國立高雄科技大學
National Kaohsiung University of Science and Technology

Deep Residual Network

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun
Microsoft Research

IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR), 2016

Speaker: Shih-Shinh Huang

June 24, 2021

Kaiming He, *et. al.*, "Deep Residual Learning for Image Classification", CVPR 2016

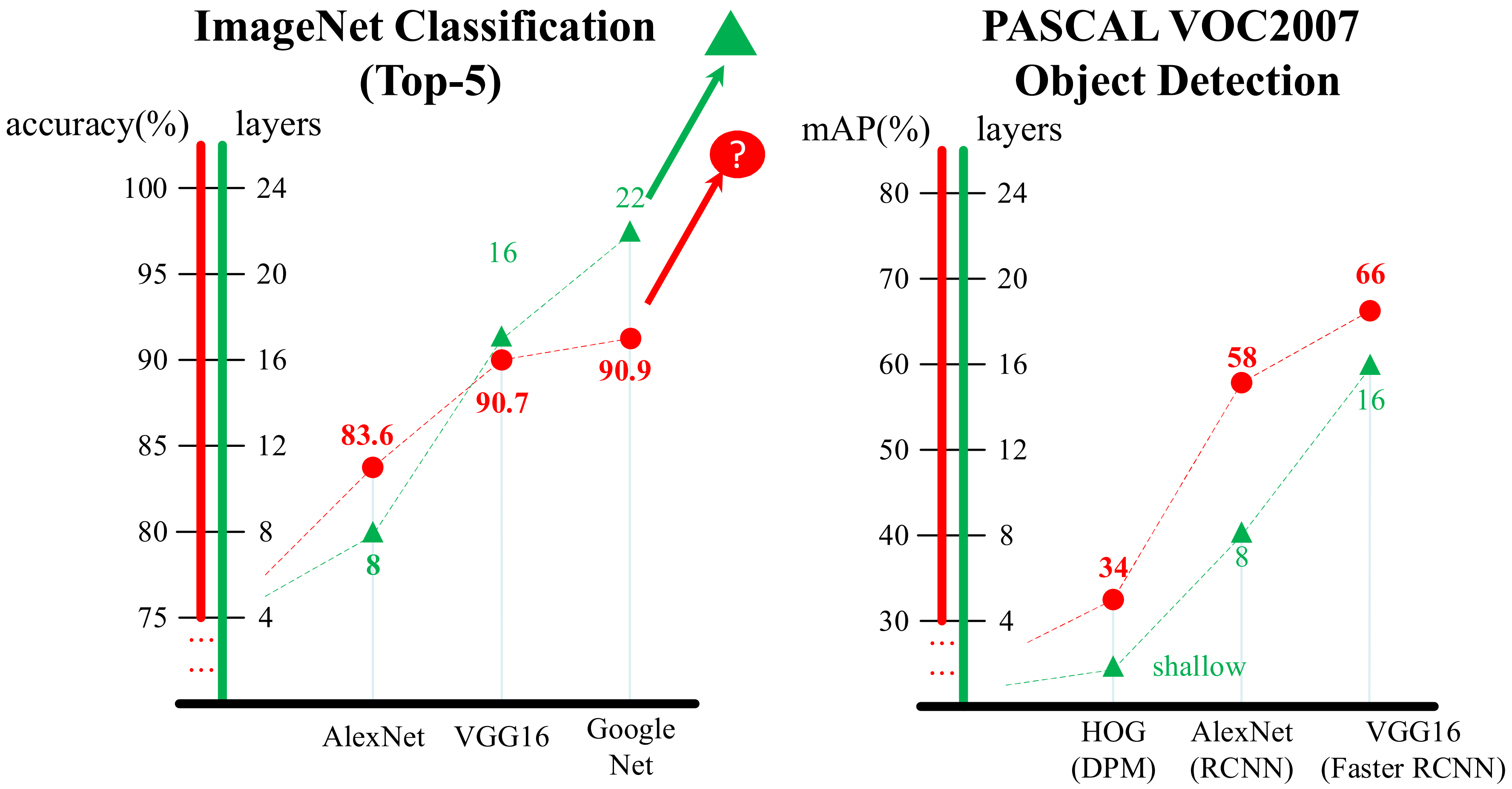
Outline

- Introduction
  - Evolution of Depth
  - Issue for More Layers
- Deep Residual Network
  - Main Idea
  - Residual Block
  - Network Architecture
- Deeper Residual Network
  - Bottleneck Residual Block
  - Network Architecture

Introduction

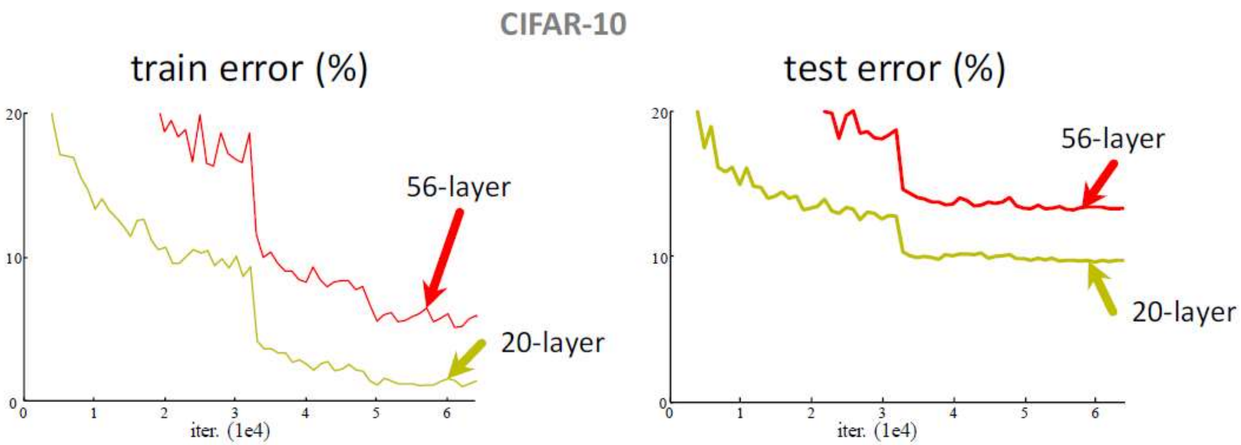
- Evolution of Depth
  - The levels of features can be enriched by the number of stacked layers (depth).
  - Network depth is of crucial importance in many visual applications
    - image classification
    - object detection
    - …….

Introduction



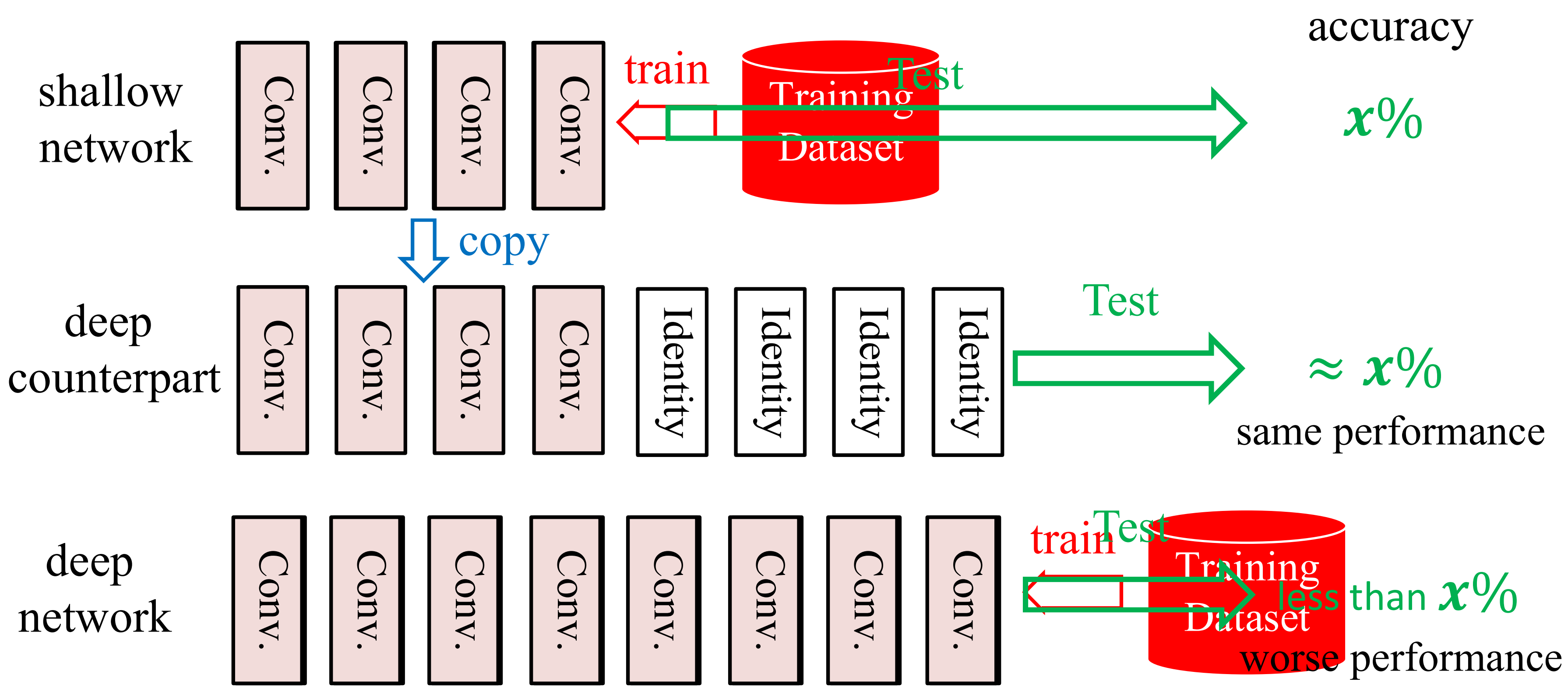
Introduction

- Issue for More Layers
  - degradation problem: accuracy gets saturated and then degrades.



