Deep Residual Learning for Image Recognition

arXiv 2015, 66424 citation

Identity Mappings in Deep Residual Networks

arXiv 2016, 4839 citation

GIST EECS
20205035 김연혁

Main Problem – Deeper network

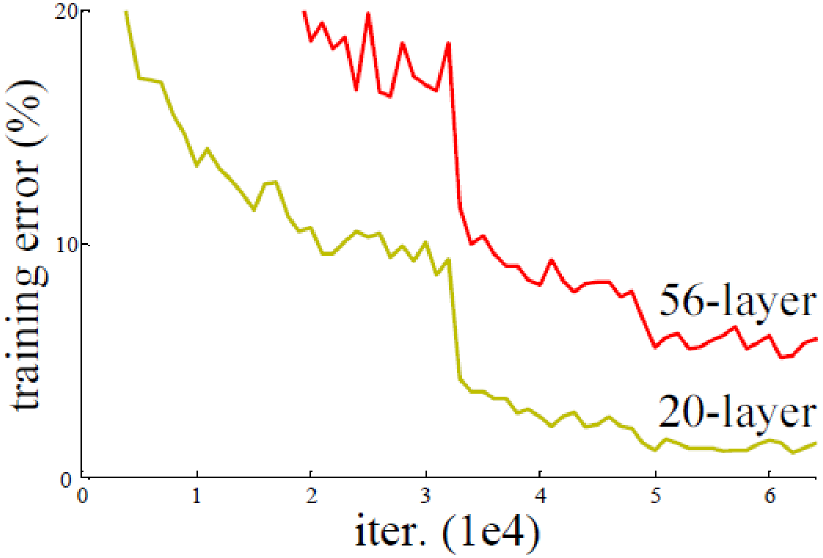
Advantage

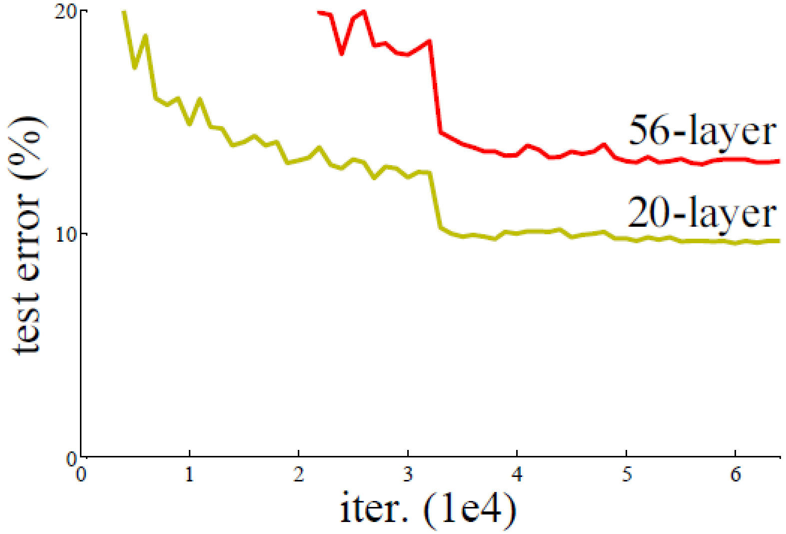
Deeper network
-> Better extracting representative concepts in the learning data
-> Better Result

Disadvantage

1. Deeper network
   -> Degradation problem
   -> Increase error

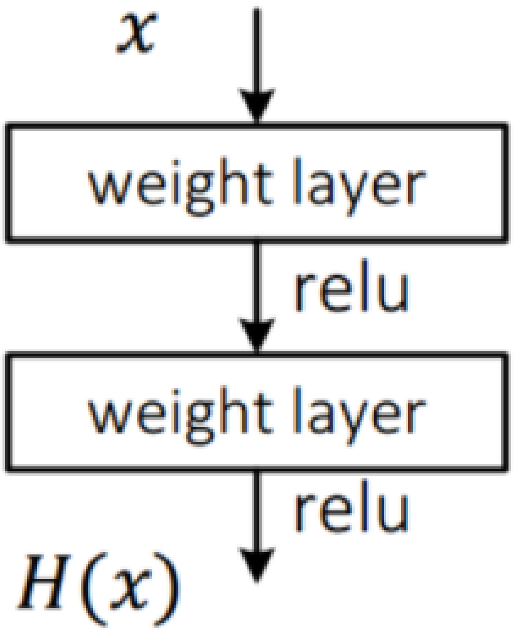
2. Deeper network
   -> Big number of parameters
   -> Lots of computation, Increase error



