 1, 0.75, 0.6 and 0.25
- **Resolution Multiplier Reduced Representation**
  - The second hyper parameter to reduce the computational cost of a neural network is a resolution multiplier ρ
  - 0< ρ≤ 1, which is typically set of implicitly so that input resolution of network is 224, 192, 160 or 128
    (ρ = 1, 0.857, 0.714, 0.571)
- **Computational cost:**
  - $D_K \times D_K \times \boldsymbol{\alpha} M \times \boldsymbol{\rho} D_F \times \boldsymbol{\rho} D_F + M \times N \times \boldsymbol{\rho} D_F \times \boldsymbol{\rho} D_F$

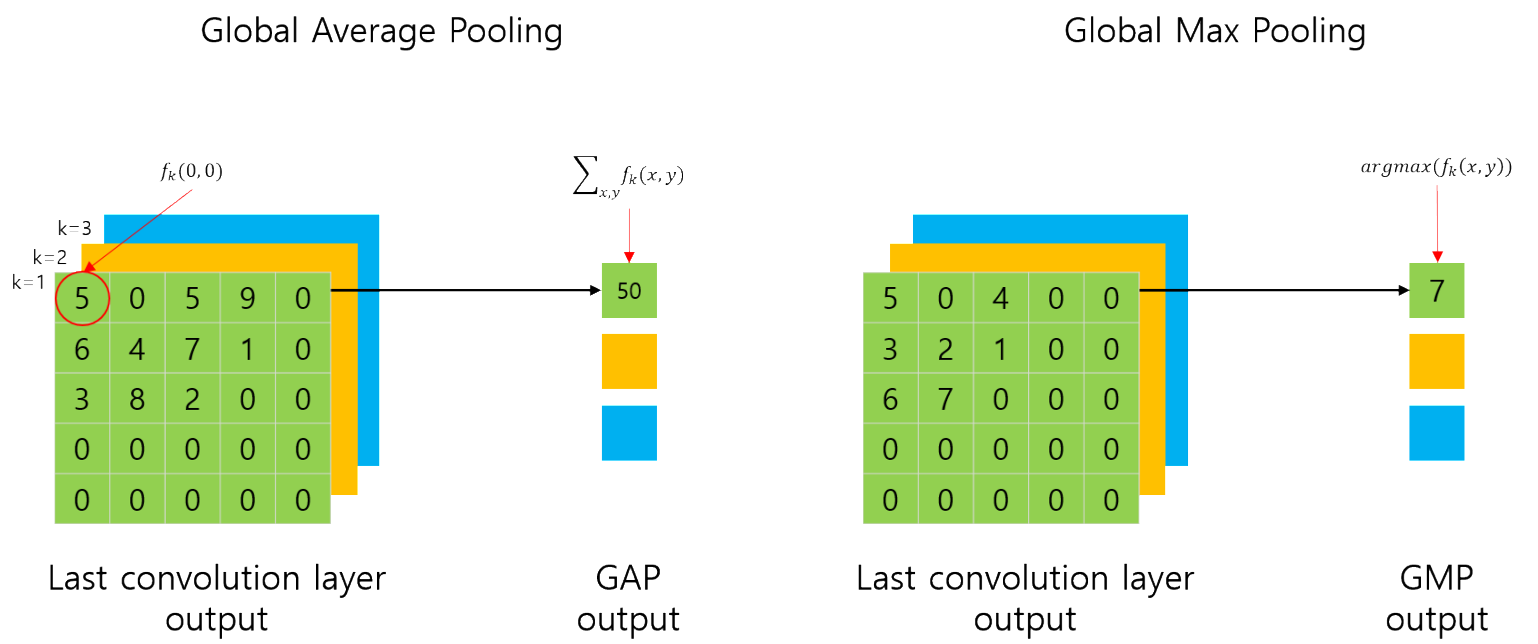
Changeable Input Size

- **Recap: Fully Connected Layer**
  - All pixels in feature maps are connected to FCL
  - A lot of connections
  - Non-flexible to input size change



Changeable Input Size

Global Average Pooling



Model Comparison

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

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

Table 7. MobileNet Resolution

Resolution	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
1.0 MobileNet-192	69.1%	418	4.2
1.0 MobileNet-160	67.2%	290	4.2
1.0 MobileNet-128	64.4%	186	4.2

Model Comparison

Table 8. MobileNet Comparison to Popular Models

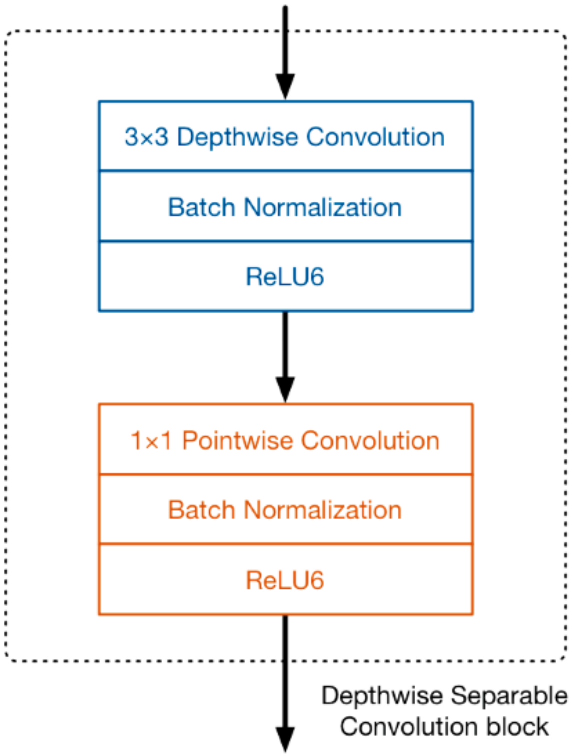
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