- train error: 56-layer > 20-layer ⇨ not caused by overfitting
- test error: 56-layer > 20-layer

Introduction



Optimization Difficulty

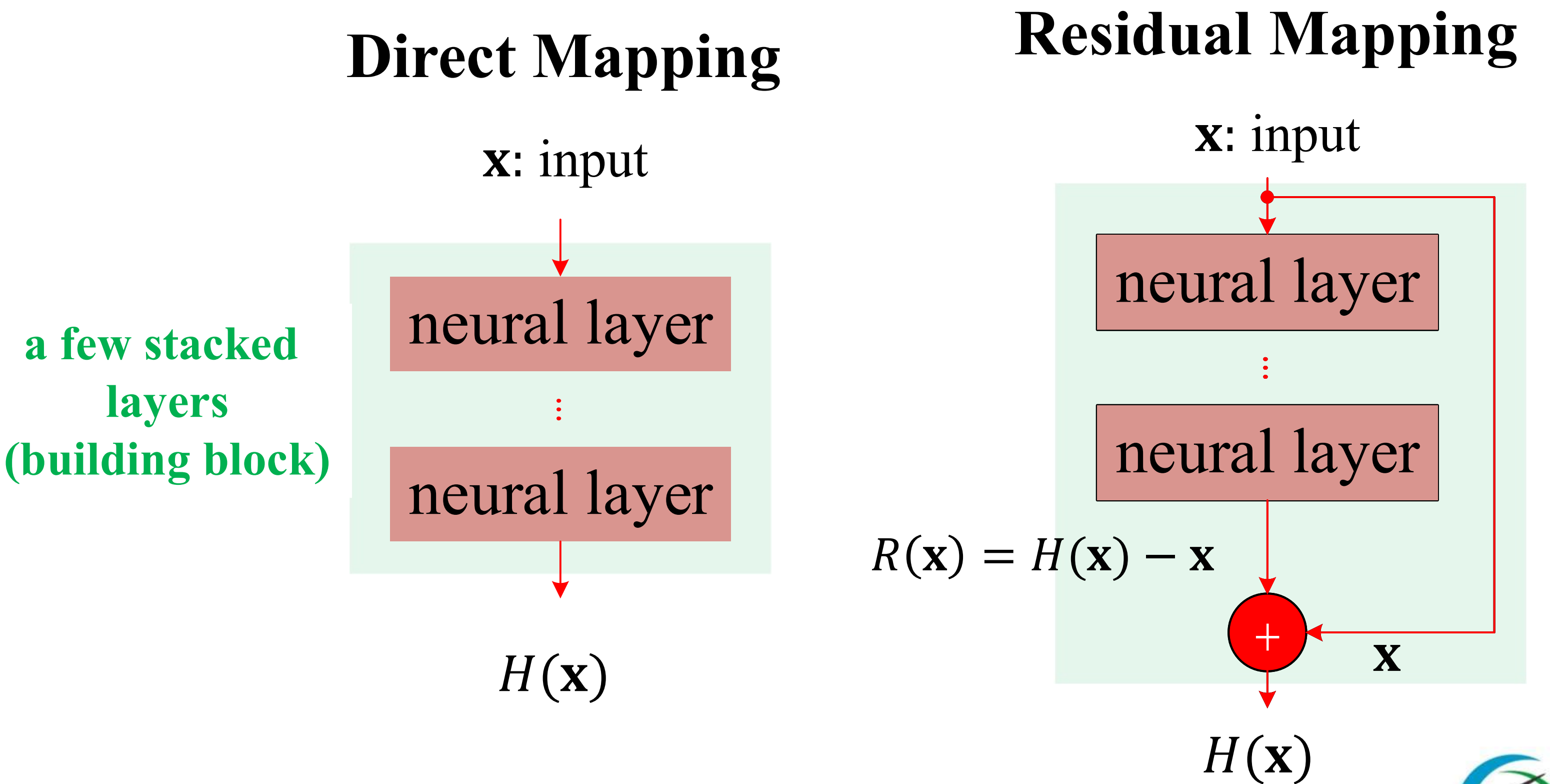
finding the solution is not easy when network goes deeper

Deep Residual Network

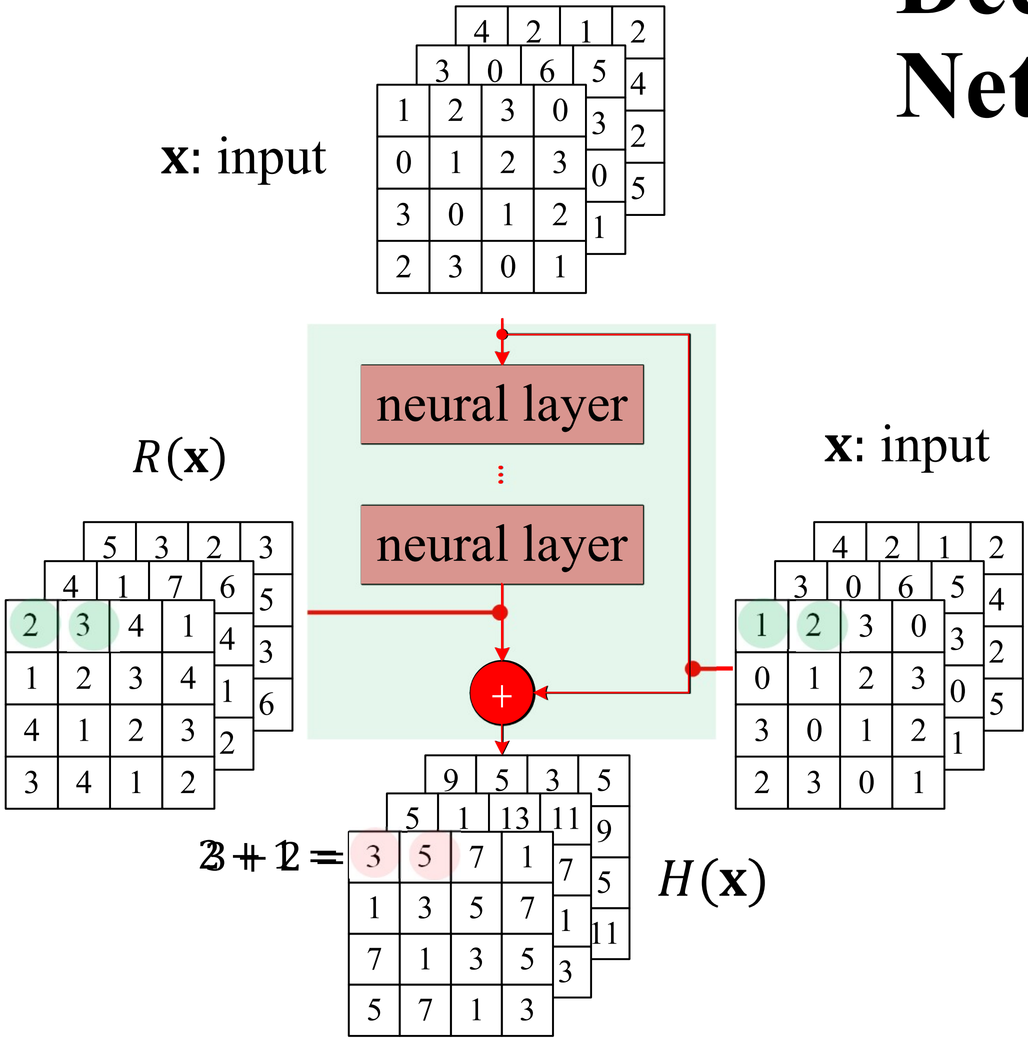
- Main Idea
  - assumption: input $\mathbf{x}$ and the target $H(\mathbf{x})$ is of the same sizes.
  - residual mapping: address degradation problem
    - approximate residual function $R(\mathbf{x}) = H(\mathbf{x}) - \mathbf{x}$ instead of target function $H(x)$
    - resume $H(x)$ by adding $\mathbf{x}$ back to $R(\mathbf{x})$

Deep Residual Network

- Main Idea



Residual Mapping



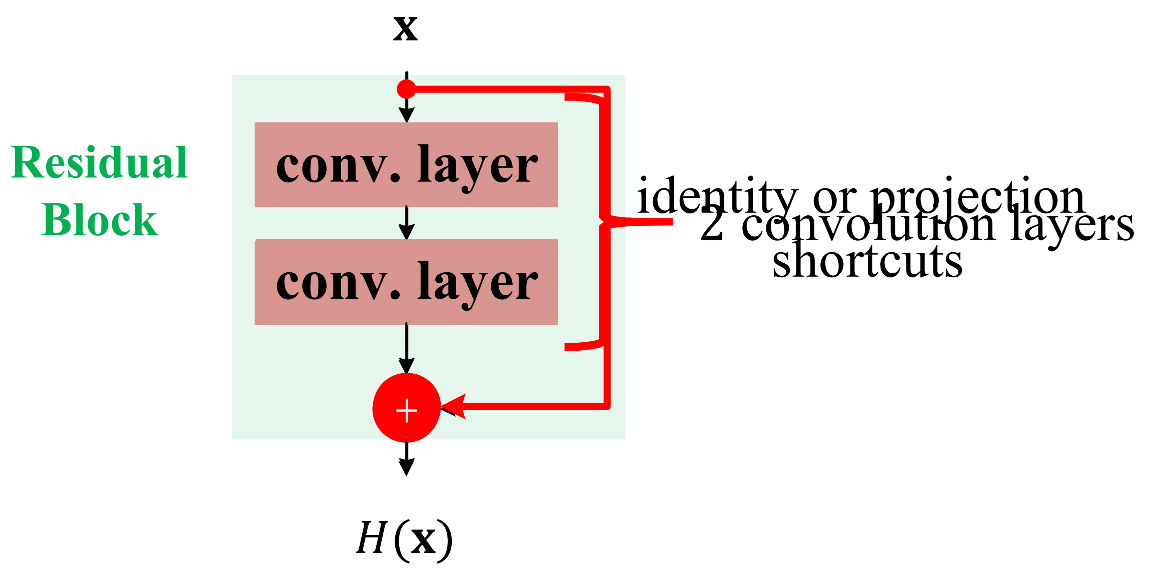
Deep Residual Network

- approximating $R(\mathbf{x})$ is easier than unreferenced $H(\mathbf{x})$
- the residual network can be trained by SGD end-to-end with back-propagation