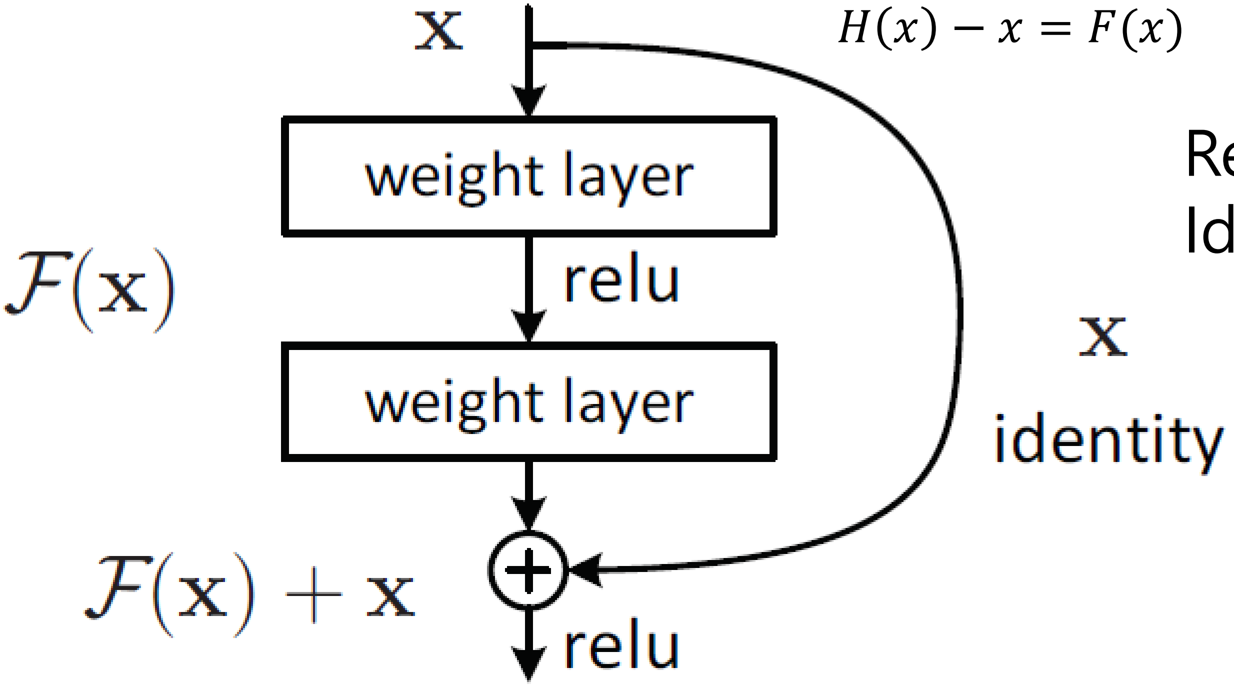


Residual Learning

For making deeper network (more than 100 layers), getting effect of deep layer



Find $H(x)$

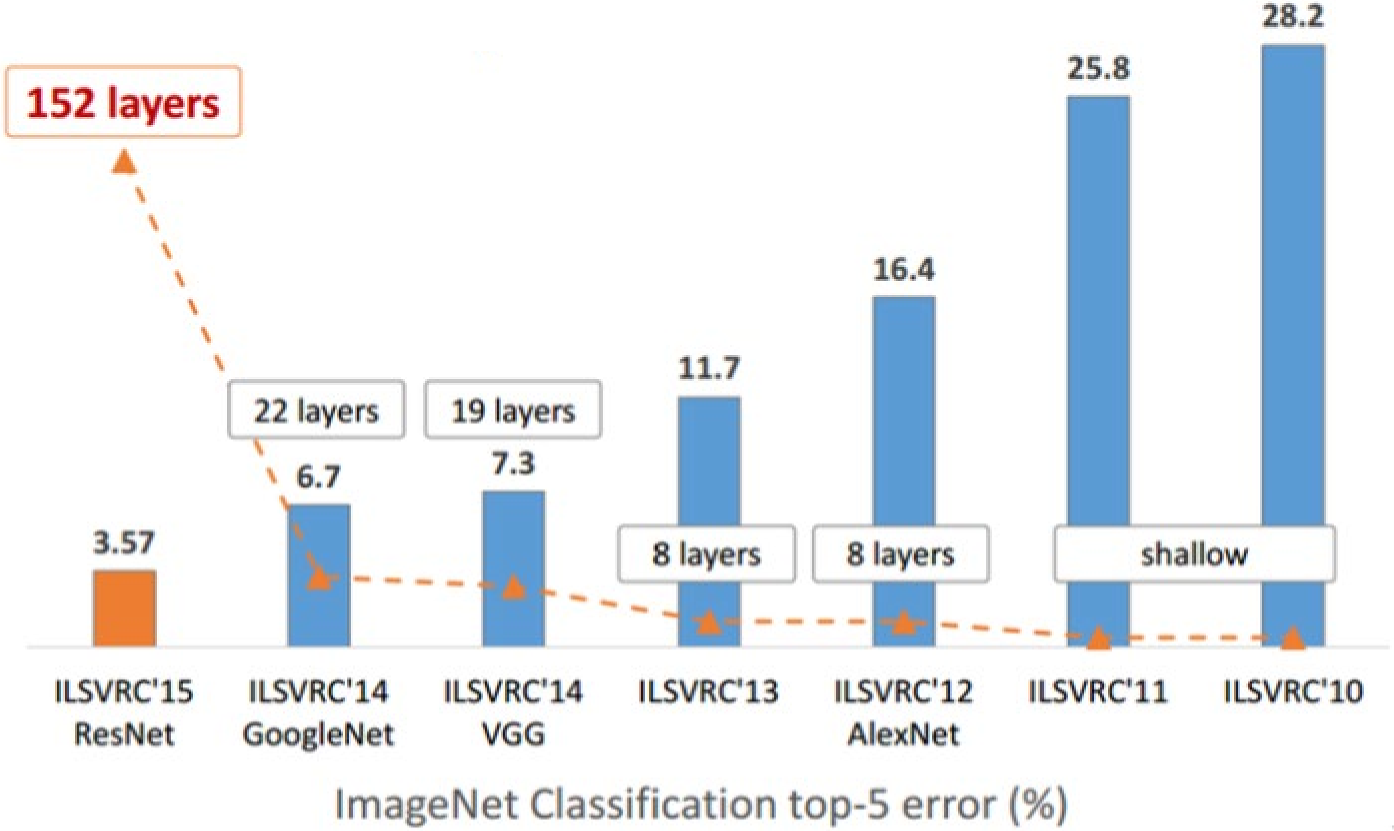


Find $F(x)$

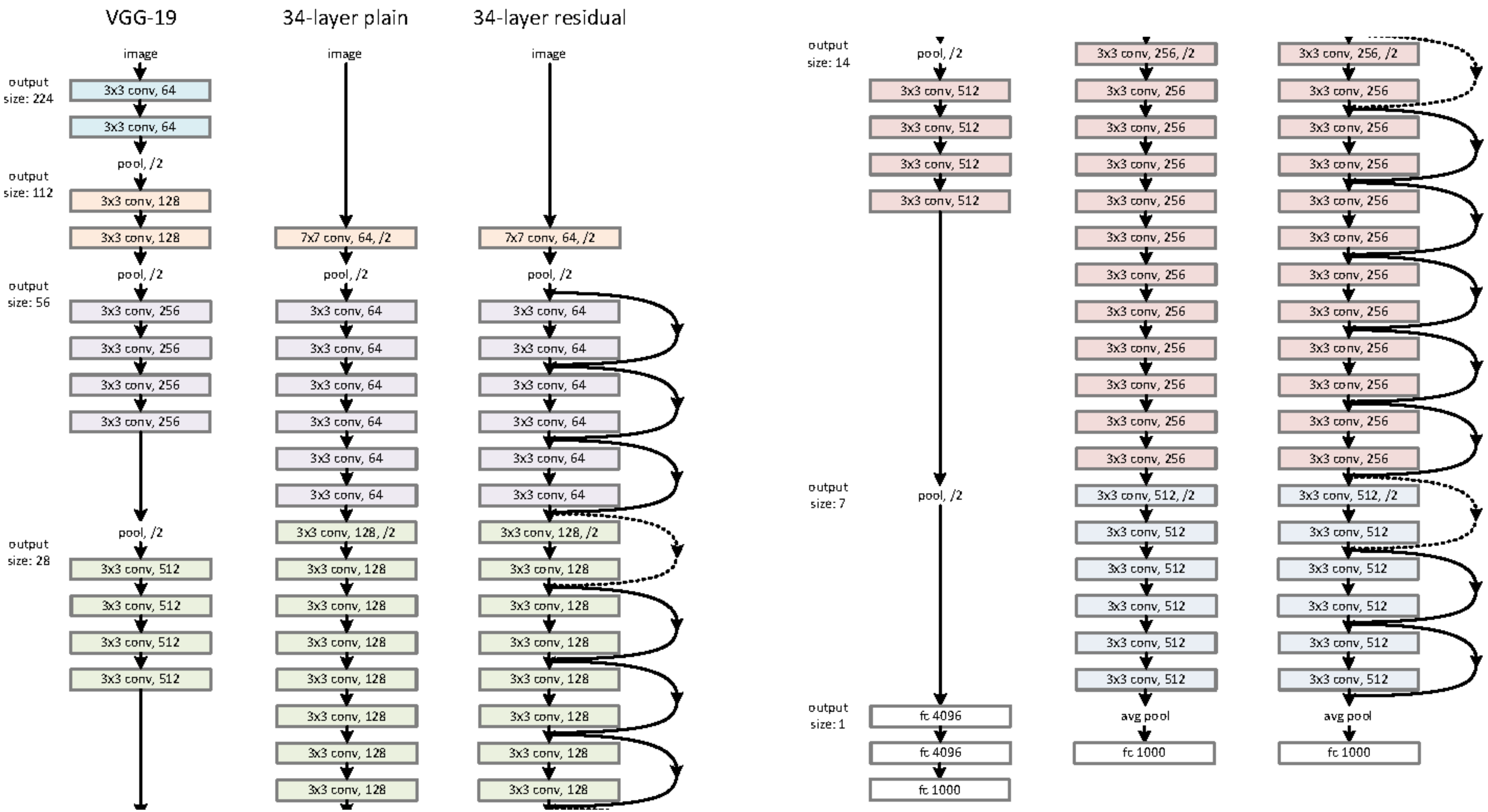
Residual Units,