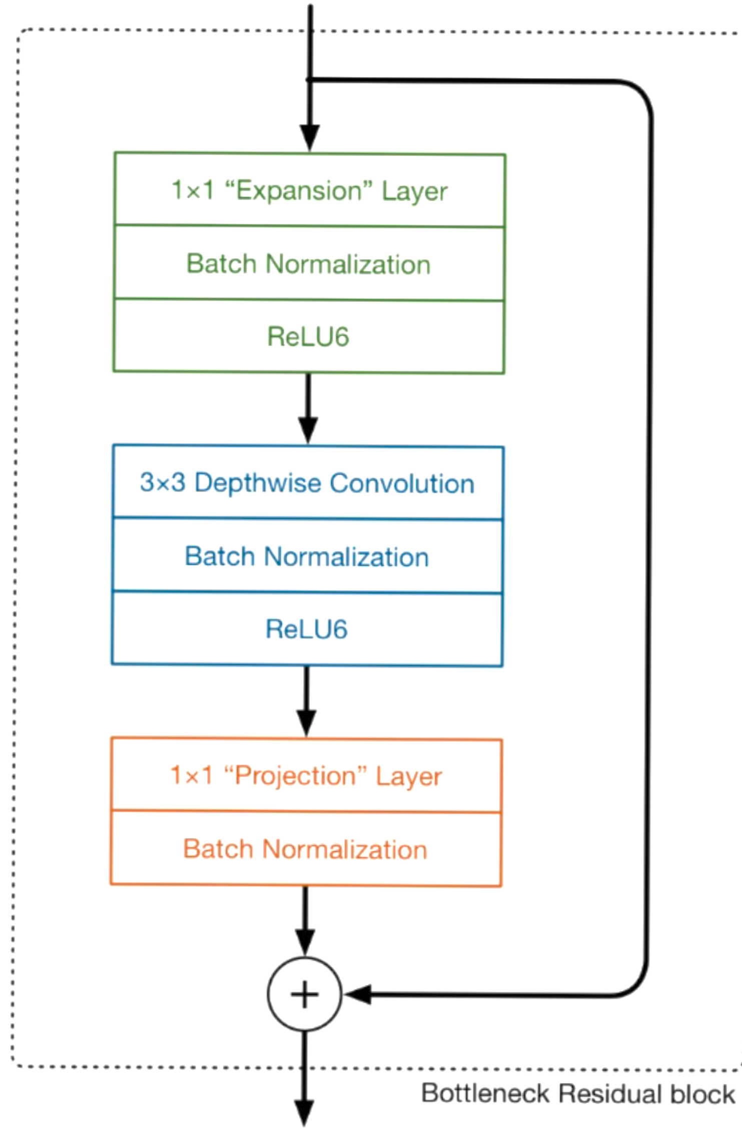
Table 9. Smaller MobileNet Comparison to Popular Models

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

MobileNet-v2

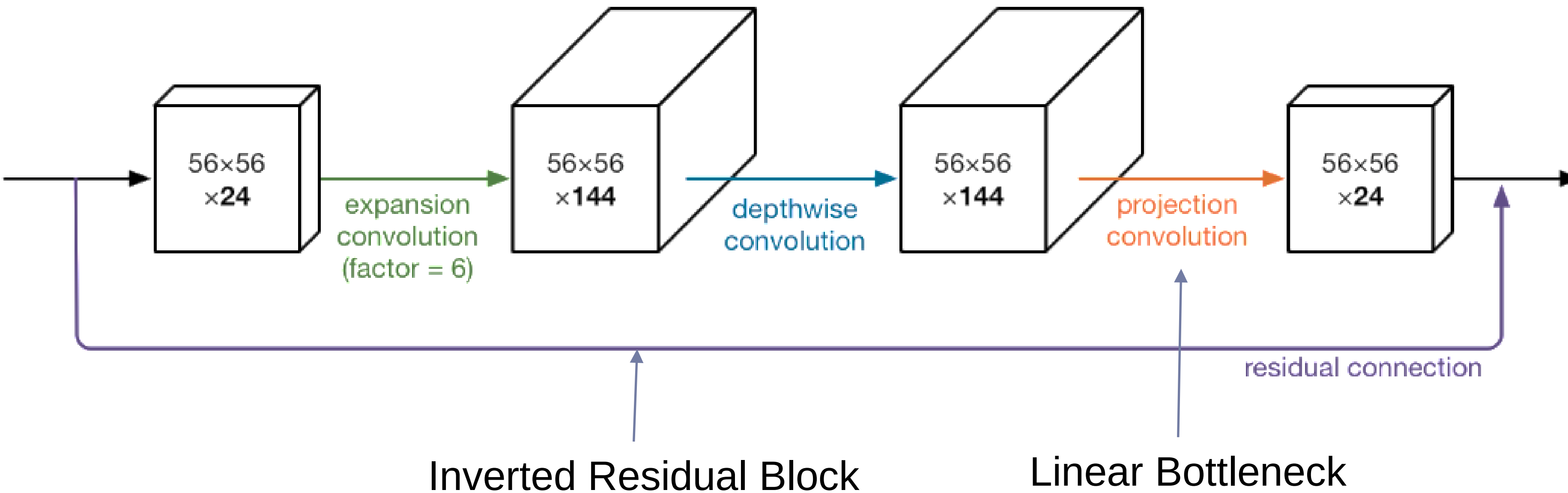
Blocks


3x3 Depthwise Convolution
Batch Normalization
ReLU6
1x1 Pointwise Convolution
Batch Normalization
ReLU6
Depthwise Separable Convolution block