SGD: stochastic gradient descent

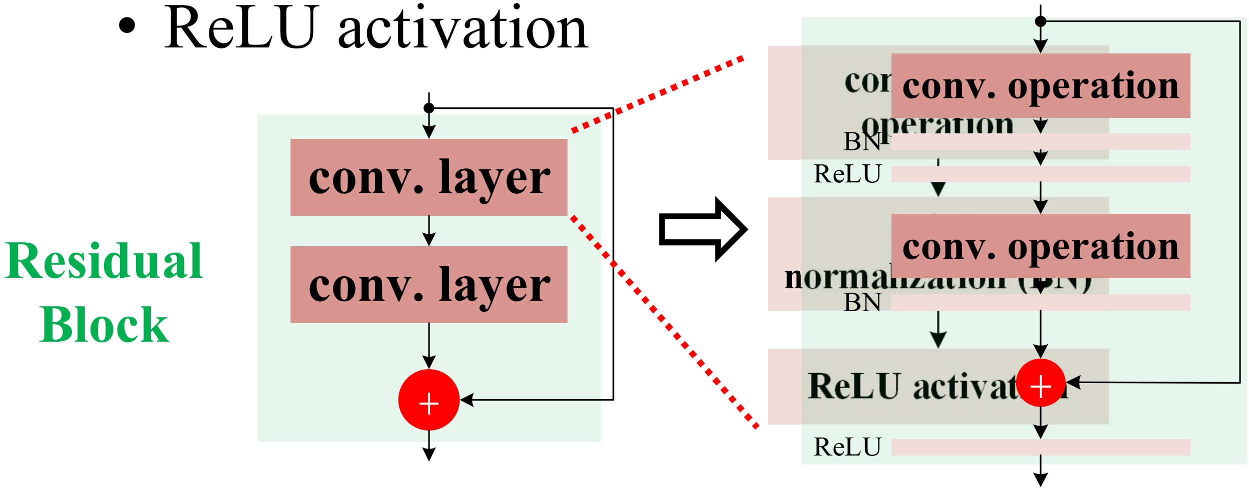
Deep Residual Network

- Residual Block
  - neural layers: 2 convolution layers
  - shortcut: identity or projection



Deep Residual Network

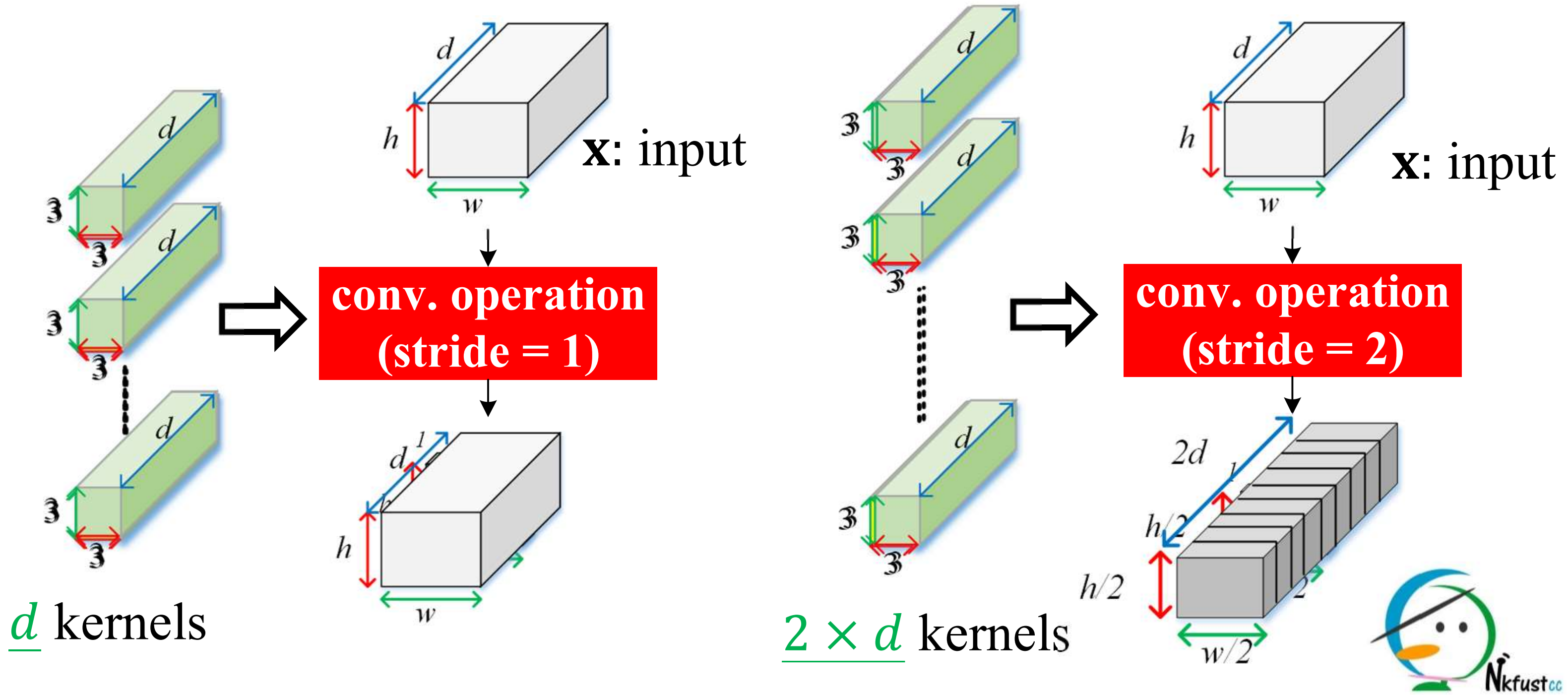
- Residual Block: convolution layer
  - A conv. layer consists of three operations
    - convolution operation
    - batch normalization (BN)
    - ReLU activation



S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariance Shift", ICML 2015

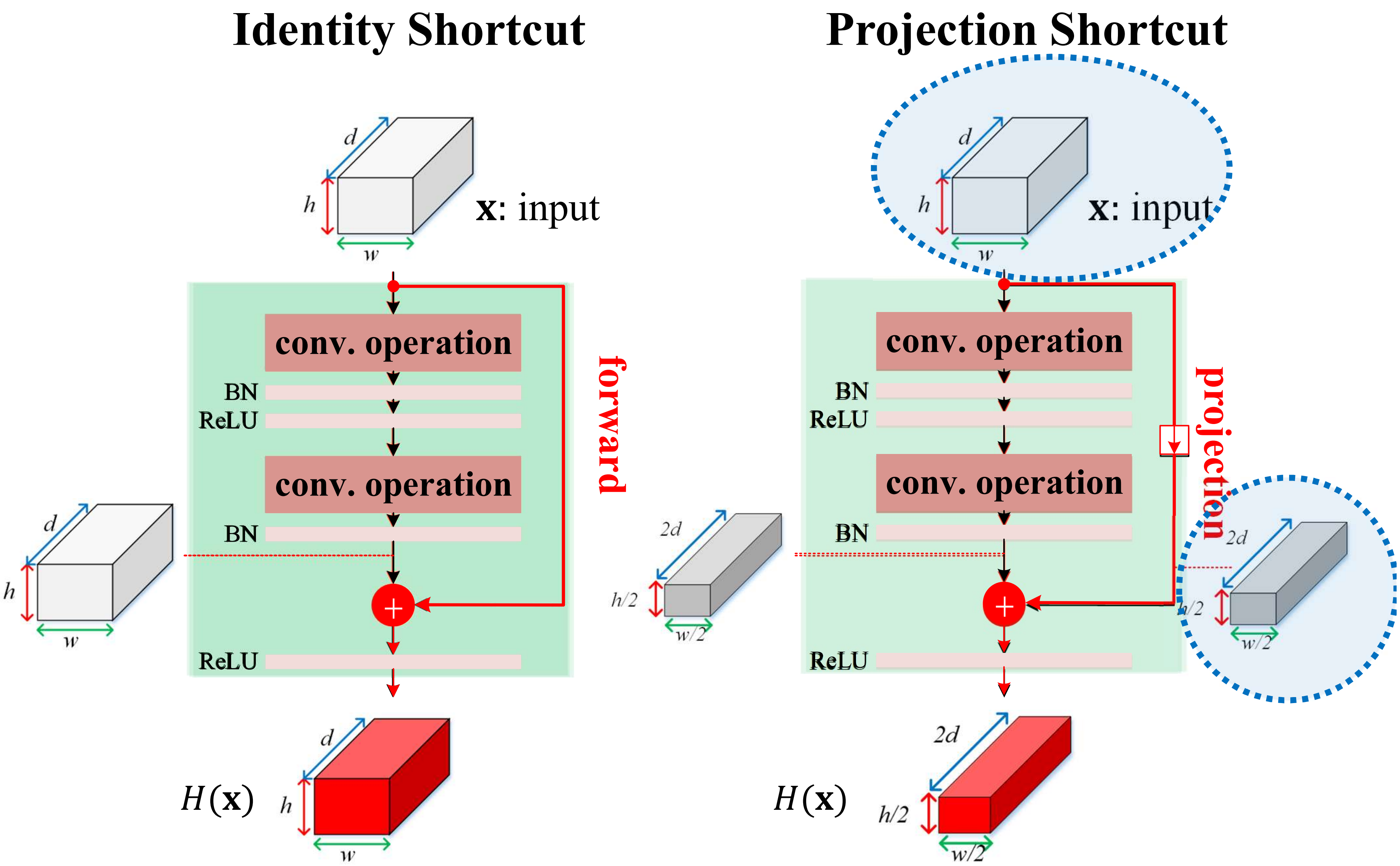
Deep Residual Network

- Residual Block: convolution layer
  - kernel size: almost 3 × 3
  - kernel no.: preserve computation complexity



