Designing Efficient AI Model

Chia-Chi Tsai (蔡家齊)
cctsai@gs.ncku.edu.tw
AI System Lab
Department of Electrical Engineering
National Cheng Kung University

Outline

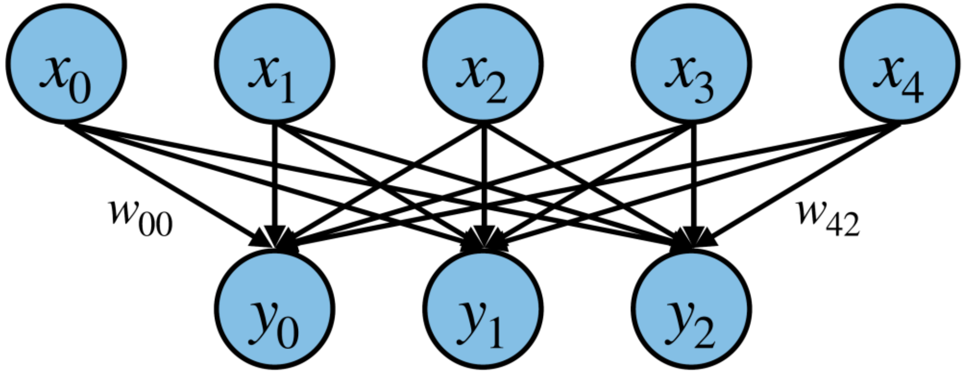
- Basic Concepts
- Manual Designed Neural Networks

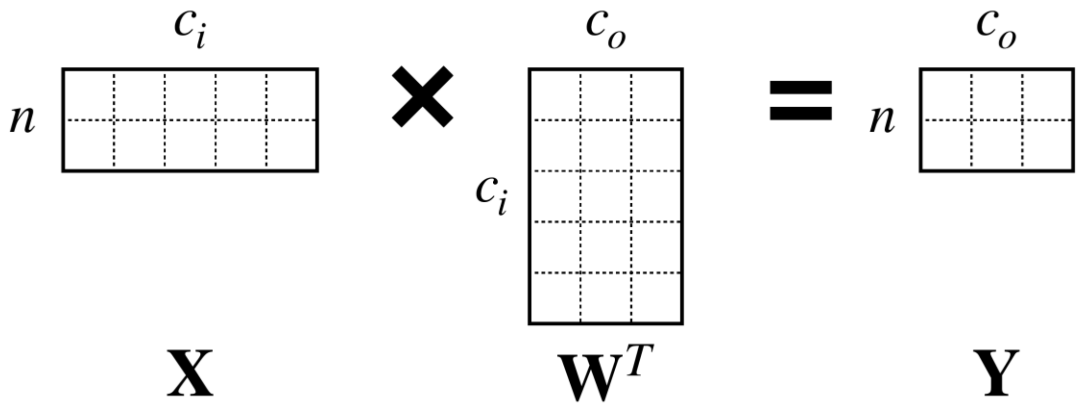
Outline

- Basic Concepts
- Manual Designed Neural Networks

Number of MAC Operations

- Linear Layer





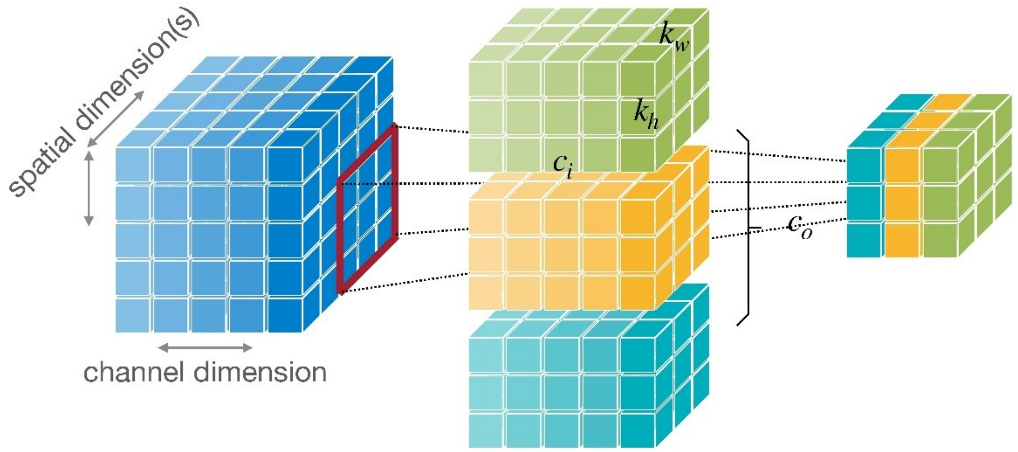
Layer Type	MACs (batch size n=1)
Linear Layer	$c_o \times c_i$
Convolution	
Grouped Convolution	
Depthwise Convolution	
1x1 Convolution	

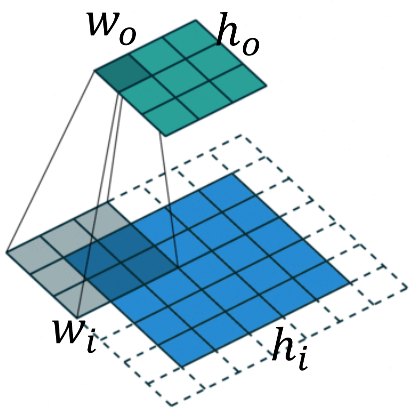
Notation	
n	Batch size
c_i, c_o	Input/output channel
w_i, w_o	Input/output width
h_i, h_o	Input/output height
k_w, k_h	Kernel width/height
g	Groups

Number of MAC Operations

- Convolution

Layer Type	MACs (batch size n=1)
Linear Layer	$c_o \times c_i$
Convolution	$c_i \times k_h \times k_w \times h_o \times w_o \times c_o$
Grouped Convolution	
Depthwise Convolution	
1x1 Convolution	