Identity mapping

Advantages of Residual Learning

1. No changes of number of parameters
2. Easier optimization for deep network
3. Increase accuracy by deeper network



Experiment for ResNet



Experiment for ResNet with ImageNet dataset

3x3 kernel convolutional layer – similar with VGGNet

No max-pooling (except last layer)
No hidden FC layer
No dropout

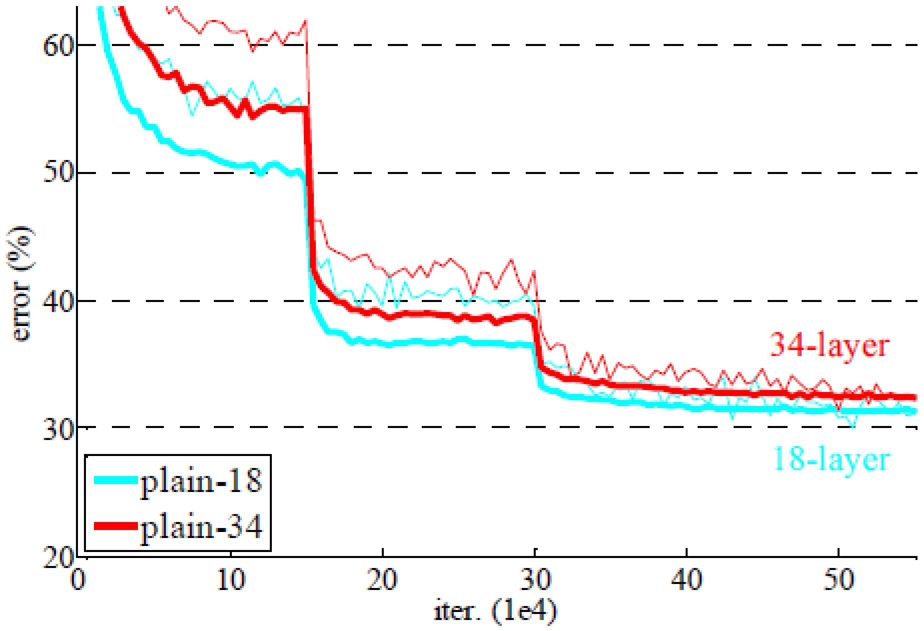
Lower complexity, Less computation

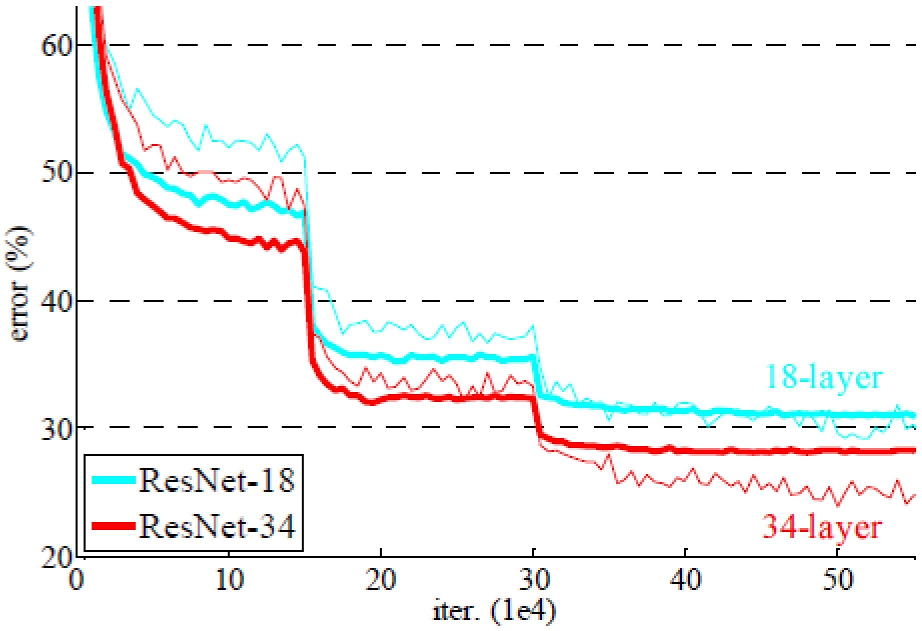
layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