Deep Residual Network

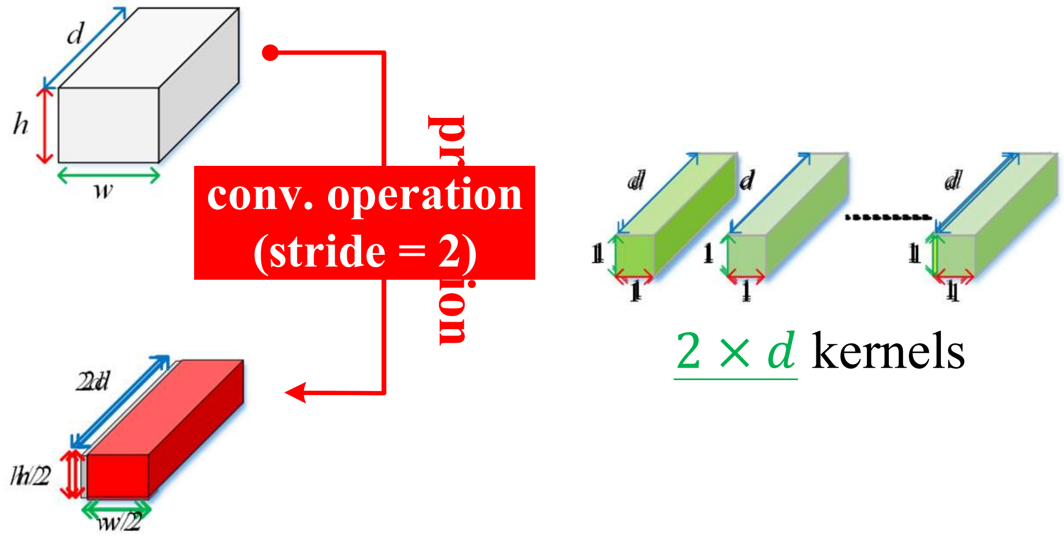
- Residual Block: shortcut connection
  - bypass the convolution operations
  - transform input to the feature map with the same size as output
    - identity shortcut: forward input directly if input and output feature maps are of the same sizes
    - projection shortcut: map input size to output size if their sizes are different



Identity Shortcut
Projection Shortcut
d
h
w
x: input
conv. operation
BN
ReLU
conv. operation
BN
ReLU
forward
projection
2d
h/2
w/2
H(x)

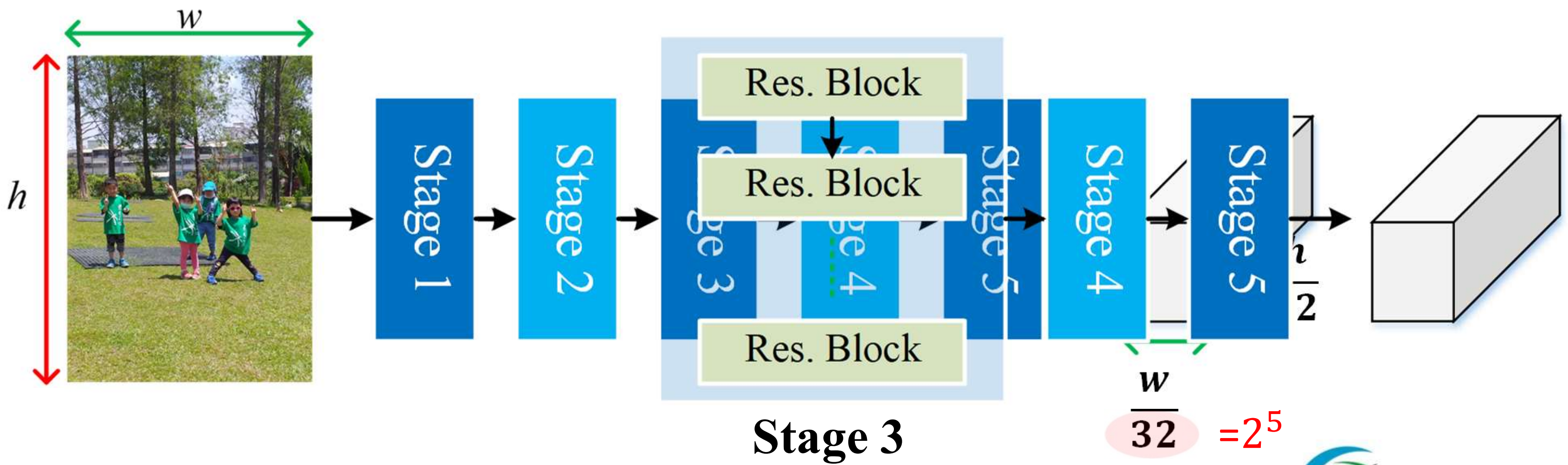
Deep Residual Network

- Residual Block: shortcut connection
  - perform size mapping by convolution operation.
    - use 1×1 kernel with a stride of 2
    - use $2d$ kernels to match depth



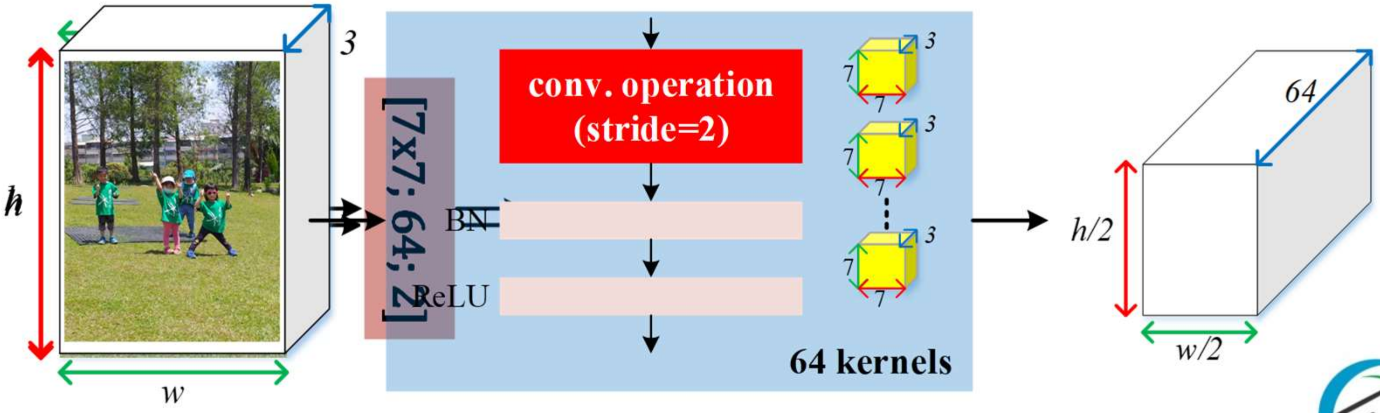
Deep Residual Network

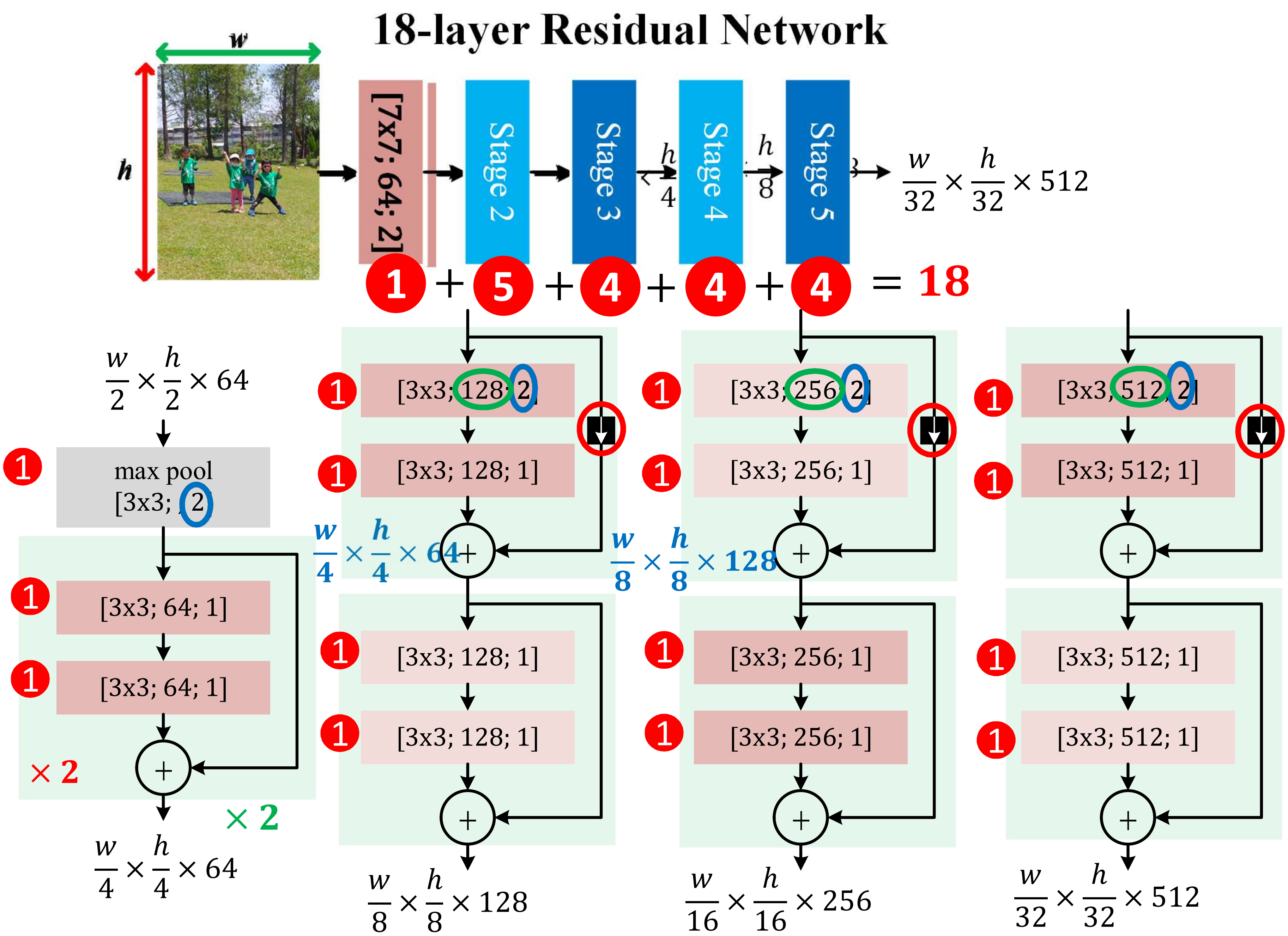
- Network Architecture
  - The proposed residual networks consist of 5 stages
    - each stage downsamples feature map by a factor of 2
    - each stage is composed of several residual blocks