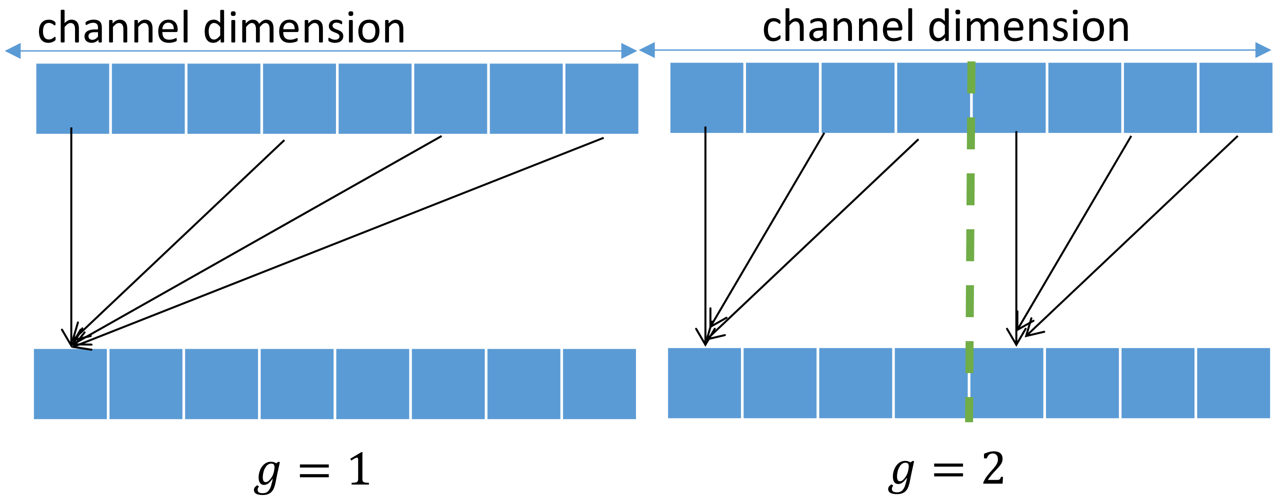


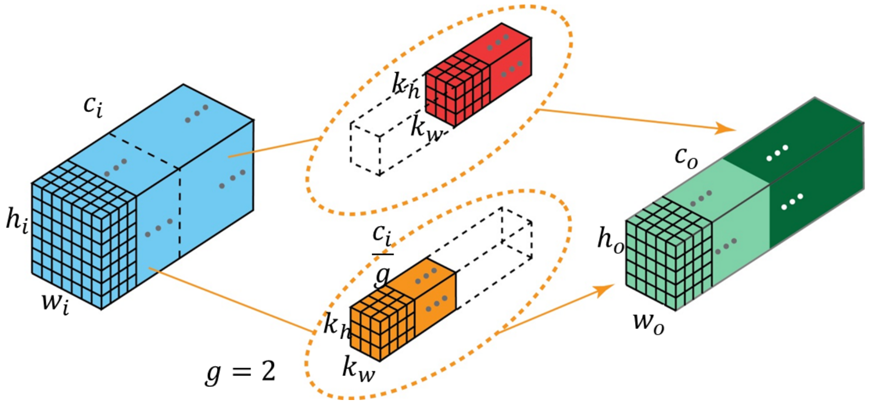
Notation	
n	Batch size
c_i, c_o	Input/output channel
w_i, w_o	Input/output width
h_i, h_o	Input/output height
k_w, k_h	Kernel width/height
g	Groups

Number of MAC Operations

- Grouped Convolution

Layer Type	MACs (batch size n=1)
Linear Layer	$c_o \times c_i$
Convolution	$c_i \times k_h \times k_w \times h_o \times w_o \times c_o$
Grouped Convolution	$\frac{c_i}{g} \times k_h \times k_w \times h_o \times w_o \times c_o$
Depthwise Convolution	
1x1 Convolution	



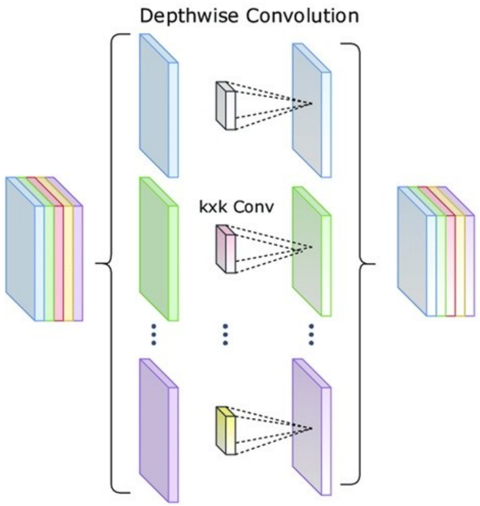


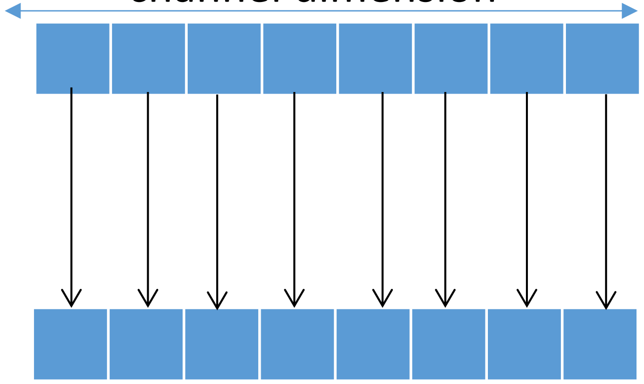
Notation	
n	Batch size
c_i, c_o	Input/output channel
w_i, w_o	Input/output width
h_i, h_o	Input/output height
k_w, k_h	Kernel width/height
g	Groups

Number of MAC Operations

- Depthwise Convolution

Layer Type	MACs (batch size n=1)
Linear Layer	$c_o \times c_i$
Convolution	$c_i \times k_h \times k_w \times h_o \times w_o \times c_o$
Grouped Convolution	$\frac{c_i}{g} \times k_h \times k_w \times h_o \times w_o \times c_o$
Depthwise Convolution	$k_h \times k_w \times h_o \times w_o \times c_o$
1x1 Convolution	