VGGNet 19-layer: 19.6 billion FLOPs, ResNet 34-layer Plain: 3.6 billion FLOPs

FLOPs: Floating Point Operation Per Second

Experiment for ResNet – layer 18 and 34





	plain	ResNet
18 layers	27.94	27.88
34 layers	28.54	**25.03**

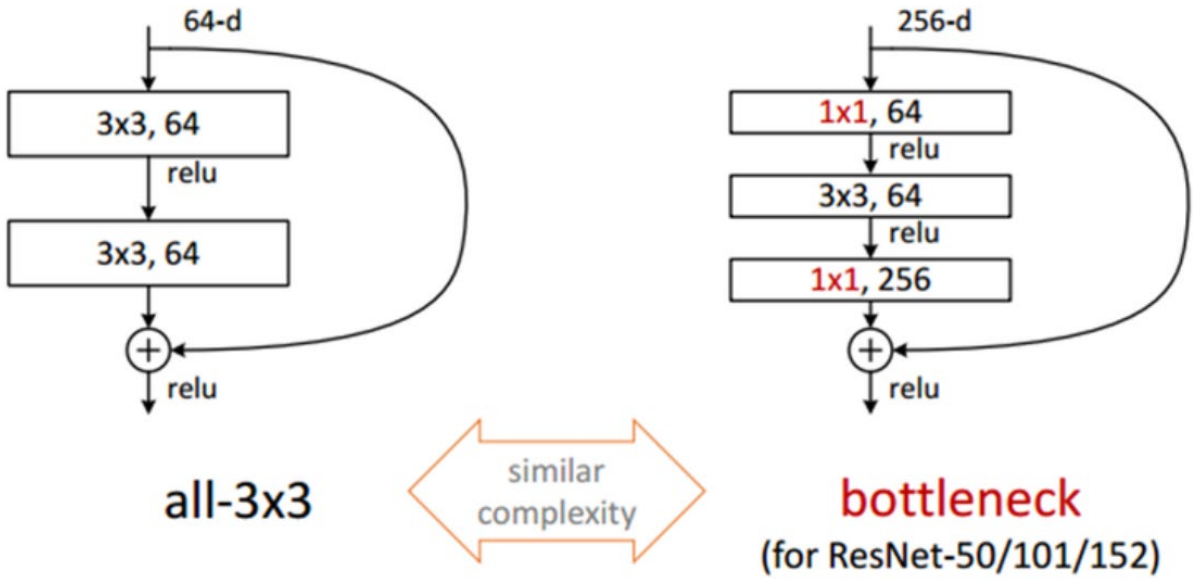
Better Accuracy, Faster learning!

Experiment for ResNet

model	top-1 err.	top-5 err.
VGG-16 [41]	28.07	9.33
GoogLeNet [44]	-	9.15
PReLU-net [13]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

method	top-5 err. (**test**)
VGG [41] (ILSVRC'14)	7.32
GoogLeNet [44] (ILSVRC'14)	6.66
VGG [41] (v5)	6.8
PReLU-net [13]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

Deeper Bottleneck Architectures



layer name	output size	18-layer	34-layer	50-layer
conv1	112×112			7×7, 64, stride 2
				3×3 max pool, strid
conv2_x	56×56	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1			average pool, 1000-d fc,
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9

First 1x1 convolution: Reduce Dimension
Last 1x1 convolution: Expand Dimension

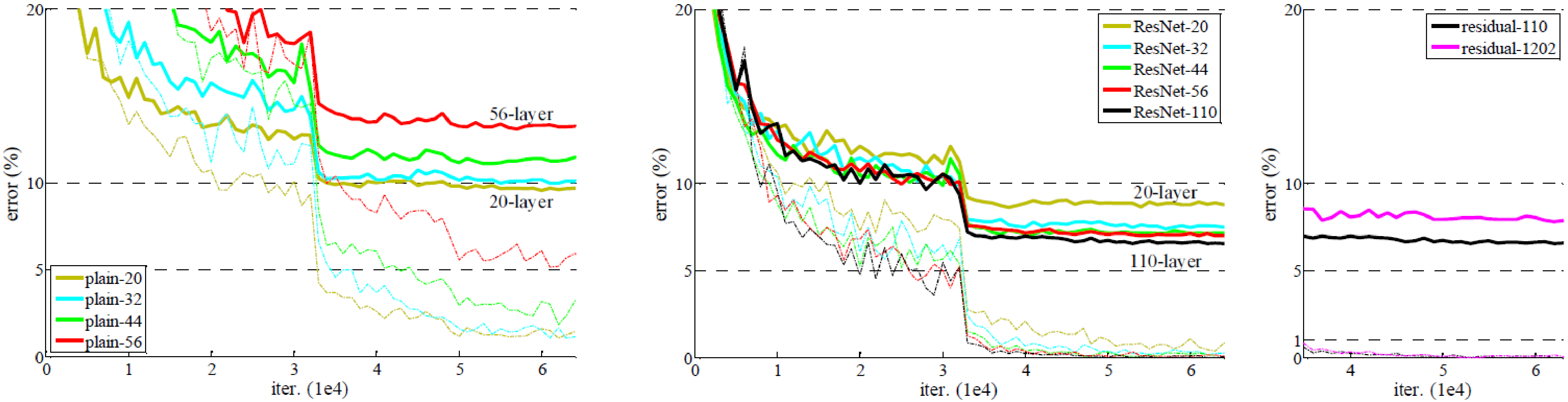
Reduce Computation!