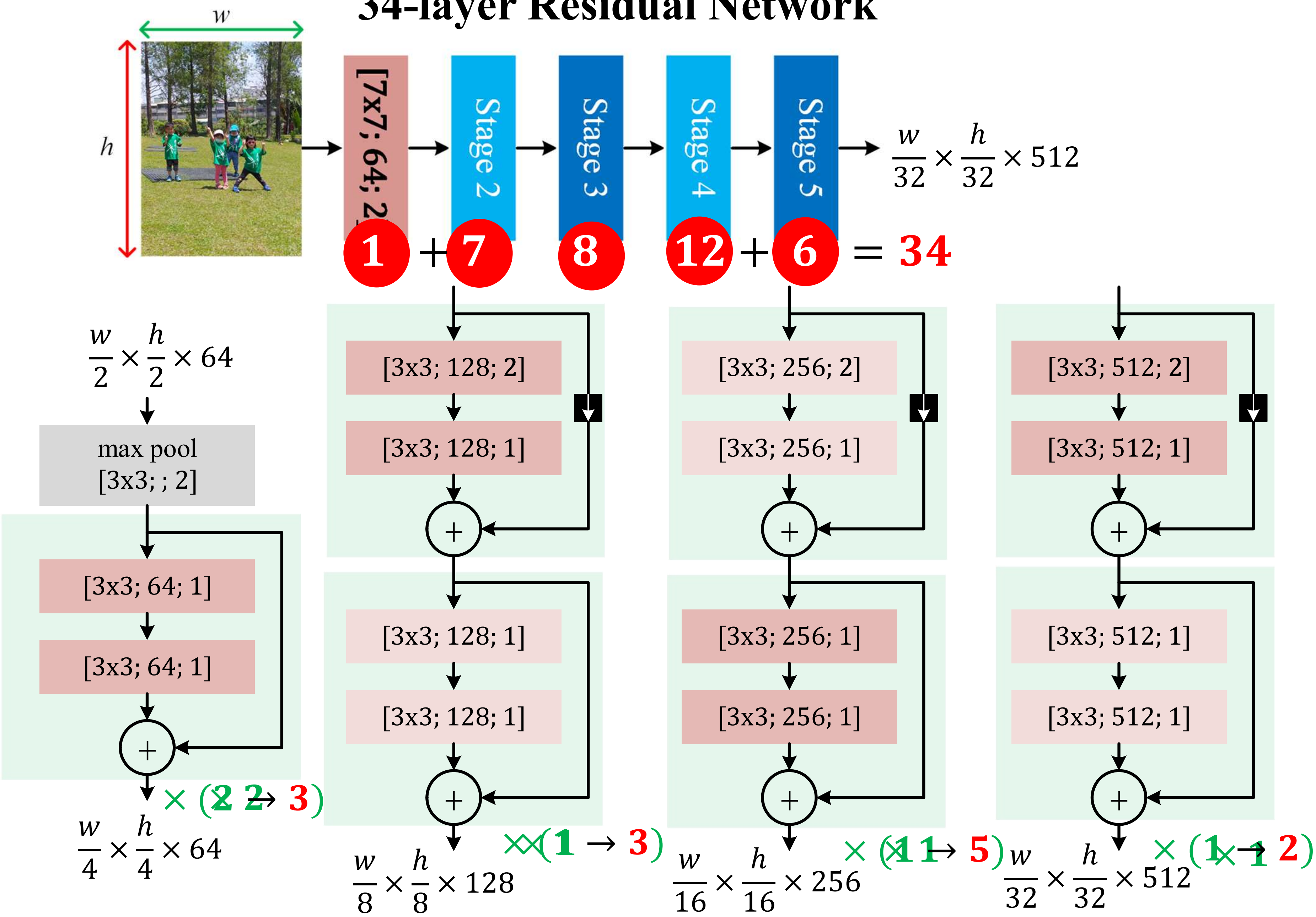
Deep Residual Network

- Network Architecture
  - residual networks: 18-layer and 34-layer
  - first stage: convolution layer with a stride of 2
    - kernel size: 7 × 7
    - kernel no.: 64

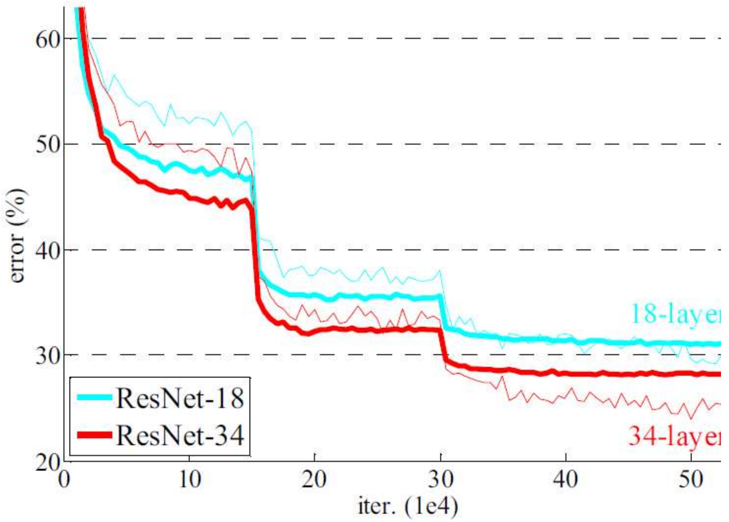


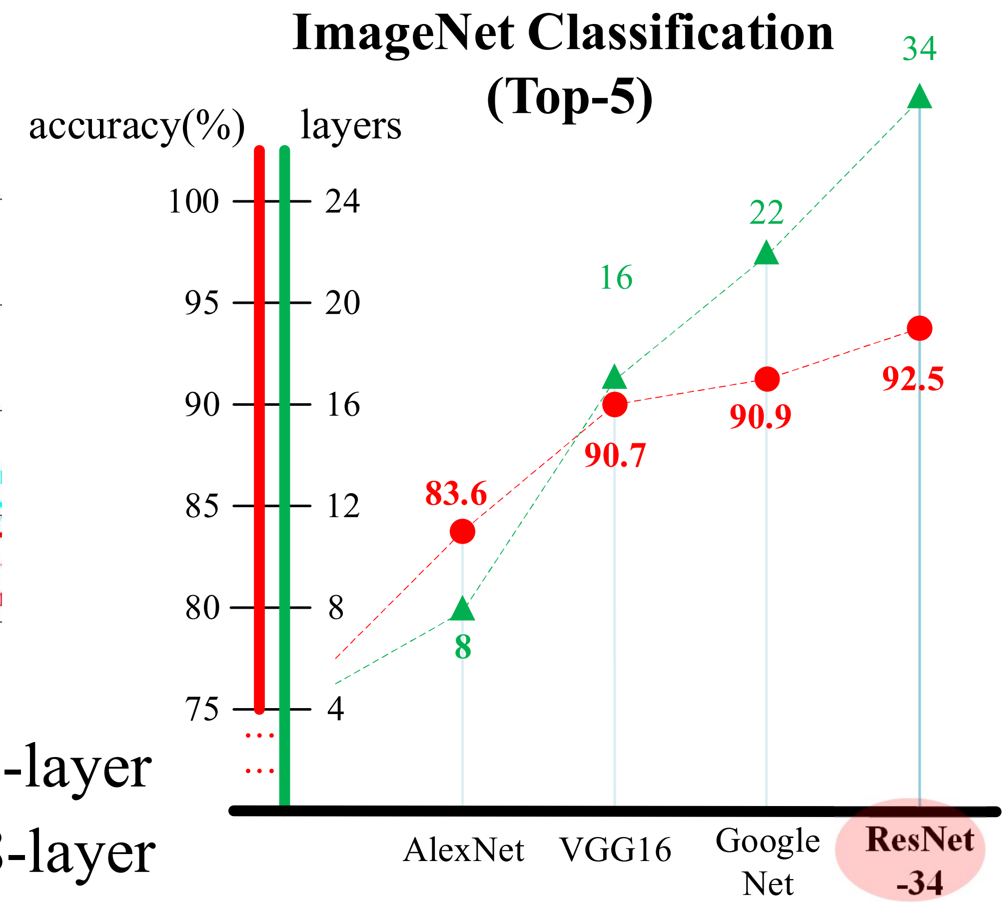

18-layer Residual Network
w
h
[7x7; 64; 2]
Stage 2
Stage 3
Stage 4
Stage 5
$\frac{h}{4}$
$\frac{h}{8}$
$\frac{w}{32} \times \frac{h}{32} \times 512$
1 + 5 + 4 + 4 + 4 = 18
$\frac{w}{2} \times \frac{h}{2} \times 64$
max pool
[3x3; 2]
[3x3; 64; 1]
[3x3; 64; 1]
× 2
× 2
$\frac{w}{4} \times \frac{h}{4} \times 64$
[3x3; 128; 2]
[3x3; 128; 1]
$\frac{w}{4} \times \frac{h}{4} \times 64$
[3x3; 128; 1]
[3x3; 128; 1]
$\frac{w}{8} \times \frac{h}{8} \times 128$
[3x3; 256; 2]
[3x3; 256; 1]
$\frac{w}{8} \times \frac{h}{8} \times 128$
[3x3; 256; 1]
[3x3; 256; 1]
$\frac{w}{16} \times \frac{h}{16} \times 256$
[3x3; 512; 2]
[3x3; 512; 1]
[3x3; 512; 1]
[3x3; 512; 1]
$\frac{w}{32} \times \frac{h}{32} \times 512$