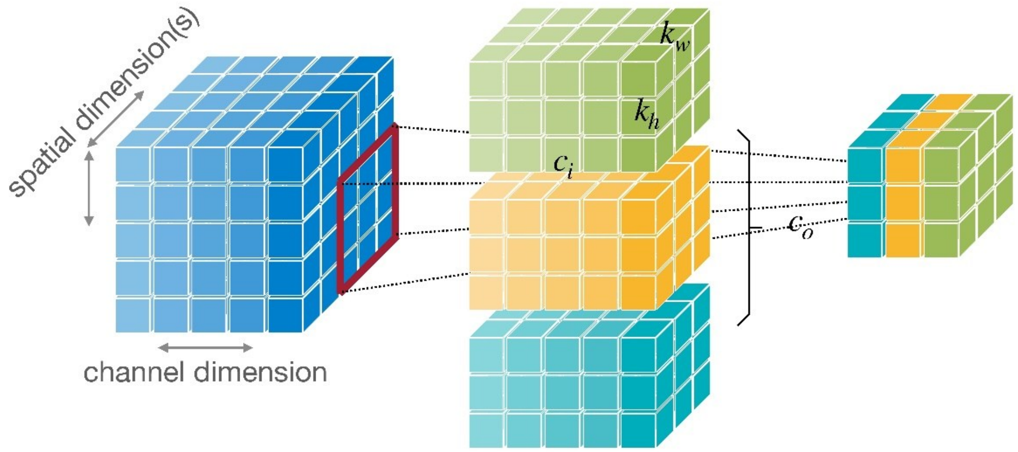
$g = c_i = c_o$

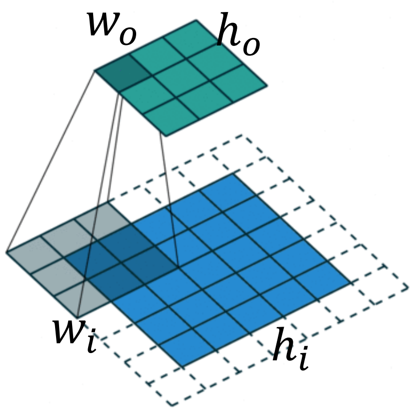
Notation	
n	Batch size
c_i, c_o	Input/output channel
w_i, w_o	Input/output width
h_i, h_o	Input/output height
k_w, k_h	Kernel width/height
g	Groups

Number of MAC Operations

- Convolution

Layer Type	MACs (batch size n=1)
Linear Layer	$c_o \times c_i$
Convolution	$c_i \times k_h \times k_w \times h_o \times w_o \times c_o$
Grouped Convolution	$\frac{c_i}{g} \times k_h \times k_w \times h_o \times w_o \times c_o$
Depthwise Convolution	$k_h \times k_w \times h_o \times w_o \times c_o$
1x1 Convolution	$c_o \times c_i \times h_o \times w_o$





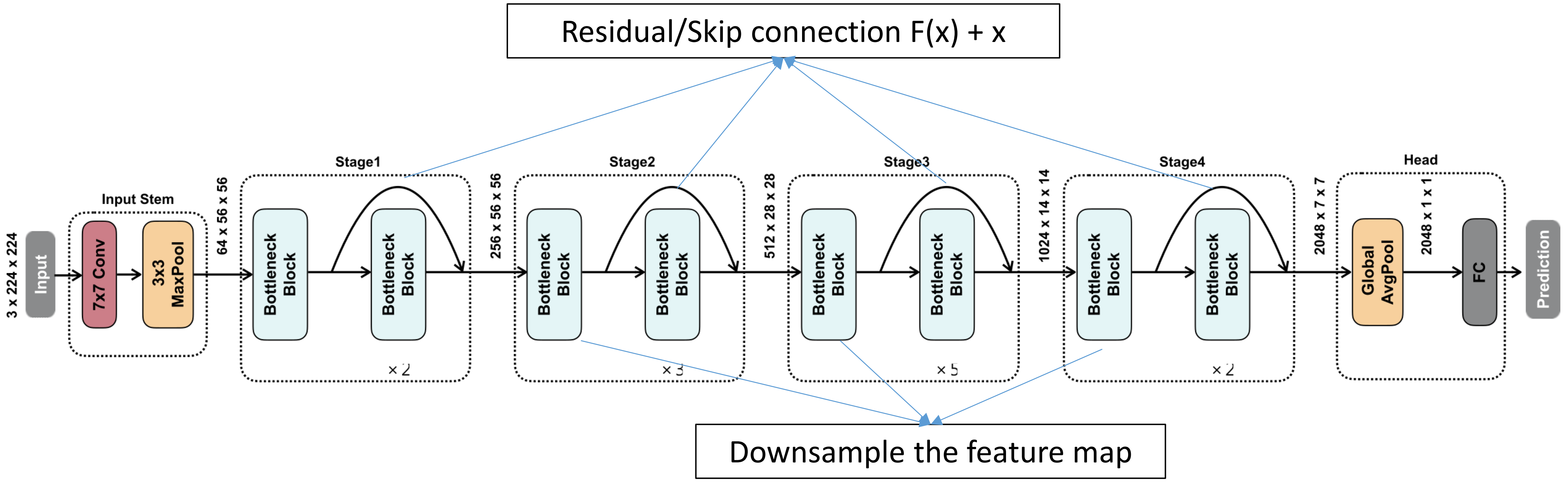
Notation	
n	Batch size
c_i, c_o	Input/output channel
w_i, w_o	Input/output width
h_i, h_o	Input/output height
k_w, k_h	Kernel width/height
g	Groups

Stages of Model

- A neural network architecture typically consists
  - Input stem,
  - Head
  - Several stages
- Early stages have larger feature map sizes, so we need to keep the width small to reduce the cost.
- In contrast, late stages have smaller feature map sizes so that we can increase the width.

