ResNet

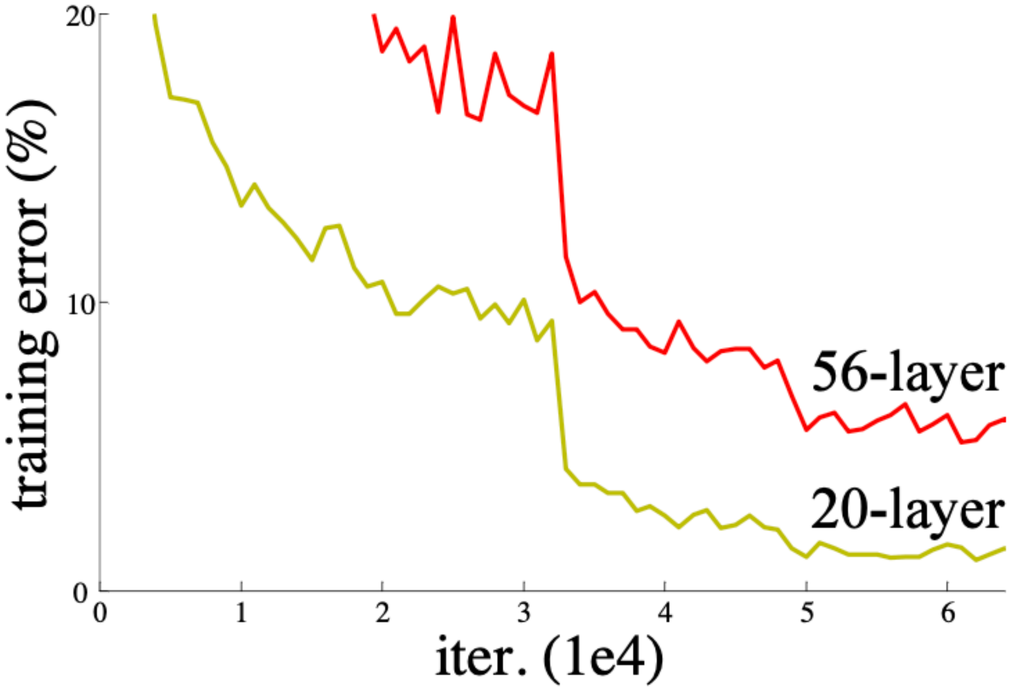
Deep Residual Learning for Image Recognition

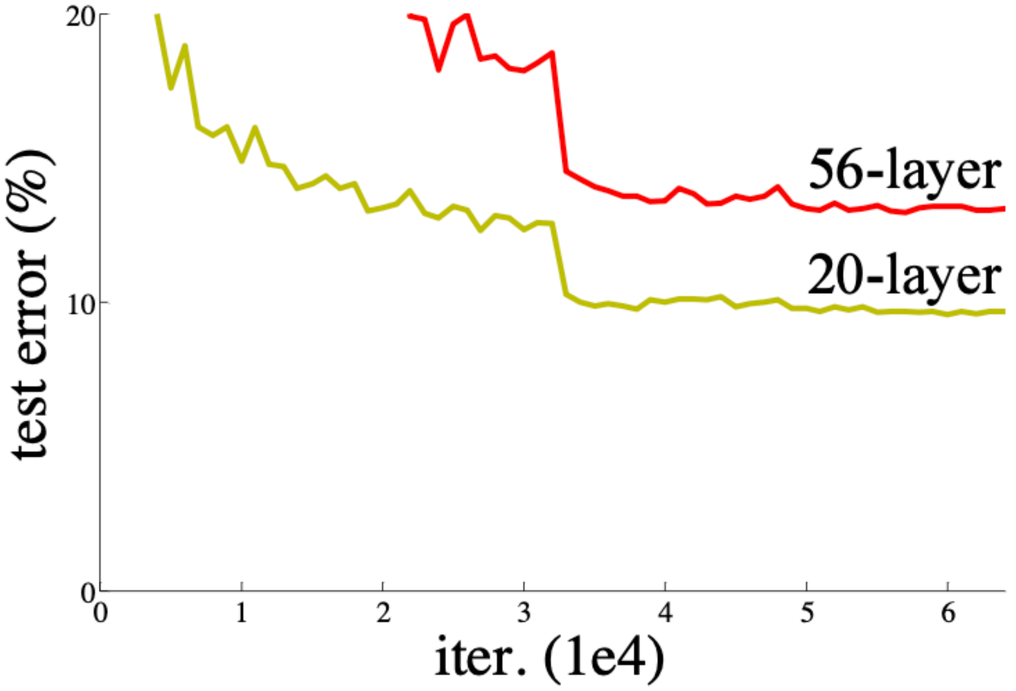
Younjung Kang

Introduction

“Is learning better networks as easy as stacking more layers?”