34-layer Residual Network



Deep Residual Network

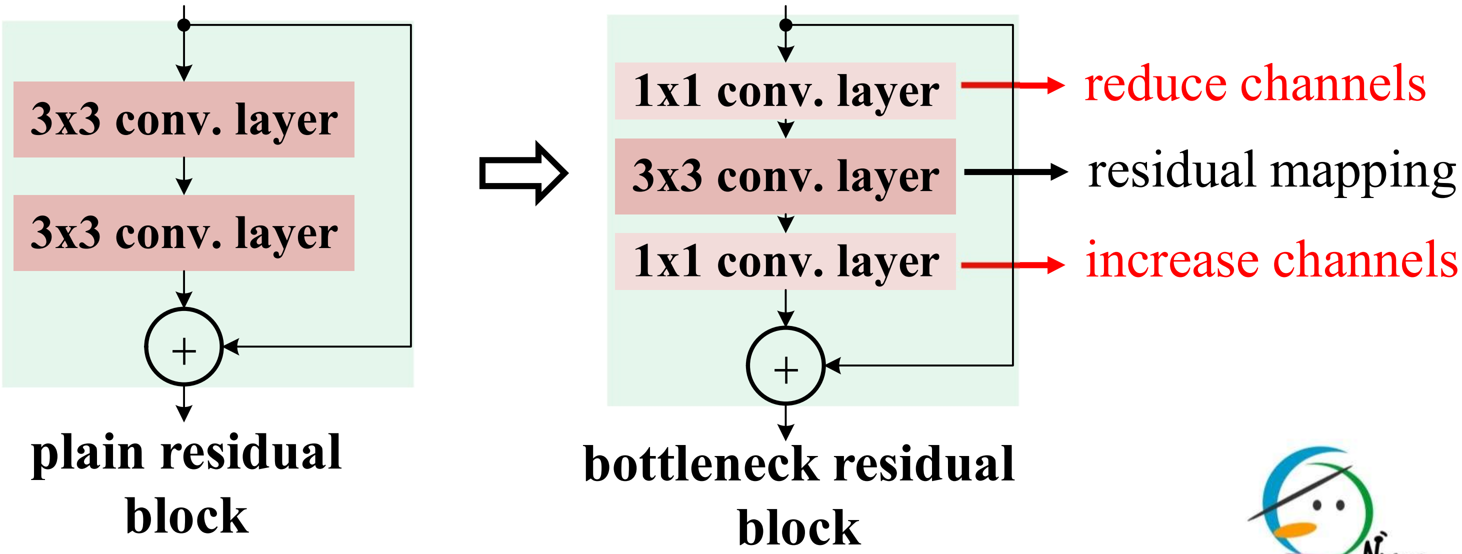




- train error: 34-layer < 18-layer
- test error: 34-layer < 18-layer

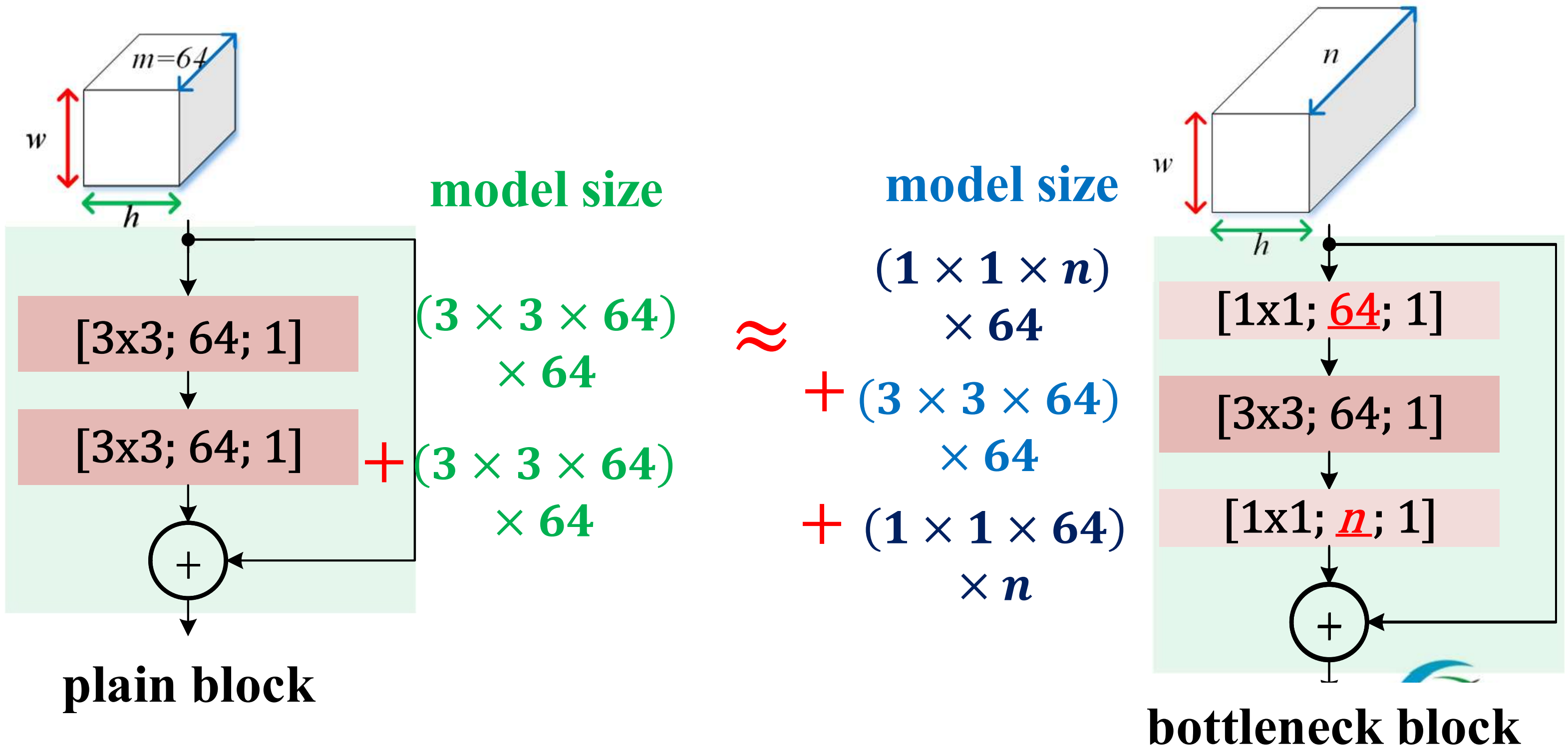
Deeper Residual Network

- Bottleneck Residual Block
  - consist of 3 convolution layers
    - 3×3 convolution layer: perform residual mapping
    - 1×1 convolution layer: reduce / increase channels