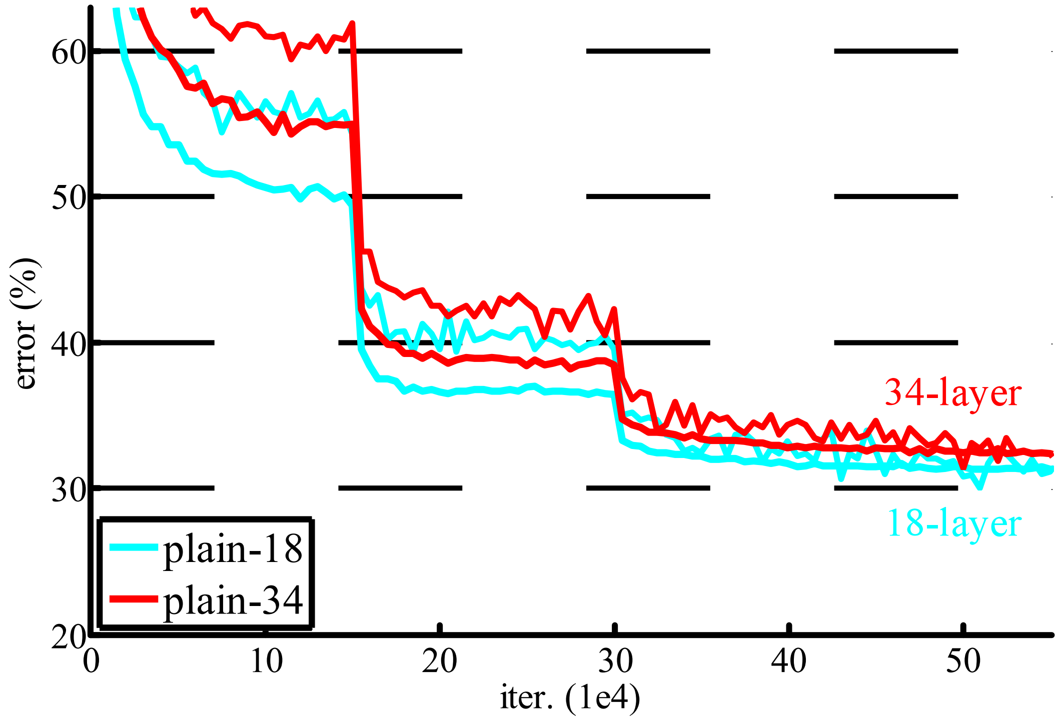
Stages of Model

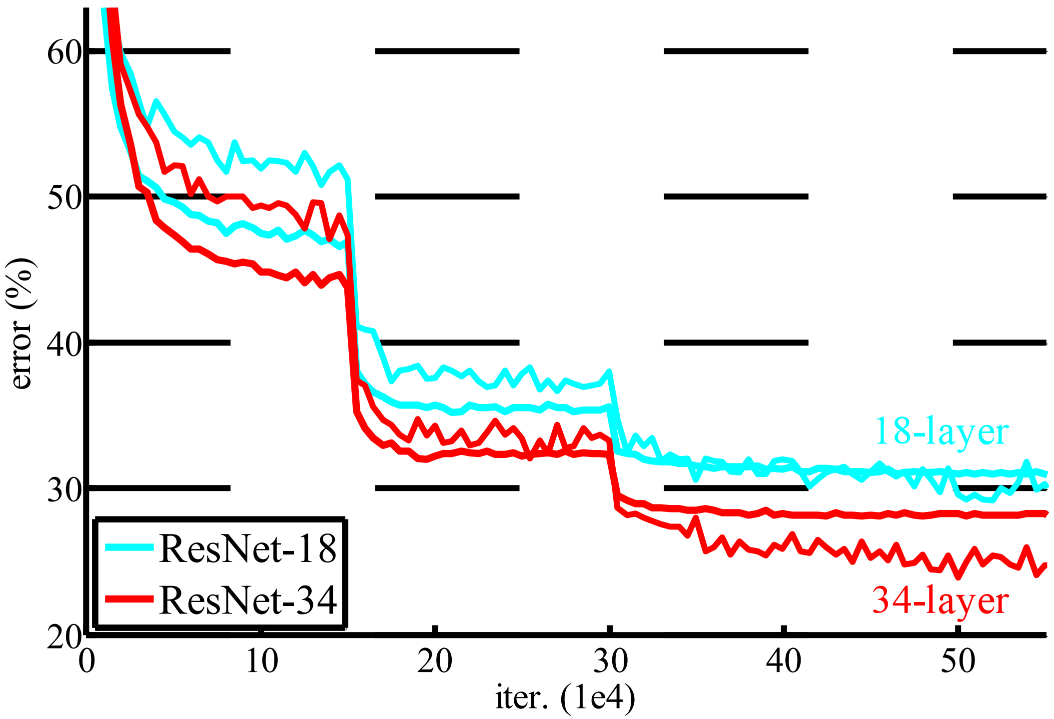
- Downsample
  - Feature map downsampling is usually done at the first block in each stage via stride convolution or pooling
- Residual/Skip connection
  - We can add residual/skip connections for the remaining blocks as their input and output
  - dimensions are the same



Basic Concepts

- Residual/Skip connection





Outline

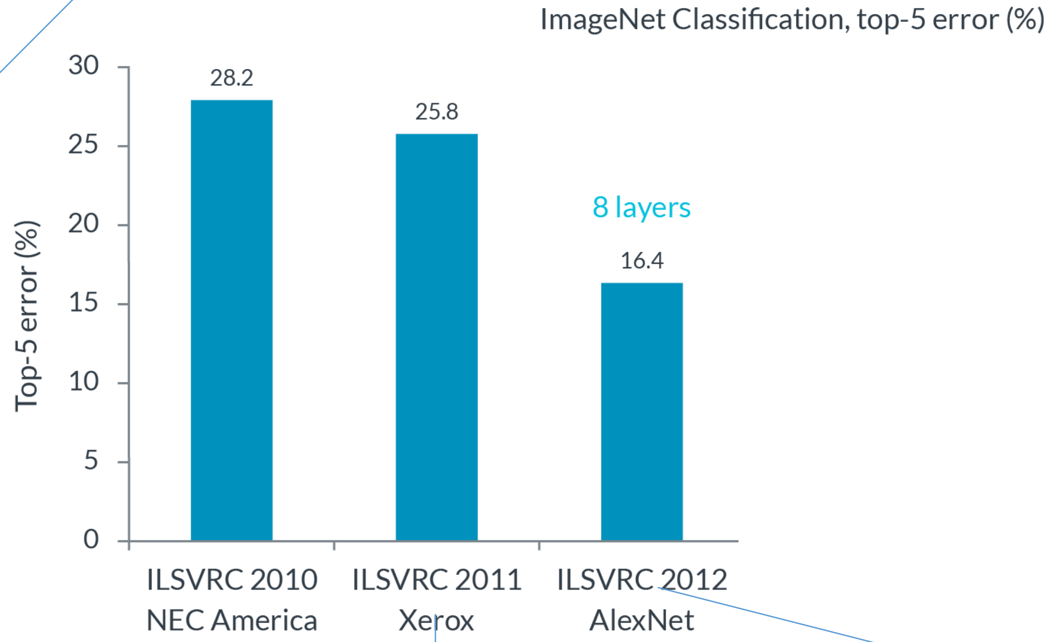
- Basic Concepts
- Manual Designed Neural Networks

AlexNet

- Remarkable improvements over previous non-DL methods

Image (3×224×224)
11×11 Conv, channel 96, stride 4, pad 2
3×3 MaxPool, stride 2
5×5 Conv, channel 256, pad 2, groups 2
3×3 MaxPool, stride 2
3×3 Conv, channel 384, pad 1
3×3 Conv, channel 384, pad 1, groups 2
3×3 Conv, channel 256, pad 1, groups 2
3×3 MaxPool, stride 2
Linear, channel 4096
Linear, channel 4096
Linear, channel 1000