Experiment with CIFAR-10 dataset

32x32 pixel size (smaller than ImageNet dataset: 224x224)
10 classes, total 60k images

output map size	32×32	16×16	8×8
# layers	1+2n	2n	2n
# filters	16	32	64

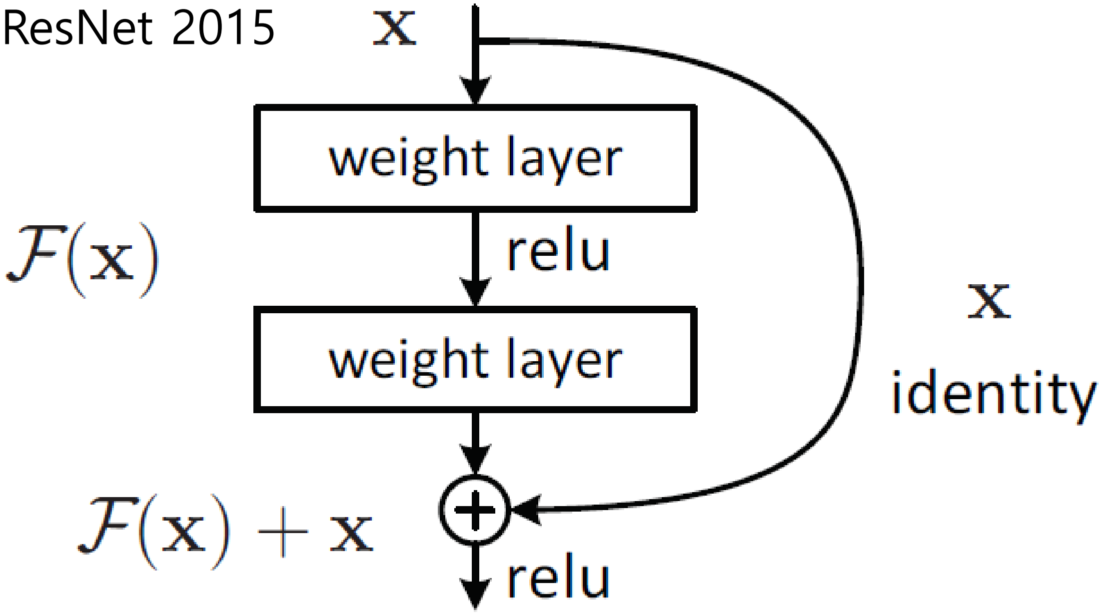
method			error (%)
Maxout [10]			9.38
NIN [25]			8.81
DSN [24]			8.22
	# layers	# params	
FitNet [35]	19	2.5M	8.39
Highway [42, 43]	19	2.3M	7.54 (7.72±0.16)
Highway [42, 43]	32	1.25M	8.80
ResNet	20	0.27M	8.75
ResNet	32	0.46M	7.51
ResNet	44	0.66M	7.17
ResNet	56	0.85M	6.97
ResNet	110	1.7M	**6.43** (6.61±0.16)
ResNet	1202	19.4M	7.93



But
Why Residual Learning?
Why Identity Mapping?

Shortcut with identity mapping

ResNet 2015



$$y_l = h(x_l) + F(x_l, W_l),$$
$$x_{l+1} = f(y_l)$$

Original Residual Unit

$$y_l = x_l + F(x_l, W_l),$$
$$x_{l+1} = y_l,$$
$$x_{l+1} = x_l + F(x_l, W_l),$$
$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i)$$

h is identity

ReLU

$$\frac{\partial \varepsilon}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1} F(x_i, W_i)\right)$$

Shortcut with identity mapping

$\frac{\partial \varepsilon}{\partial x_L}$: Directly Propagate

$\frac{\partial \varepsilon}{\partial x_L} \frac{\partial}{\partial x_l} \sum_{i=l}^{L-1} F(x_l, W_l)$: Propagate through weights

$$y_l = x_l + F(x_l, W_l),$$
$$x_{l+1} = y_l,$$
$$x_{l+1} = x_l + F(x_l, W_l),$$
$$x_L = x_l + \sum_{i=l}^{L-1} F(x_l, W_l)$$
$$\frac{\partial \varepsilon}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L} \frac{\partial x_L}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L} \left(1 + \frac{\partial}{\partial x_l} \sum_{i=l}^{L-1} F(x_l, W_l) \right)$$