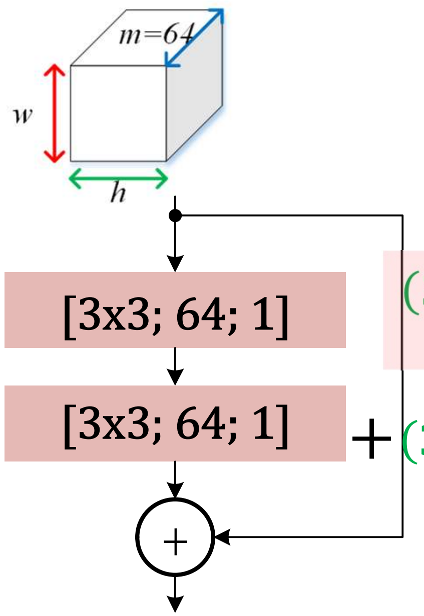
Deeper Residual Network

- Bottleneck Residual Block
  - have similar model size as its plain counterpart



Deeper Residual Network

- Bottleneck Residual Block
  - have similar model size as its plain counterpart

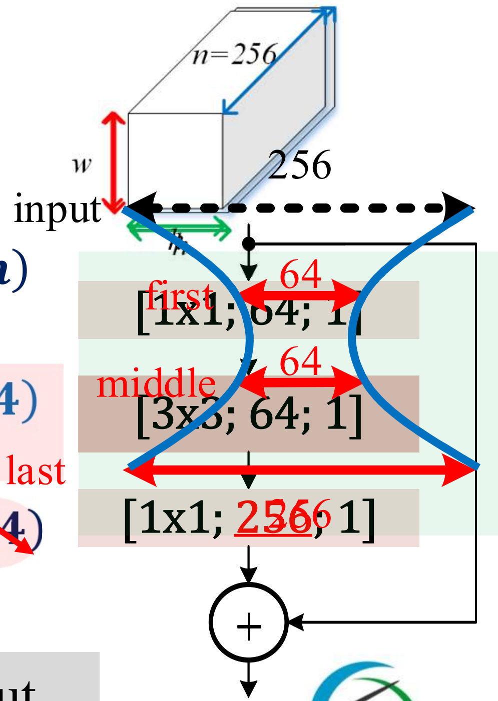


$\Rightarrow 3\times 3\times 64 \approx 2\times \boldsymbol{n}$

$\Rightarrow \boldsymbol{n} \approx 64\times (9/2)$

$\Rightarrow \boldsymbol{n} = 64\times 4 = 256$

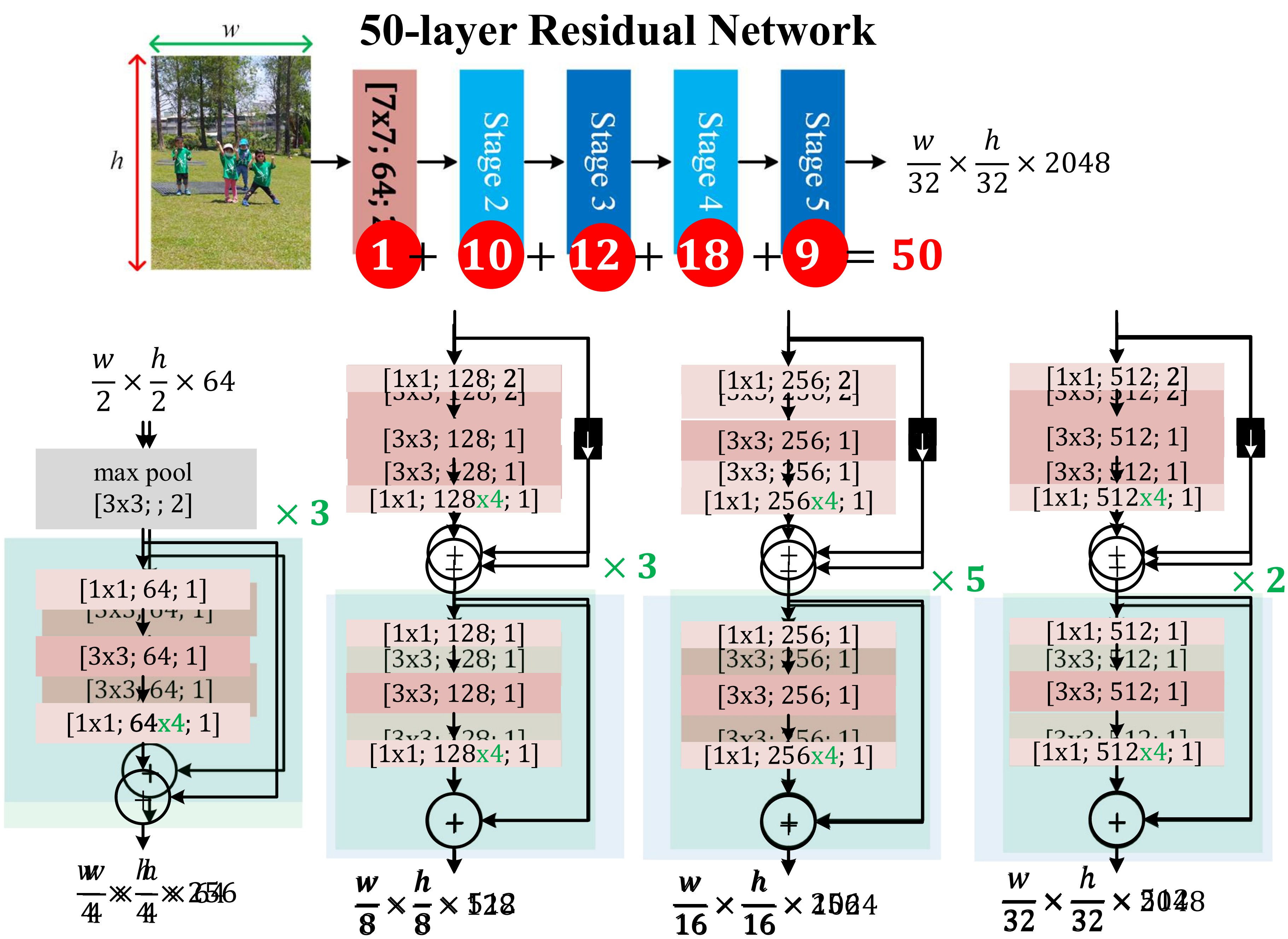
$(3\times 3\times 64)\times 64 + (3\times 3\times 64)\times 64 \approx (1\times 1\times \boldsymbol{n})\times 64 + (3\times 3\times 64)\times 64 + (1\times 1\times 64)\times \boldsymbol{n}$