1x1 "Expansion" Layer
Batch Normalization
ReLU6
3x3 Depthwise Convolution
Batch Normalization
ReLU6
1x1 "Projection" Layer
Batch Normalization
+
Bottleneck Residual block

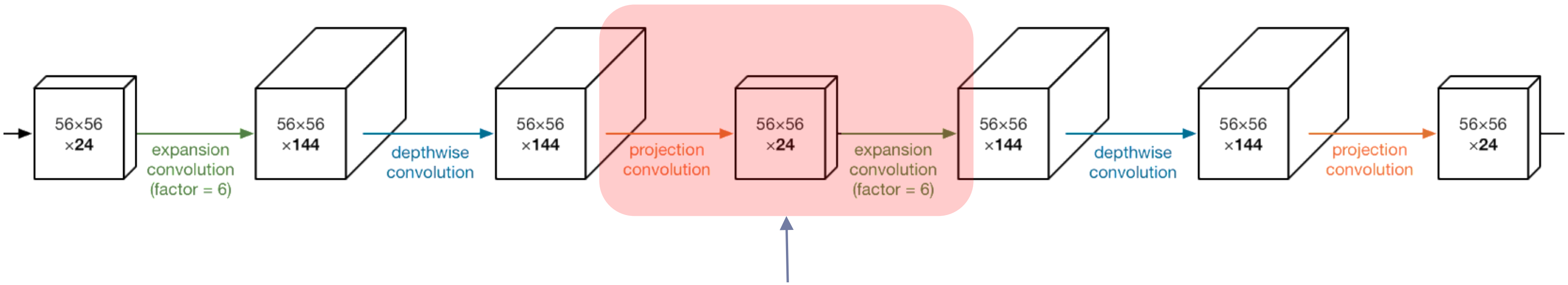
Blocks

- **This module takes as an input a low dim compressed representation which is first expanded to high dim and filtered with a lightweight depthwise convolution.**



Blocks

- If two layers are connected
  - Drawn without residual links



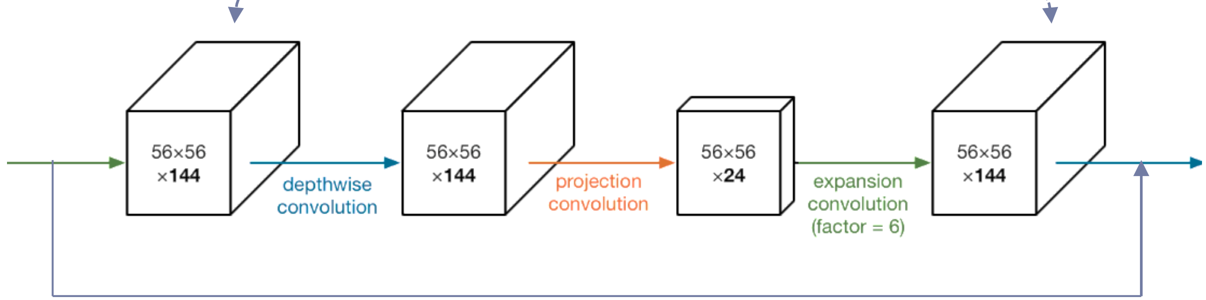
Almost equivalent to
pointwise convolution with bottleneck

Why?
- Dimension of Manifold is believed to be much lower than that of Input
- To effectively extract important information