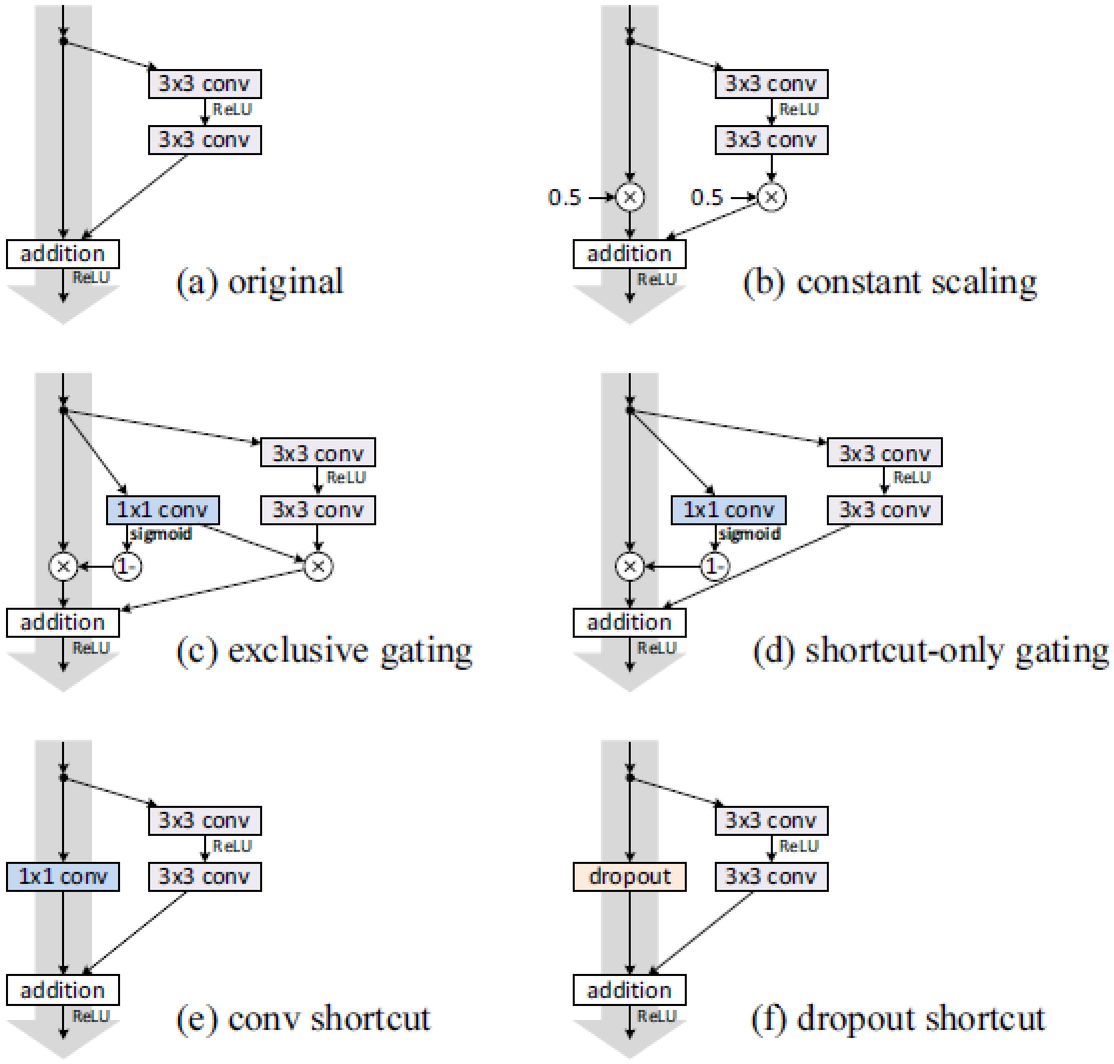
Shortcut with identity mapping

h is identity	$h(x_l) = \lambda_l x_l$
$$y_l = x_l + F(x_l, W_l),$$ $$x_{l+1} = y_l,$$ $$x_{l+1} = x_l + F(x_l, W_l),$$ $$x_L = x_l + \sum_{i=l}^{L-1} F(x_l, W_l)$$ $$\frac{\partial \varepsilon}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1} F(x_l, W_l)\right)$$	$$y_l = \lambda_l x_l + F(x_l, W_l),$$ $$x_{l+1} = y_l,$$ $$x_{l+1} = x_l + F(x_l, W_l),$$ $$x_L = \left(\prod_{i=l}^{L-1} \lambda_l\right) x_l + \sum_{i=l}^{L-1} \hat{F}(x_l, W_l)$$ $$\frac{\partial \varepsilon}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\left(\left(\prod_{i=l}^{L-1} \lambda_l\right) + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1} \hat{F}(x_l, W_l)\right)$$