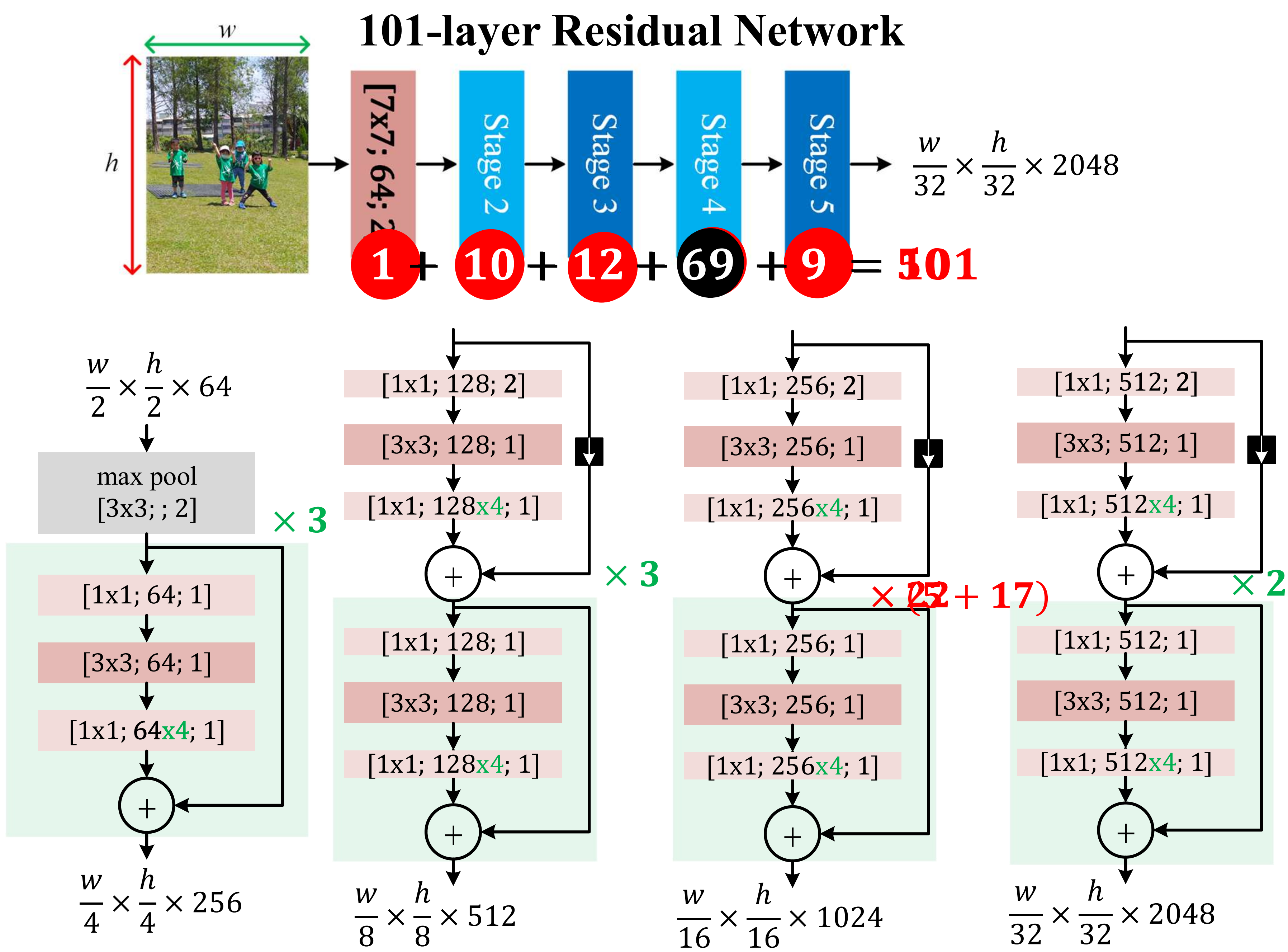
quadruple the channel of input

Deeper Residual Network

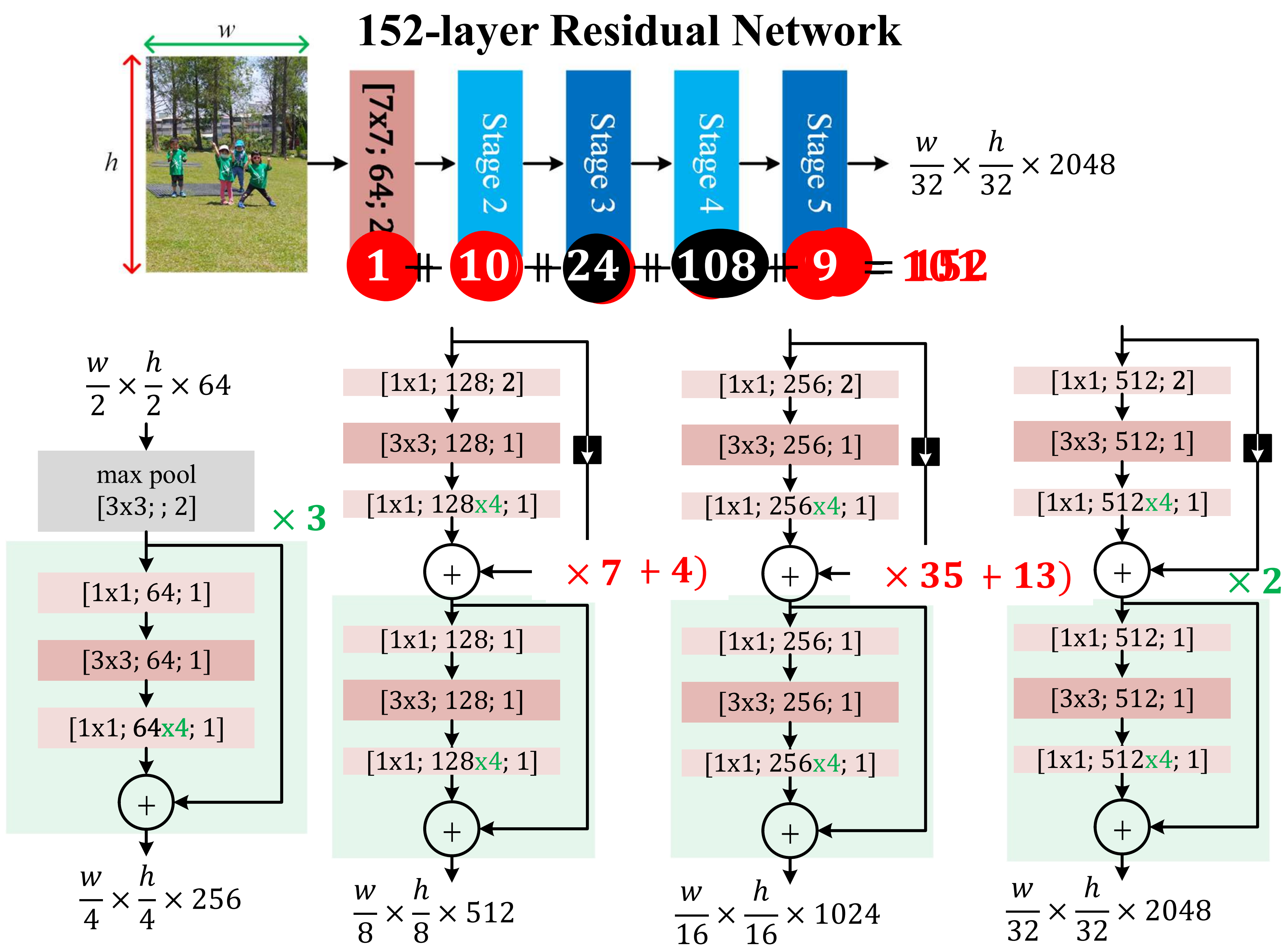
- Network Architecture
  - 50-layer: replace all plain residual blocks in 34-layer network by bottleneck ones.
  - 101-layer: increase bottleneck residual blocks to 50-layer network (stage 4: **17**)
  - 152-layer: increase bottleneck residual blocks to 101-layer network (stage 3: **4**; stage 4: **13**)



50-layer Residual Network
w
h
Stage 2
Stage 3
Stage 4
Stage 5
w/32 × h/32 × 2048
1 + 10 + 12 + 18 + 9 = 50
w/2 × h/2 × 64
max pool
[3x3; ; 2]
× 3
[1x1; 64; 1]
[3x3; 64; 1]
[1x1; 64x4; 1]
[1x1; 128; 2]
[3x3; 128; 1]
[1x1; 128x4; 1]
× 3
[1x1; 128; 1]
[3x3; 128; 1]
[1x1; 128x4; 1]
w/8 × h/8 × 512
[1x1; 256; 2]
[3x3; 256; 1]
[1x1; 256x4; 1]
× 5
[1x1; 256; 1]
[3x3; 256; 1]
[1x1; 256x4; 1]
w/16 × h/16 × 1024
[1x1; 512; 2]
[3x3; 512; 1]
[1x1; 512x4; 1]
× 2
[1x1; 512; 1]
[3x3; 512; 1]
[1x1; 512x4; 1]
w/32 × h/32 × 2048

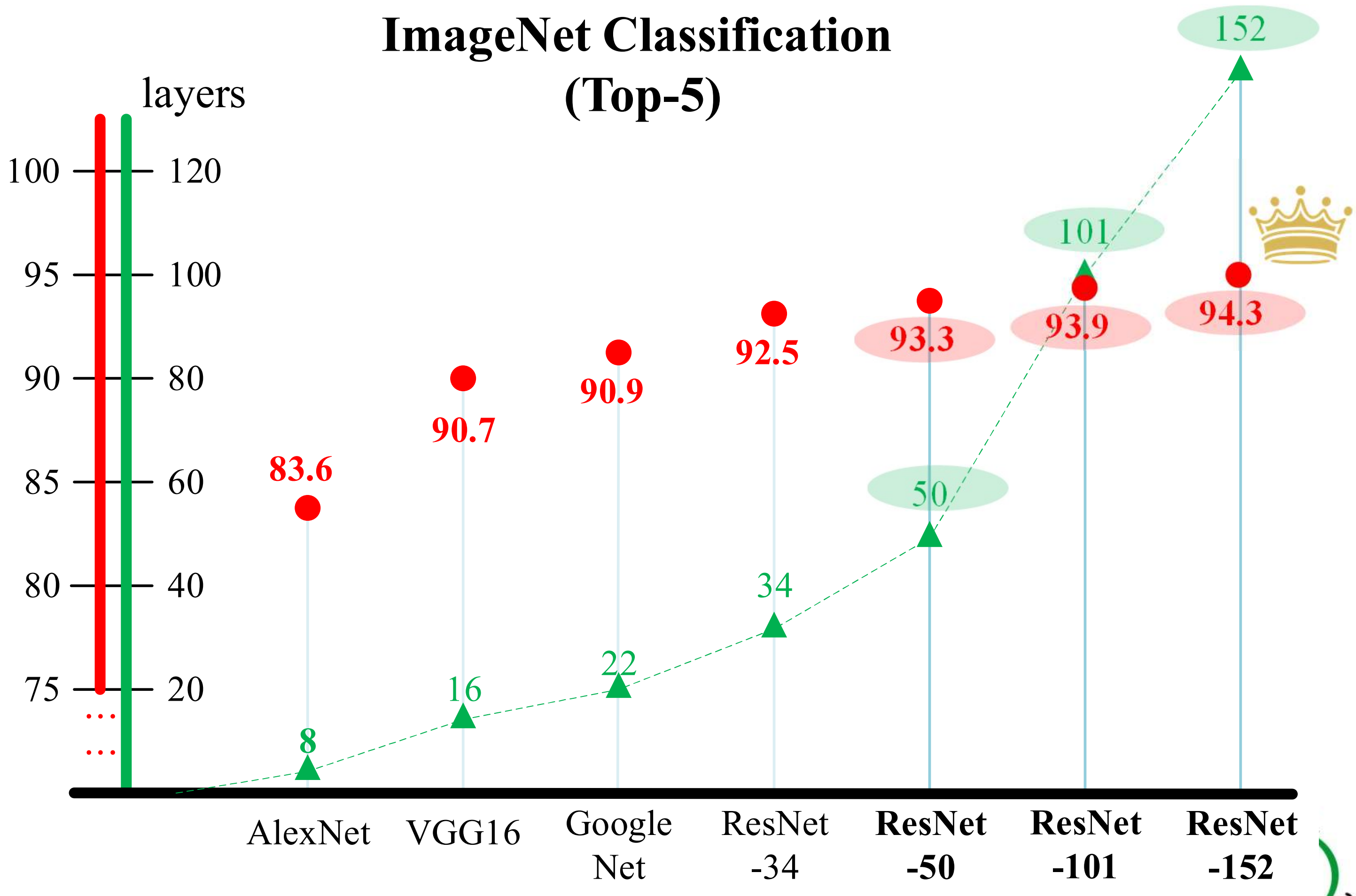


101-layer Residual Network
w
h
[7x7; 64; 2
Stage 2
Stage 3
Stage 4
Stage 5
w/32 × h/32 × 2048
1 + 10 + 12 + 69 + 9 = 101
w/2 × h/2 × 64
max pool
[3x3; ; 2]
× 3
[1x1; 64; 1]
[3x3; 64; 1]
[1x1; 64x4; 1]
w/4 × h/4 × 256
[1x1; 128; 2]
[3x3; 128; 1]
[1x1; 128x4; 1]
× 3
[1x1; 128; 1]
[3x3; 128; 1]
[1x1; 128x4; 1]
w/8 × h/8 × 512
[1x1; 256; 2]
[3x3; 256; 1]
[1x1; 256x4; 1]
× (22 + 17)
[1x1; 256; 1]
[3x3; 256; 1]
[1x1; 256x4; 1]
w/16 × h/16 × 1024
[1x1; 512; 2]
[3x3; 512; 1]
[1x1; 512x4; 1]
× 2
[1x1; 512; 1]
[3x3; 512; 1]
[1x1; 512x4; 1]
w/32 × h/32 × 2048



152-layer Residual Network
w
h
Stage 2
Stage 3
Stage 4
Stage 5
1 + 10 + 24 + 108 + 9
w/32 × h/32 × 2048
w/2 × h/2 × 64
max pool
[3x3; ; 2]
× 3
[1x1; 64; 1]
[3x3; 64; 1]
[1x1; 64x4; 1]
w/4 × h/4 × 256
[1x1; 128; 2]
[3x3; 128; 1]
[1x1; 128x4; 1]
× 7 + 4)
[1x1; 128; 1]
[3x3; 128; 1]
[1x1; 128x4; 1]
w/8 × h/8 × 512
[1x1; 256; 2]
[3x3; 256; 1]
[1x1; 256x4; 1]
× 35 + 13)
[1x1; 256; 1]
[3x3; 256; 1]
[1x1; 256x4; 1]
w/16 × h/16 × 1024
[1x1; 512; 2]
[3x3; 512; 1]
[1x1; 512x4; 1]
× 2
[1x1; 512; 1]
[3x3; 512; 1]
[1x1; 512x4; 1]
w/32 × h/32 × 2048

Deeper Residual Network




thank you
© 1996, 2002 SANRIO CO., LTD.