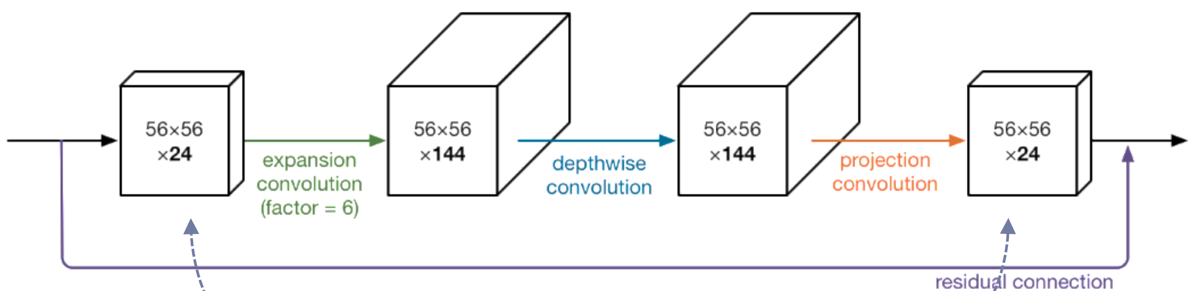
Blocks

- **Why not**

Large intermediate tensors



- **instead of**



Small intermediate tensors

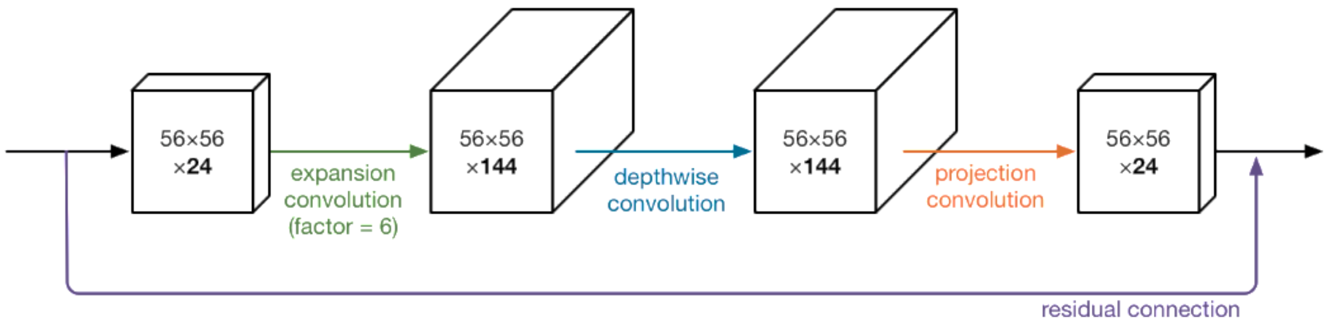
성균관대학교

Small Intermediate Tensors

- It can fit into small but very fast cache memory of mobile devises
- Allows to significantly reduce the memory footprint needed during inference
- Reduces the need for main memory access in many embedded hardware designs

Projection Convolution

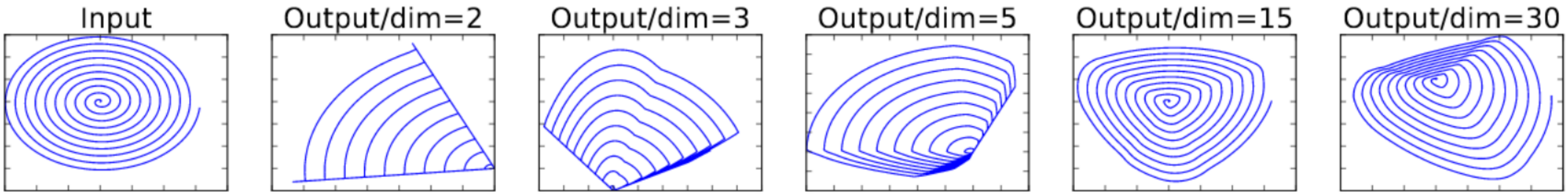
- **Why linear?**

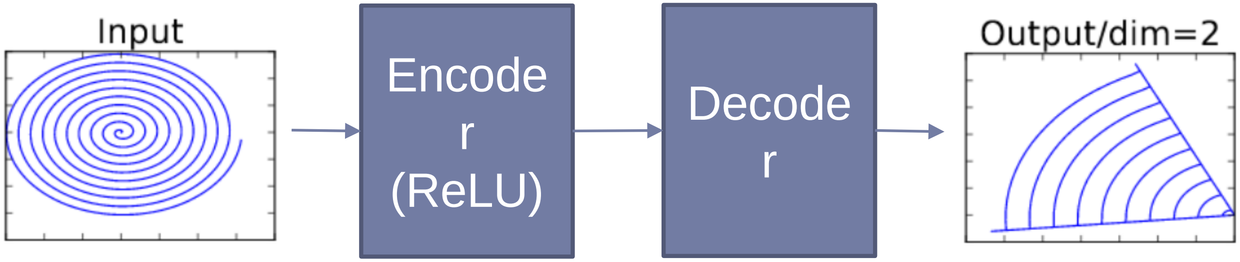


  - **Bottleneck layer -> Dimension reduction**
  - **Nee to efficiently capture important information**
- **However, ReLU can lose some important features**
  - **In average, half of feature maps are ZERO, which containing no informatio**