‘레이어를 더 많이 쌓을수록 성능이 향상될까?’





Depth가 어느 부분까지 상승하다가 그 이상은 Vanishing/exploding gradients 문제로 오히려 성능이 더 떨어지는 문제 -> Degradation problem

Introduction

‘Degradation problem’

: Depth가 깊은 상태에서 학습을 이미 많이 진행한 경우,
weight들의 분포가 균등하지 않고, 역전파시 기울기가 충분하지 않아 학습을 안정적으로 할 수 없는 문제가 발생.

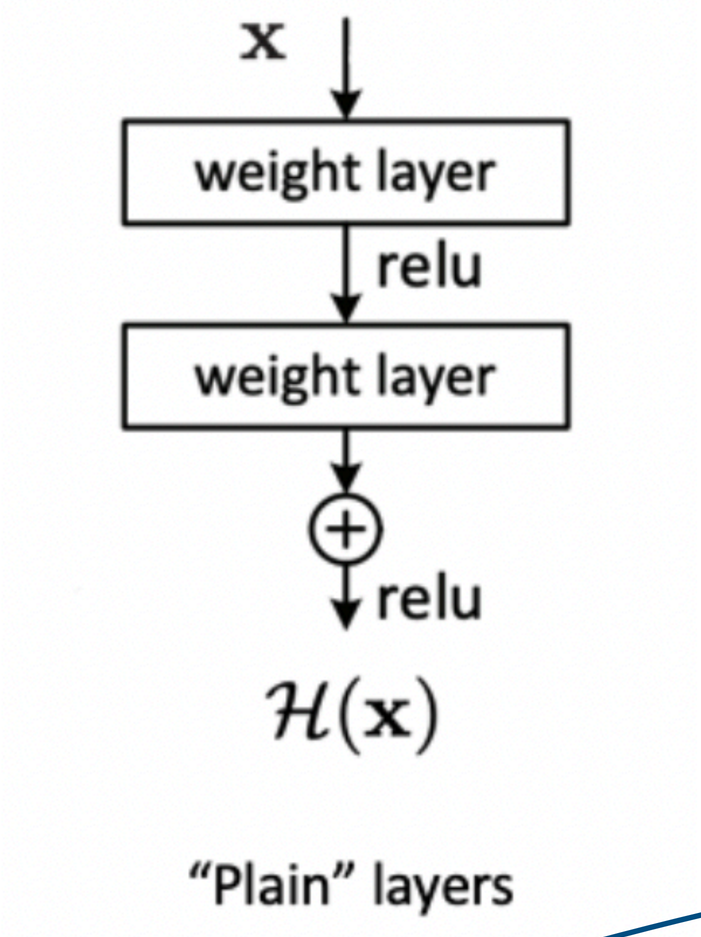
-> Overfittng 문제라고 착각할 수 있지만, 실제로 그것이 원인은 아니다!

Residual learning

+ 이를 feed-forward neural network에 적용한 것

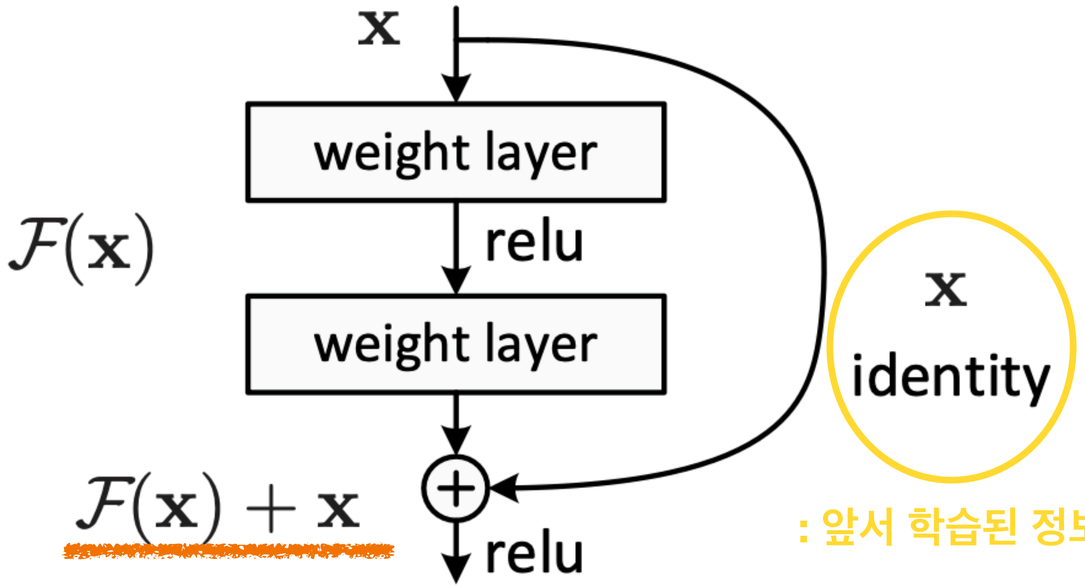
-> ‘Shortcut connection’