Large kernel convolution in early stages



Non-DL method

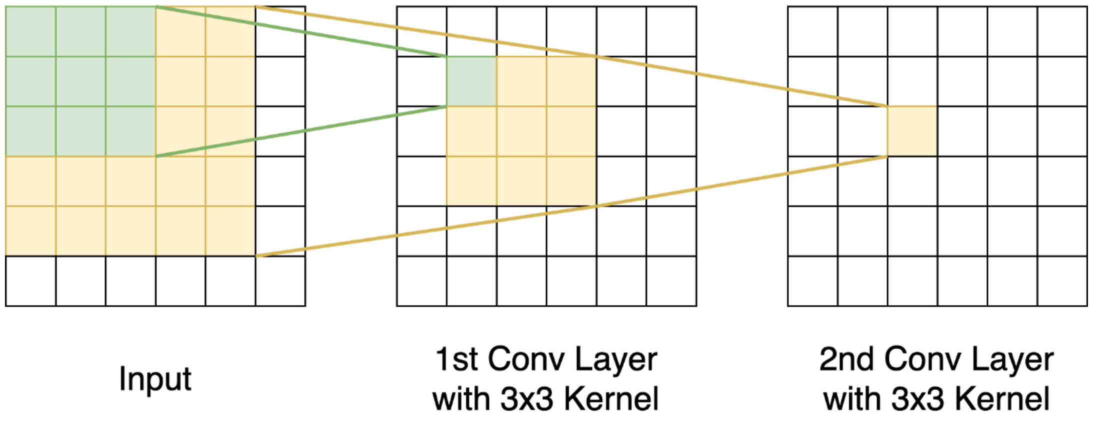
DL method

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

VGG

Image (3×224×224)
3x3 Conv, channel 64, pad 1
3x3 Conv, channel 64, pad 1
2x2 MaxPool, stride 2
3x3 Conv, channel 128, pad 1
3x3 Conv, channel 128, pad 1
2x2 MaxPool, stride 2
3x3 Conv, channel 256, pad 1
3x3 Conv, channel 256, pad 1
3x3 Conv, channel 256, pad 1
2x2 MaxPool, stride 2
3x3 Conv, channel 512, pad 1
3x3 Conv, channel 512, pad 1
3x3 Conv, channel 512, pad 1
2x2 MaxPool, stride 2
3x3 Conv, channel 512, pad 1
3x3 Conv, channel 512, pad 1
3x3 Conv, channel 512, pad 1
2x2 MaxPool, stride 2
Linear, channel 4096
Linear, channel 4096
Linear, channel 1000

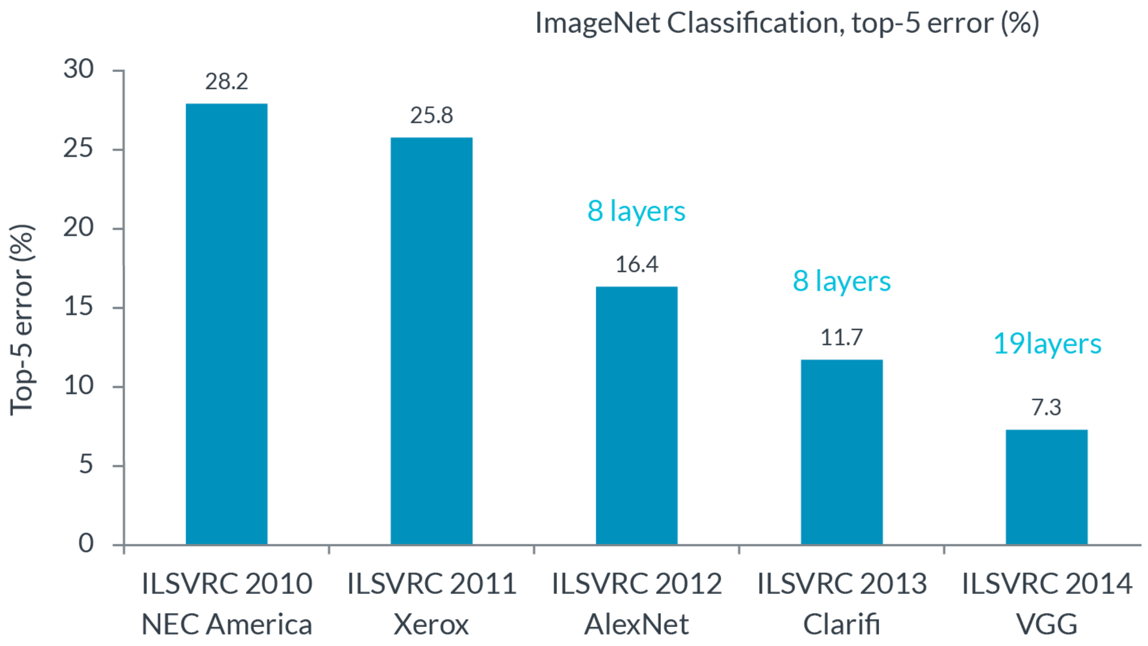
- Stacking multiple 3x3 convolution layers
- Different from AlexNet, VGG only uses 3x3 convolution. Meanwhile, VGG
- Stacks more layers to maintain a large receptive field.
  - The computational cost of two 3x3 convolutions is smaller than one 5x5 convolution: 3x3 + 3x3 = 18 < 5x5



K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representations, May 2015. 3, 4

VGG

- VGG: performance and cost breakdown

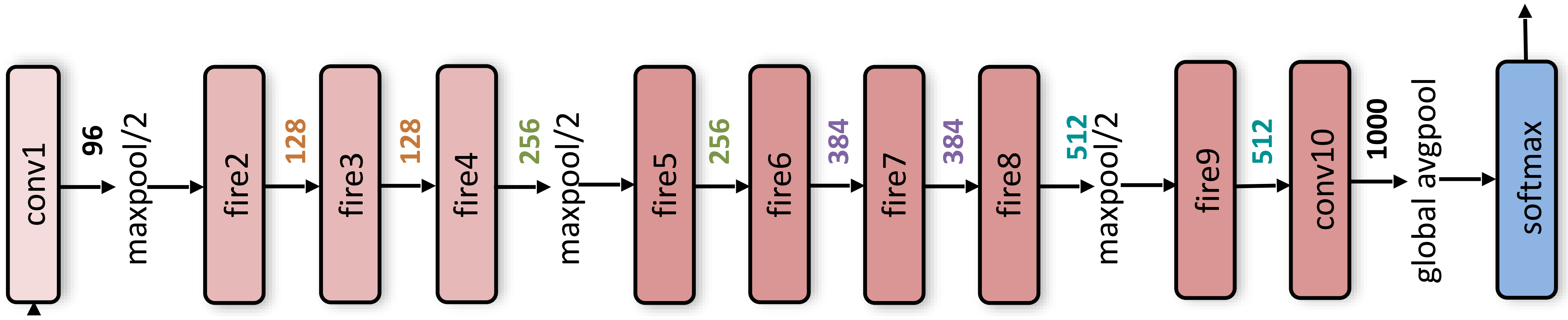


Layer	Weights	FLOP
Conv1_1	2K	0.2B
Conv1_2	37K	3.7B
Conv2_1	74K	1.8B
Conv2_2	148K	3.7B
Conv3_1	295K	1.8B
Conv3_2	590K	3.7B
Conv3_3	590K	3.7B
Conv4_1	1M	1.8B
Conv4_2	2M	3.7B
Conv4_3	2M	3.7B
Conv5_1	2M	925M
Conv5_2	2M	925M
Conv5_3	2M	925M
Fc6	103M	206M
Fc7	17M	34M
Fc8	4M	8M
Total	**138M**	**30.9B**