Projection Convolution

- **Why linear**
  - **Experiments on information loss with ReLU**

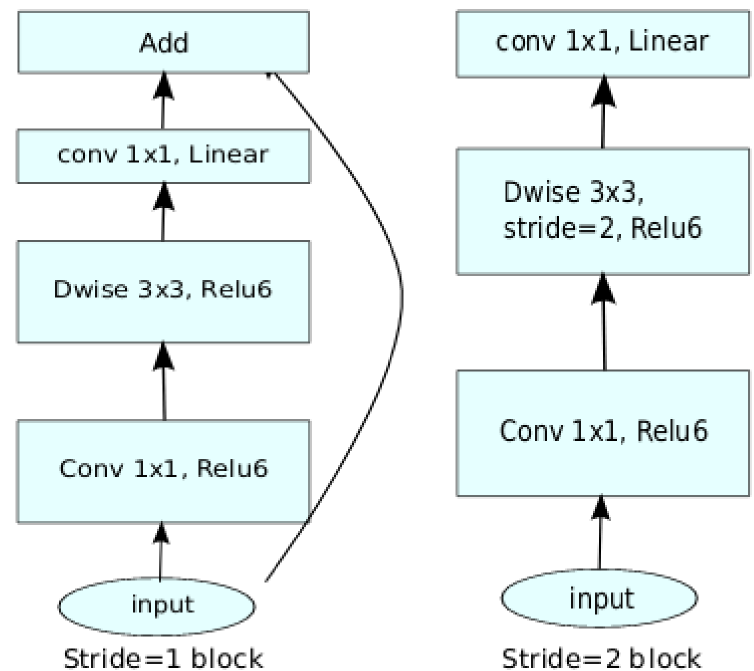




Model

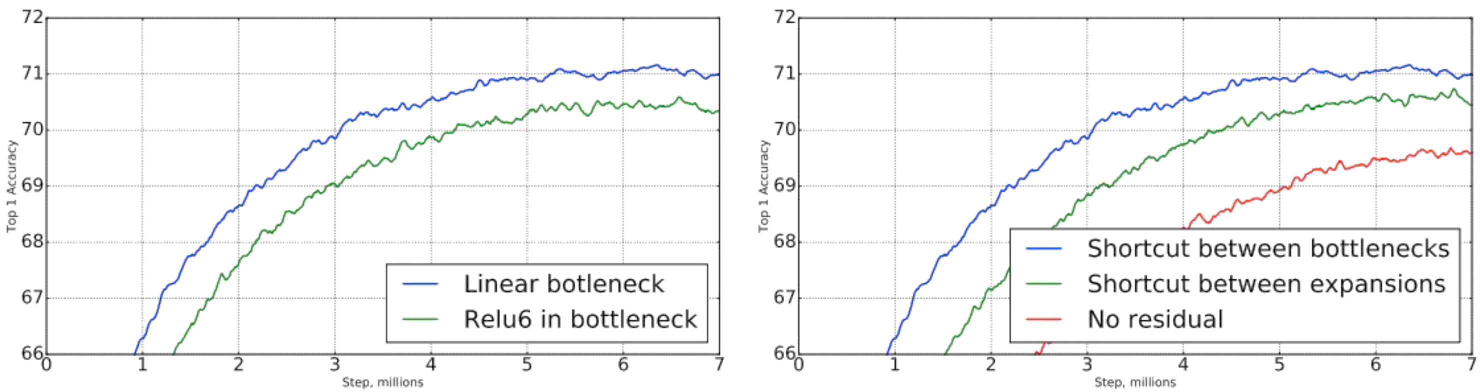
Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Table 2: MobileNetV2 : Each line describes a sequence of 1 or more identical (modulo stride) layers, repeated n times. All layers in the same sequence have the same number c of output channels. The first layer of each sequence has a stride s and all others use stride 1. All spatial convolutions use 3×3 kernels. The expansion factor t is always applied to the input size as described in Table 1.



Experiment

- ## Impact of non-linearities and residual link



(a) Impact of non-linearity in the bottleneck layer.

(b) Impact of variations in residual blocks.

Experiment

Operations and Parameters

	Operations (millions)	Parameters (millions)
MobileNet v1	569	4.24
MobileNet v2	300	3.47

Maximum FPS (frames-per-second)

	iPhone 7	iPhone X	iPad Pro 10.5
MobileNet v1	118	162	204
MobileNet v2	145	233	220