If $\lambda_l < 1$, exponentially small and vanish

If $\lambda_l > 1$, exponentially large

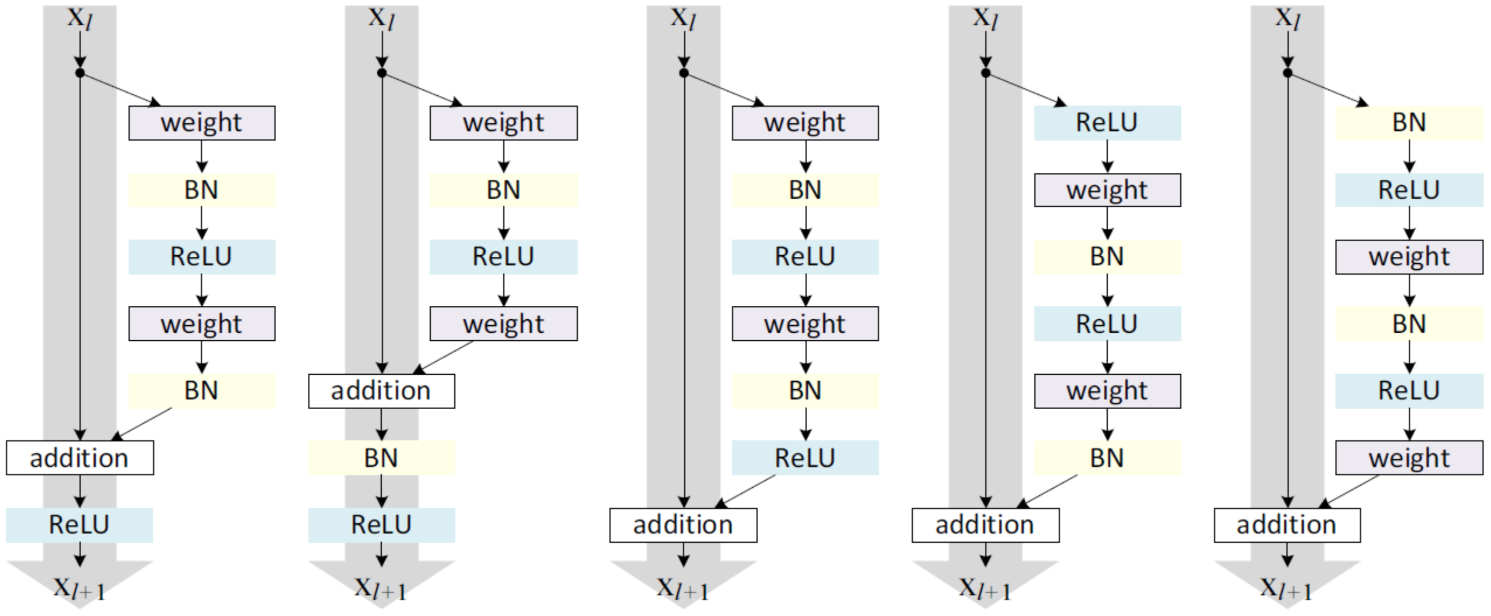
Various types of shortcut connections



(a) original
(b) constant scaling
(c) exclusive gating
(d) shortcut-only gating
(e) conv shortcut
(f) dropout shortcut

case	Fig.	on shortcut	on $\mathcal{F}$	error (%)
original [1]	Fig. 2(a)	1	1	**6.61**
constant scaling	Fig. 2(b)	0	1	fail
		0.5	1	fail
		0.5	0.5	12.35
exclusive gating	Fig. 2(c)	$1-g(\mathbf{x})$	$g(\mathbf{x})$	fail
		$1-g(\mathbf{x})$	$g(\mathbf{x})$	8.70
		$1-g(\mathbf{x})$	$g(\mathbf{x})$	9.81
shortcut-only gating	Fig. 2(d)	$1-g(\mathbf{x})$	1	12.86
		$1-g(\mathbf{x})$	1	6.91
1×1 conv shortcut	Fig. 2(e)	1×1 conv	1	12.22
dropout shortcut	Fig. 2(f)	dropout 0.5	1	fail

Various usages of activation



(a) original (b) BN after addition (c) ReLU before addition (d) ReLU-only pre-activation (e) **full pre-activation**

case	Fig.	ResNet-110	ResNet-164
original Residual Unit [1]	Fig. 4(a)	6.61	5.93
BN after addition	Fig. 4(b)	8.17	6.50
ReLU before addition	Fig. 4(c)	7.84	6.14
ReLU-only pre-activation	Fig. 4(d)	6.71	5.91
full pre-activation	Fig. 4(e)	**6.37**	**5.46**

Application to CIFAR-100 and more than 1k layers

dataset	network	baseline unit	pre-activation unit
CIFAR-10	ResNet-110 (1layer skip)	9.90	8.91
	ResNet-110	6.61	6.37
	ResNet-164	5.93	5.46
	ResNet-1001	7.61	4.92
CIFAR-100	ResNet-164	25.16	24.33
	ResNet-1001	27.82	22.71

Conclusion

Deeper Network can be optimized by Residual Network without degradation problem

We can reduce the number of parameters with Bottleneck architecture in deep network

Identity mapping is the best way for shortcut connection

We can solve vanishing gradient problem with identity mapping

We can increase the performance in shortcut connection with pre-activation

References

Deep Residual Learning for Image Recognition

https://blog.naver.com/laonple/220761052425
https://blog.naver.com/laonple/220764986252
https://blog.naver.com/laonple/220770760226
http://openresearch.ai/t/resnet-deep-residual-learning-for-image-recognition/41
https://bskyvision.com/644?category=635506
https://ganghee-lee.tistory.com/41
https://dnddnjs.github.io/cifar10/2018/10/09/resnet/#toward-deeper-network

Identity Mappings in Deep Residual Networks

http://openresearch.ai/t/identity-mappings-in-deep-residual-networks/47
https://kangbk0120.github.io/articles/2018-01/identity-mapping-in-deep-resnet
https://curaai00.tistory.com/6