3x3 convolution layers are the key computational bottleneck

SqueezeNet

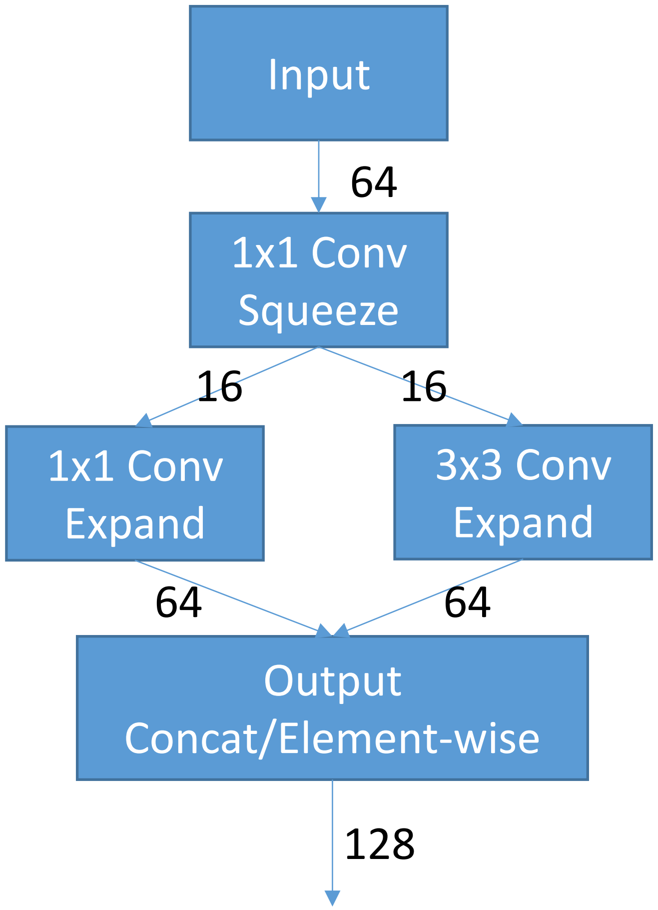
- Replace 3x3 convolution with fire modules
- Use global average pooling in the head to reduce the cost of the head



Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. *arXiv preprint arXiv:1602.07360*.

Fire Module

- Reduce the cost by using 1x1 convolution
  - Replace some 3x3 filters with 1x1 filters
  - Decrease the number of input channels to 3x3 fiters.

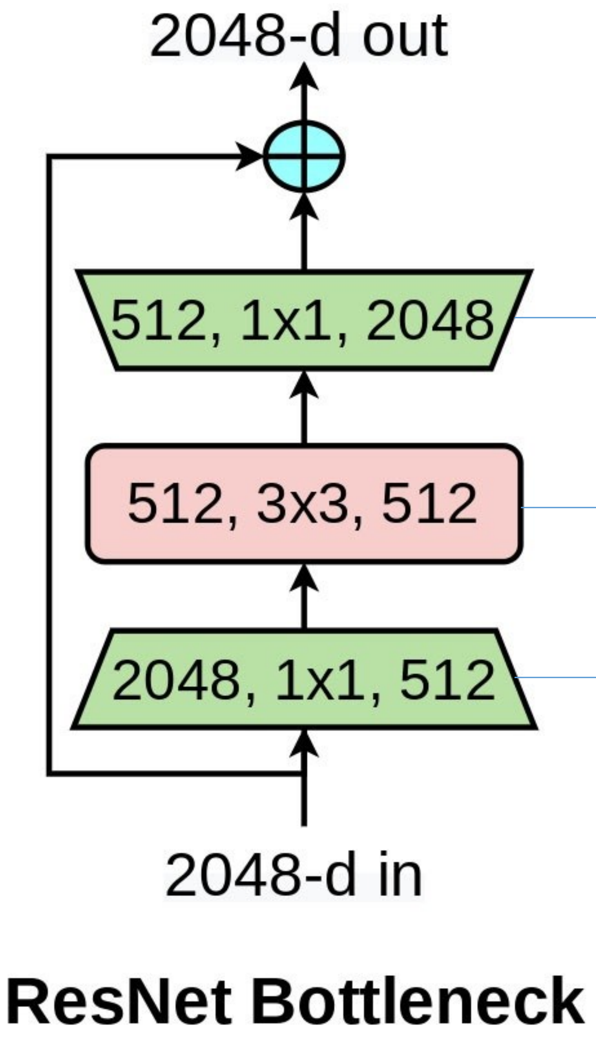


SqueezeNet

- AlexNet-level accuracy with 50x-510x smaller model size

CNN architecture	Compression Approach	Data Type	Original $\rightarrow$ Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD (Denton et al., 2014)	32 bit	240MB $\rightarrow$ 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning (Han et al., 2015b)	32 bit	240MB $\rightarrow$ 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression (Han et al., 2015a)	5-8 bit	240MB $\rightarrow$ 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB $\rightarrow$ 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB $\rightarrow$ 0.47MB	**510x**	57.5%	80.3%

ResNet50: Bottleneck Block



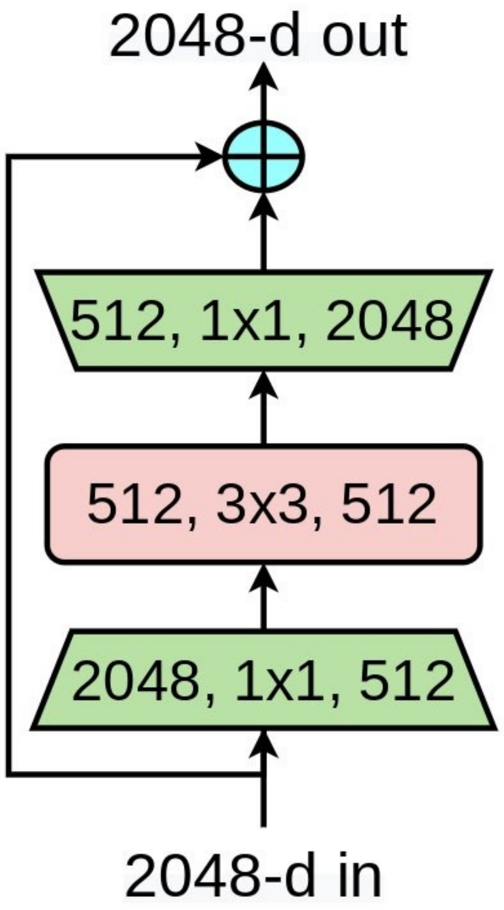
Expand the number of channels via 1x1 convolution