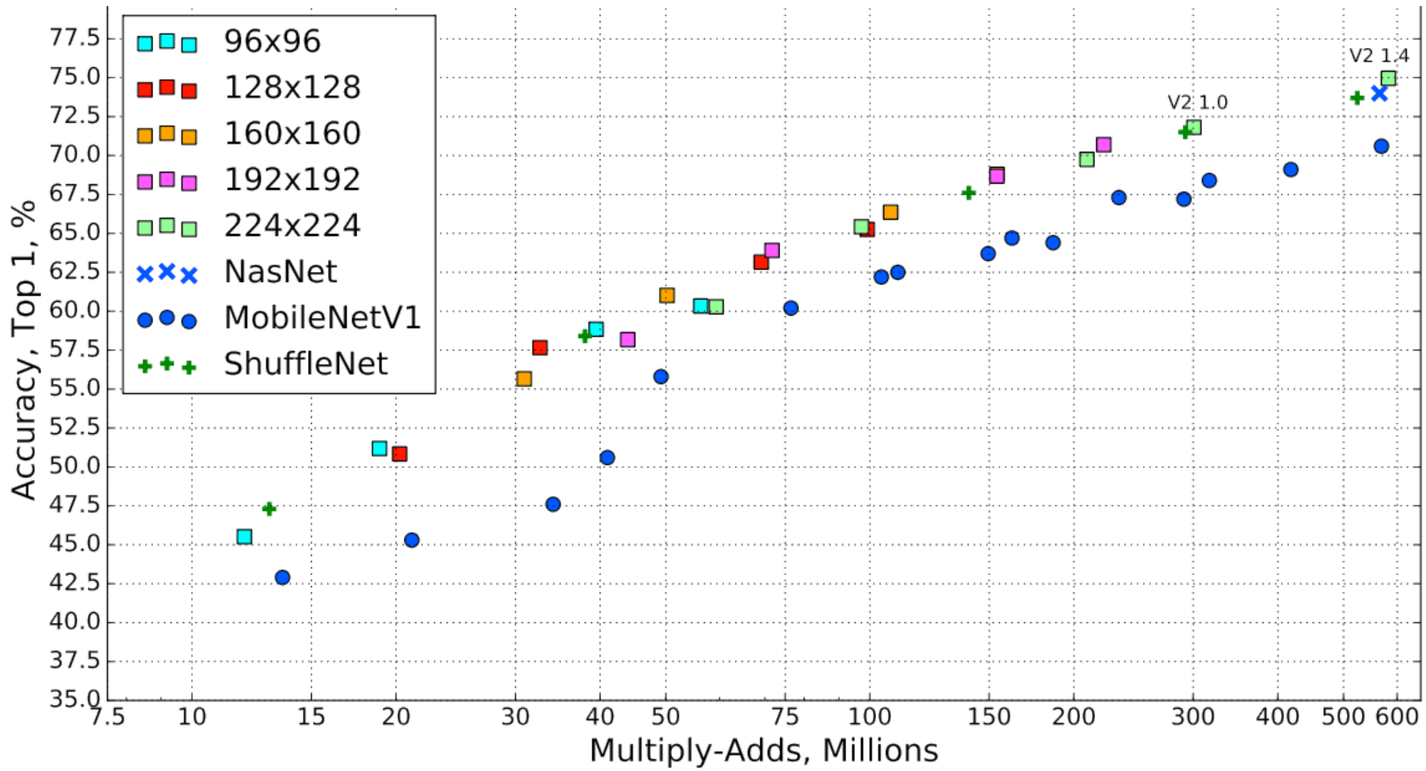
Experiment

- **ImageNet Classification**

Network	Top 1	Params	MAdds	CPU
MobileNetV1	70.6	4.2M	575M	113ms
ShuffleNet (1.5)	71.5	**3.4M**	292M	-
ShuffleNet (x2)	73.7	5.4M	524M	-
NasNet-A	74.0	5.3M	564M	183ms
MobileNetV2	**72.0**	**3.4M**	**300M**	**75ms**
MobileNetV2 (1.4)	**74.7**	6.9M	585M	**143ms**