Feed reduced feature map to 3x3 convolution

Reduce the number of channels by 4x via 1x1 convolution

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

ResNet50: Bottleneck Block



ResNet Bottleneck

\# of MACs

$2048 \times 512 \times H \times W \times 1$

$512 \times 512 \times H \times W \times 9$

$2048 \times 512 \times H \times W \times 1$

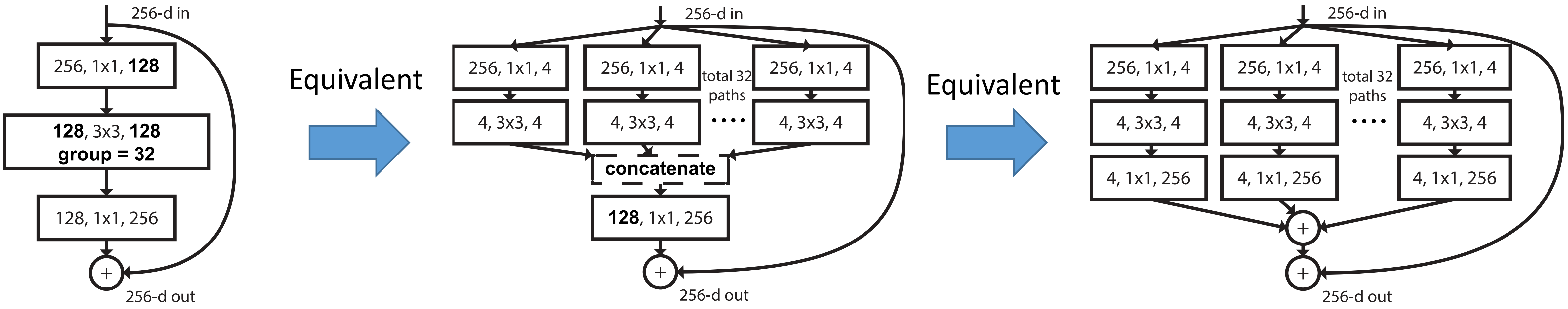
$$\boxed{512 \times 512 \times H \times W \times 17}$$

$$\boxed{\begin{aligned} &2048 \times 2048 \times H \times W \times 9 \\ &= 512 \times 512 \times H \times W \times 144 \end{aligned}}$$

8.5x reduction

ResNeXt: Grouped Convolution

- Replace 3x3 convolution with 3x3 grouped convolution
- Equivalent to a multi-path block



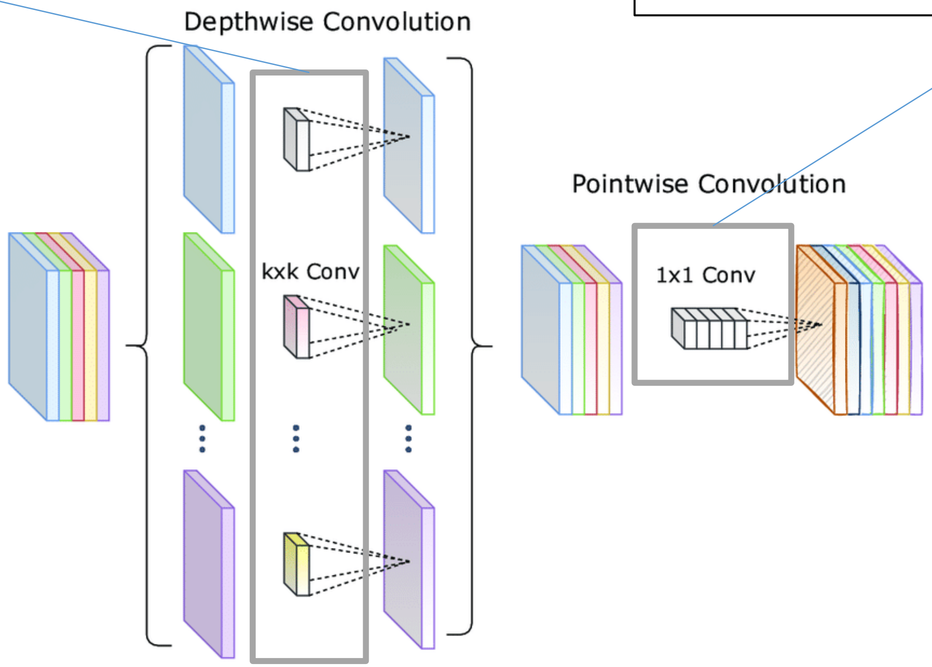
Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1492-1500).

MobileNet: Depthwise-Separable Block

- Depthwise convolution - An extreme case of group convolution
  - Where the group number equals the number of input channels

Use depthwise convolution to capture spatial information

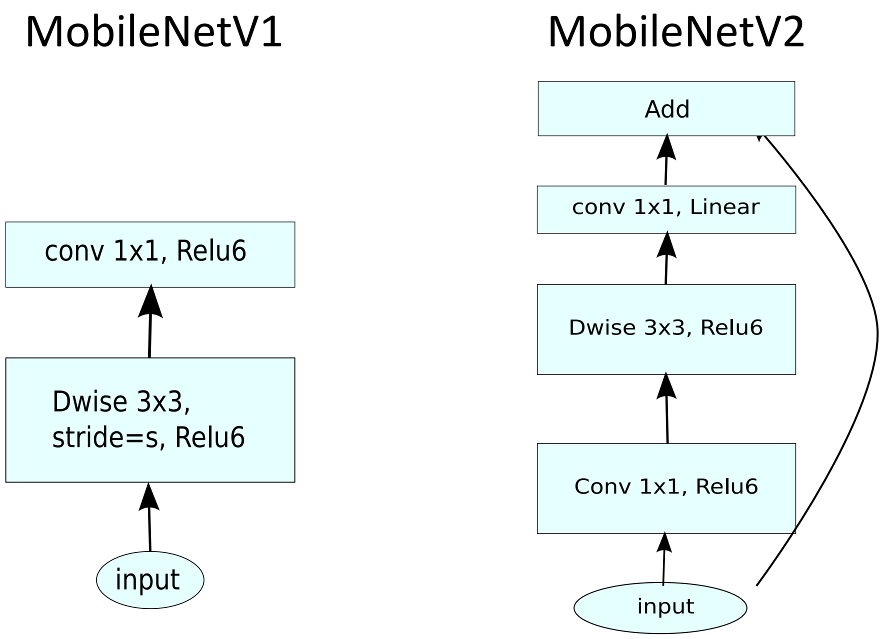
Use 1x1 convolution to fuse/exchange information across different channels



Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

MobileNetV2: Inverted Bottleneck Block

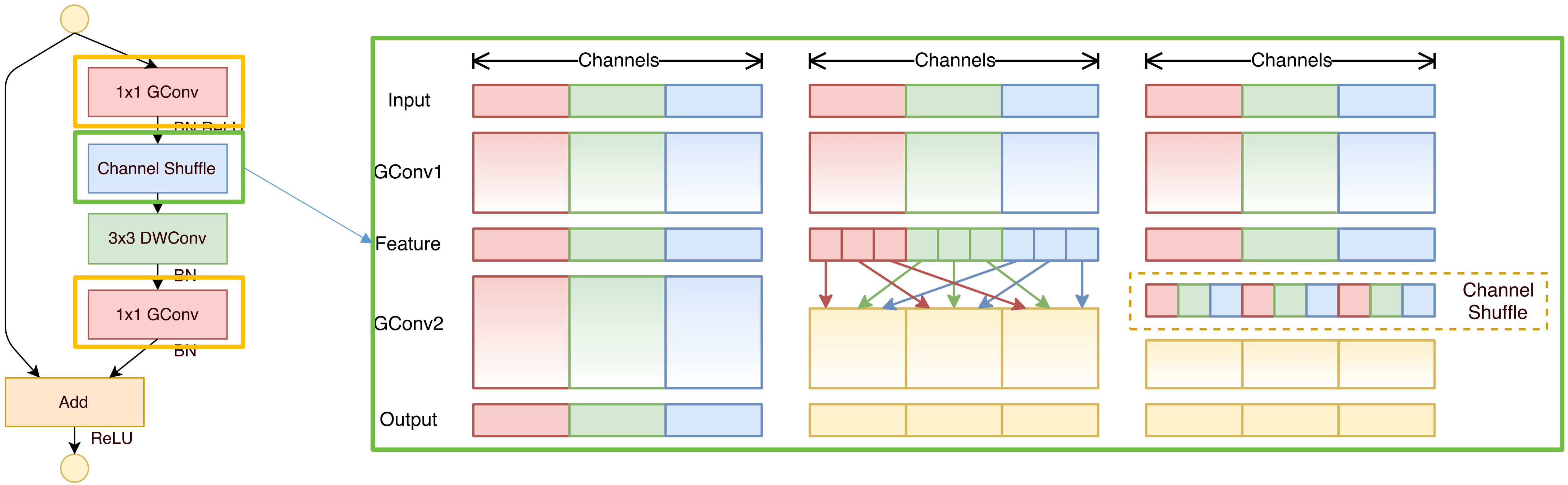
- Depthwise convolution has a much lower capacity compared to normal convolution
  - Increase the depthwise convolution's input and output channels to improve its capacity
  - Depthwise convolution's cost only grows linearly. Therefore, the cost is still affordable.



Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510-4520).

ShuffleNet

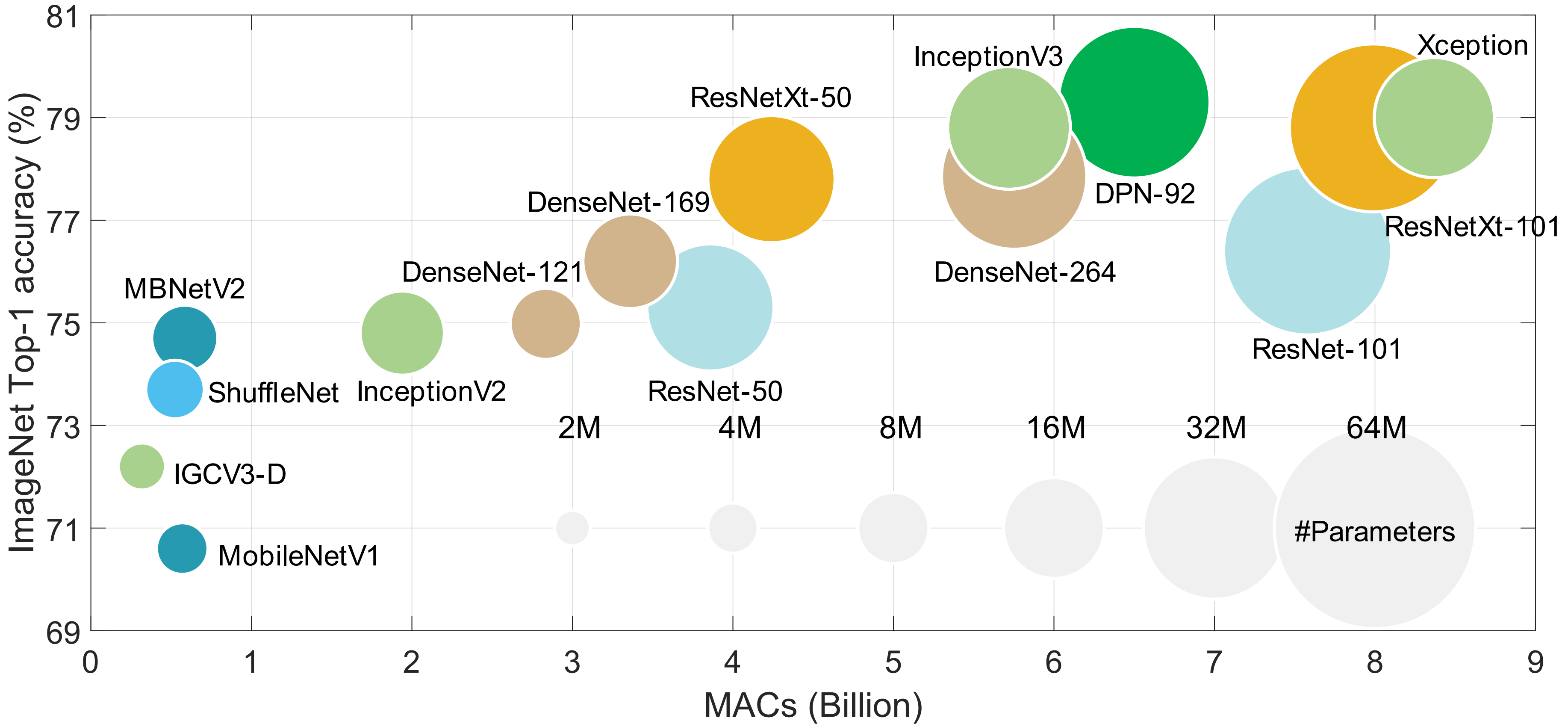
- 1x1 Group Convolution
  - Further reduce the cost by replacing 1x1 convolution with 1x1 group convolution
- Channel Shuffle
  - Exchange information across different groups via channel shuffle



Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6848-6856).

Accuracy-Efficiency Trade-off

- Huge design space, manual design is unscalable



From Manual Design to Automatic Design

- Automatic Architecture Search