Experiment

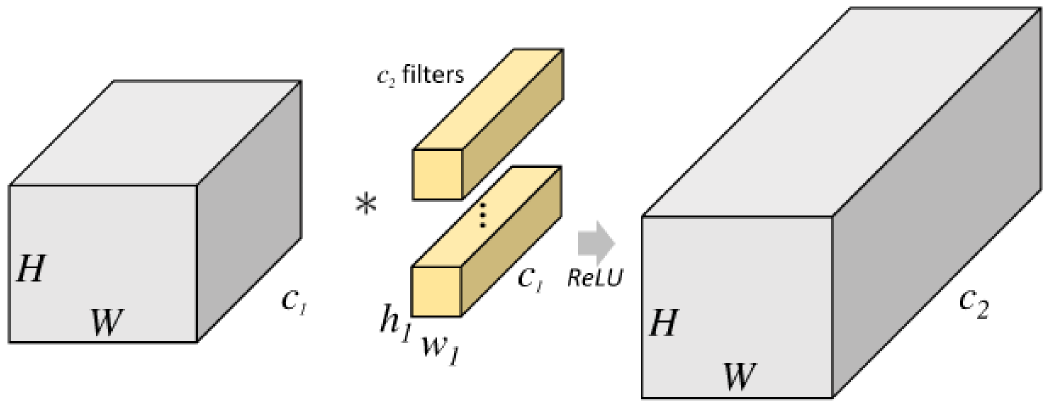
- ImageNet Classification



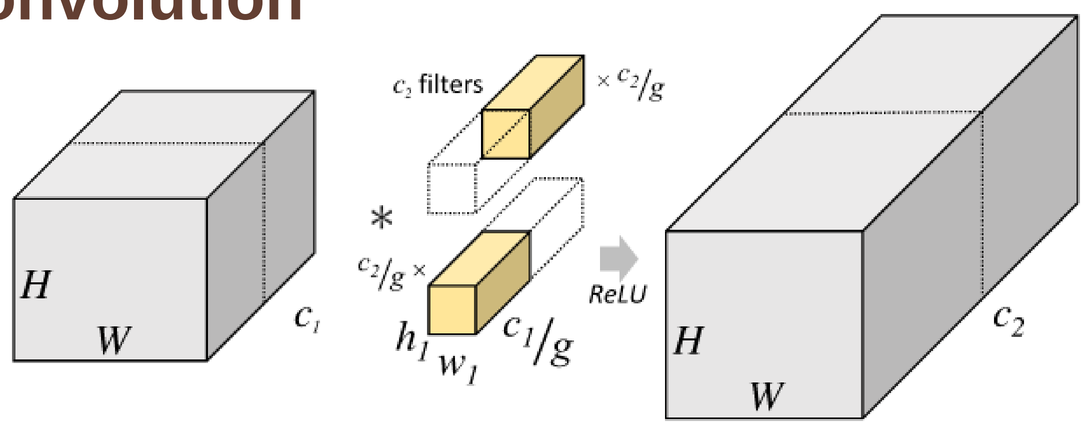
Other Models

ShuffleNet

- **Regular Convolution**



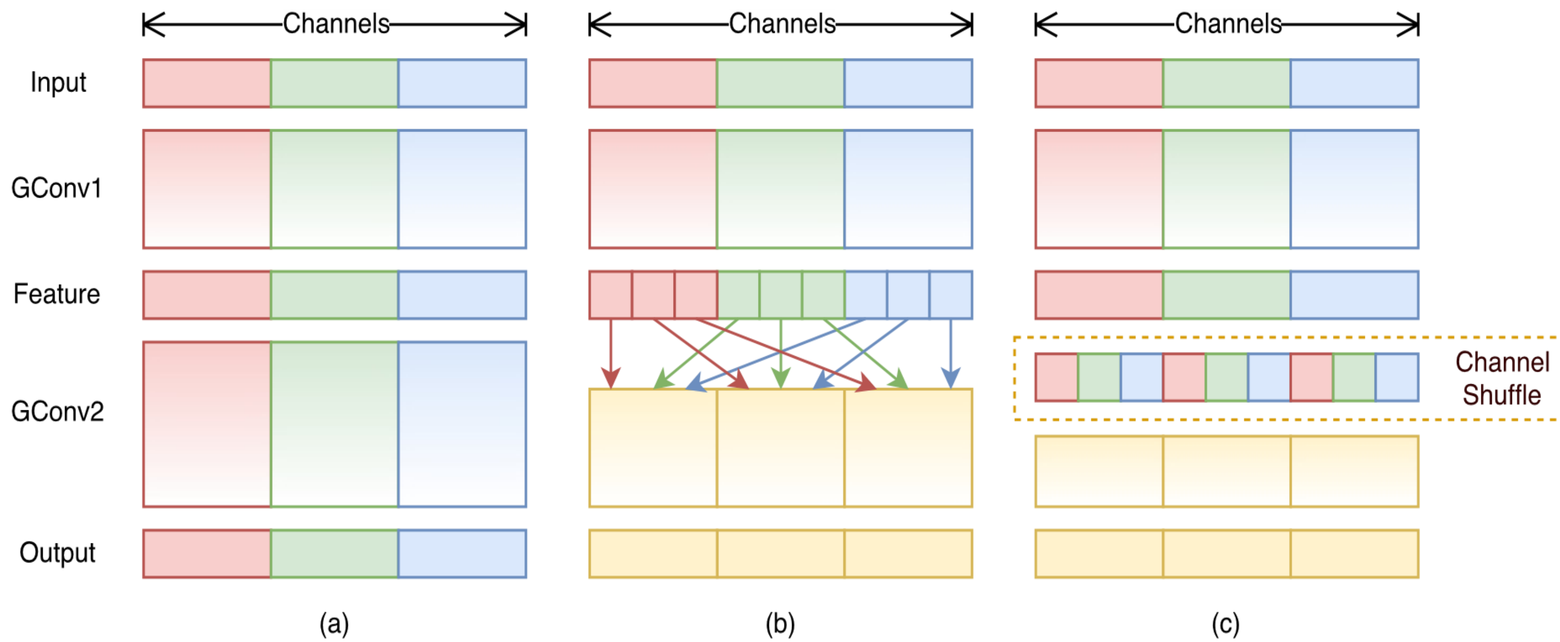
- **Group Convolution**



https://blog.yani.io/filter-group-tutorial/

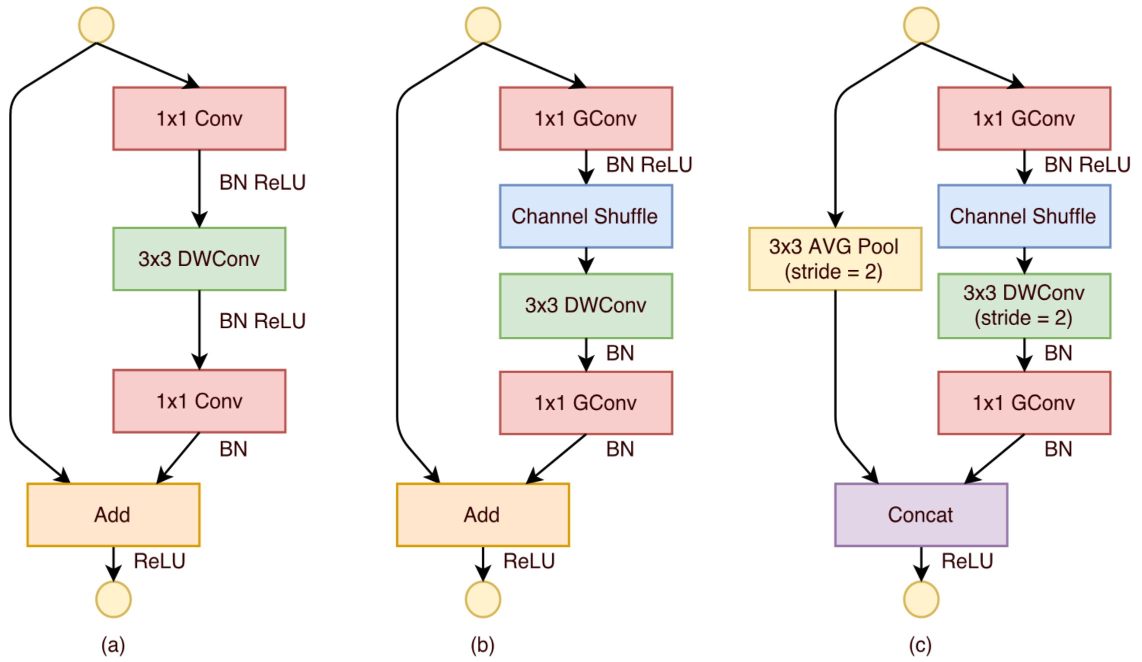
ShuffleNet

- ## Channel Shuffling

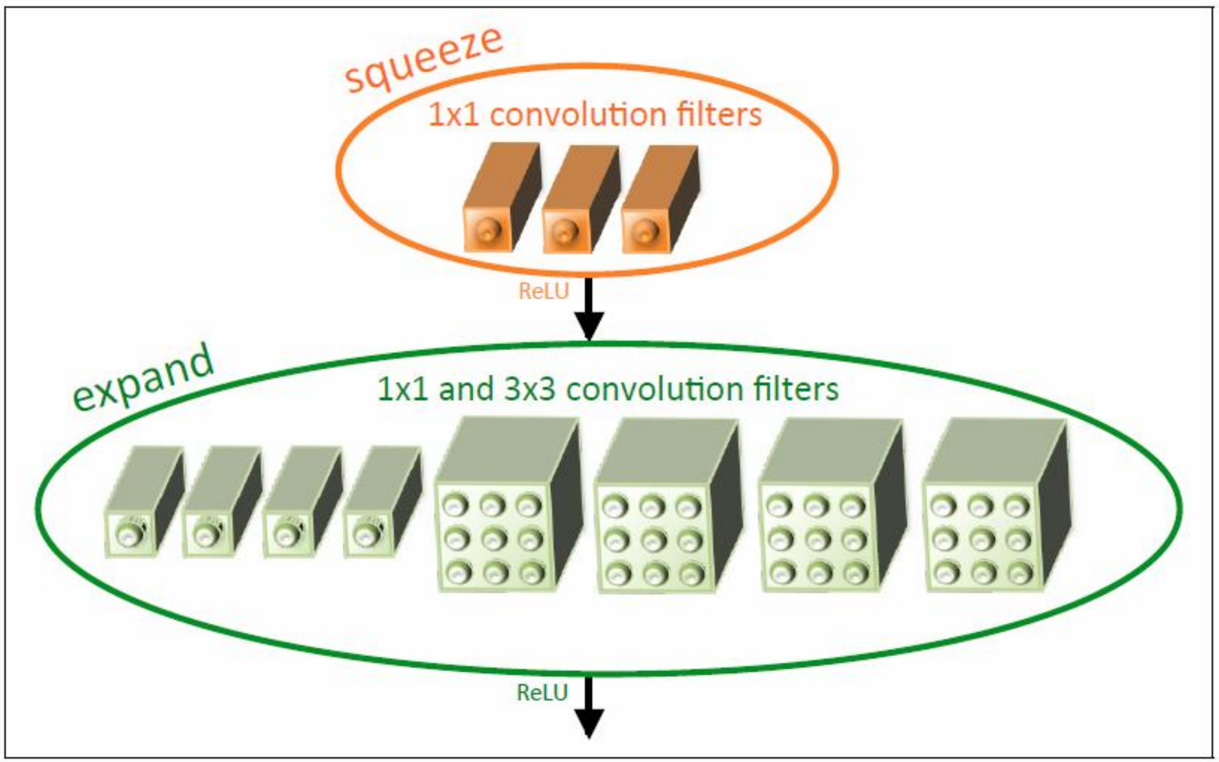


ShuffleNet

- ## Blocks

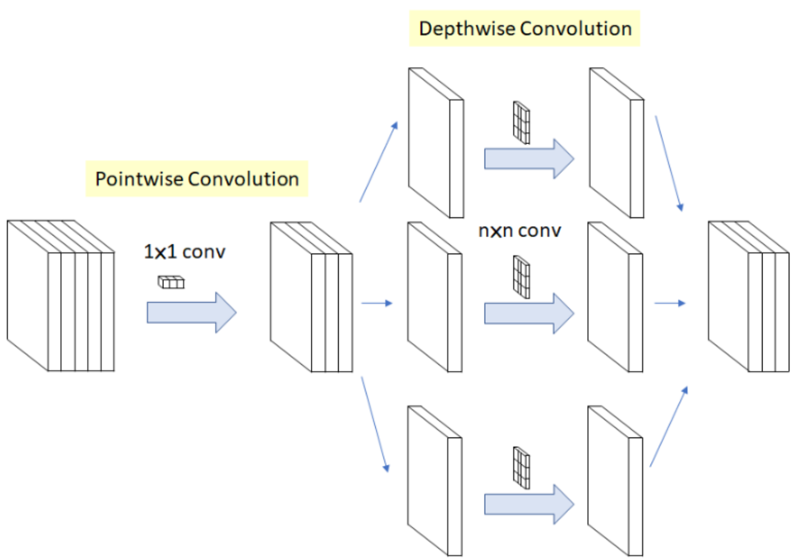


SqueezeNet

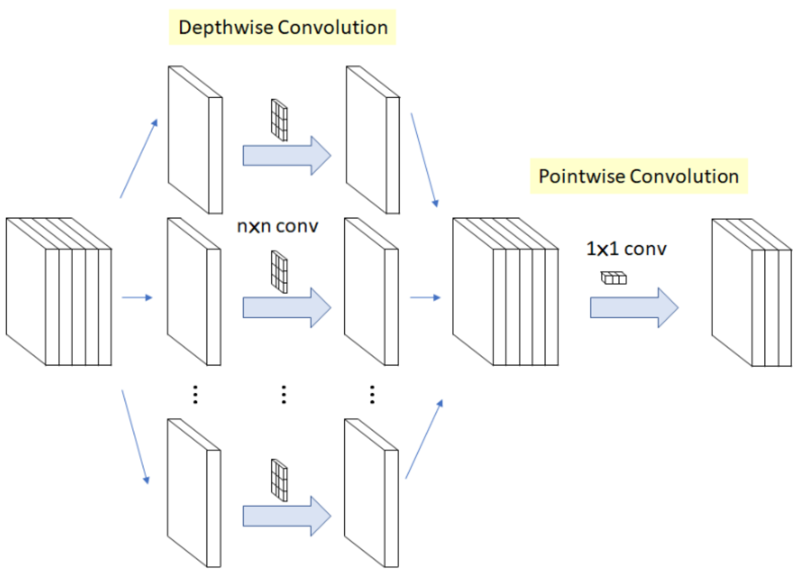

squeeze
1x1 convolution filters
ReLU
expand
1x1 and 3x3 convolution filters
ReLU

Xception

Xception



Depthwise Separable Convolution



Reference

A. Howard et al, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv 2017

M. Sandler et al, MobileNetV2: Inverted Residuals and Linear Bottlenecks, arXiv 2018

X. Zhang et al, ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices, arXiv 2017

F. Iandolar et al, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size, arXiv 2016

F. Chollet, Xception: Deep Learning with Depthwise Separable Convolutions, arXiv 201

Question and Answer