: skipping one or more layers.



“Plain” layers

일반 네트워크 : H(x)



: 앞서 학습된 정보

Figure 2. Residual learning: a building block.

$F(x) := H(x) - x$

학습하기 쉬운 매핑으로 전환 (최적화 용이)

: F(x) + x를 H(x)에 근사하도록 하는 것(Residual mapping)

-> 기존에 학습했던 정보 x는 그대로 가져오되,
추가적으로 필요한 정보 F(x)를 더해주는 것

Residual learning

Residual learning을 활용하는 이유는
네트워크의 최적화(optimization) 난이도를 낮추기 위함이다.

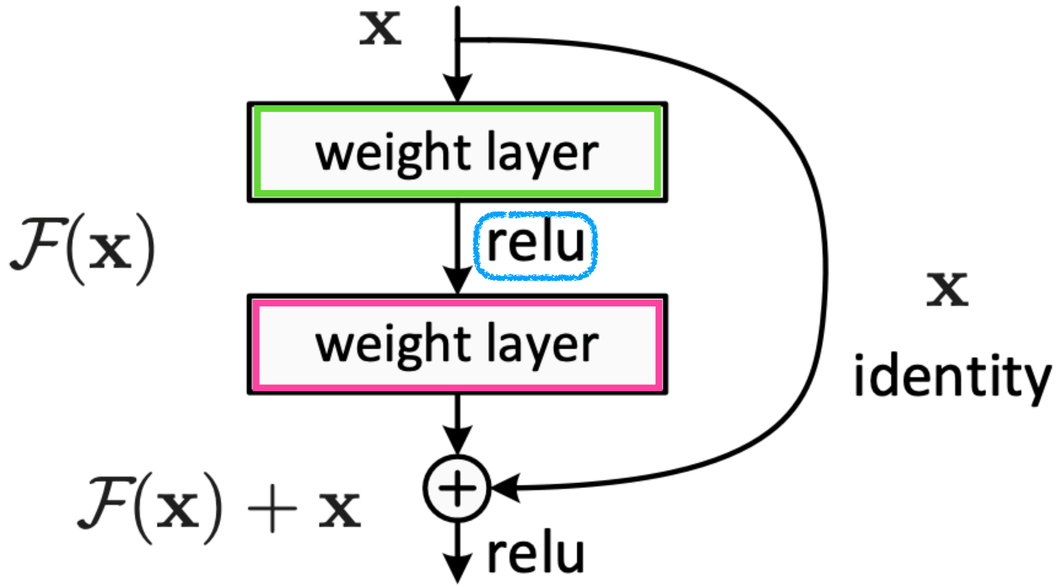


Figure 2. Residual learning: a building block.

$$\mathcal{F} = W_2\sigma(W_1\mathbf{x})$$

Shortcuts

*ResNet의 중요한 제약조건
: 입력값 x와 출력값 y의 dimension이 항상 같아야 한다.
그래야 연산을 수행할 수 있다!

입력값을 출력값으로 바로 보내주는 shortcut은 크게 두 가지 형태로 나뉜다.

1. Identity shortcut

: 입력값과 출력값의 dimension이 동일한 경우

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}.$$

2. Projection shortcut

: 입력값과 출력값의 dimension이 다른 경우
->W_S 를 통해 dimension을 맞춰준다.

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}.$$

Square matrix

Network Architectures

- **Plain Network**

  - VGG nets에서 영감을 받아 만들어졌다.

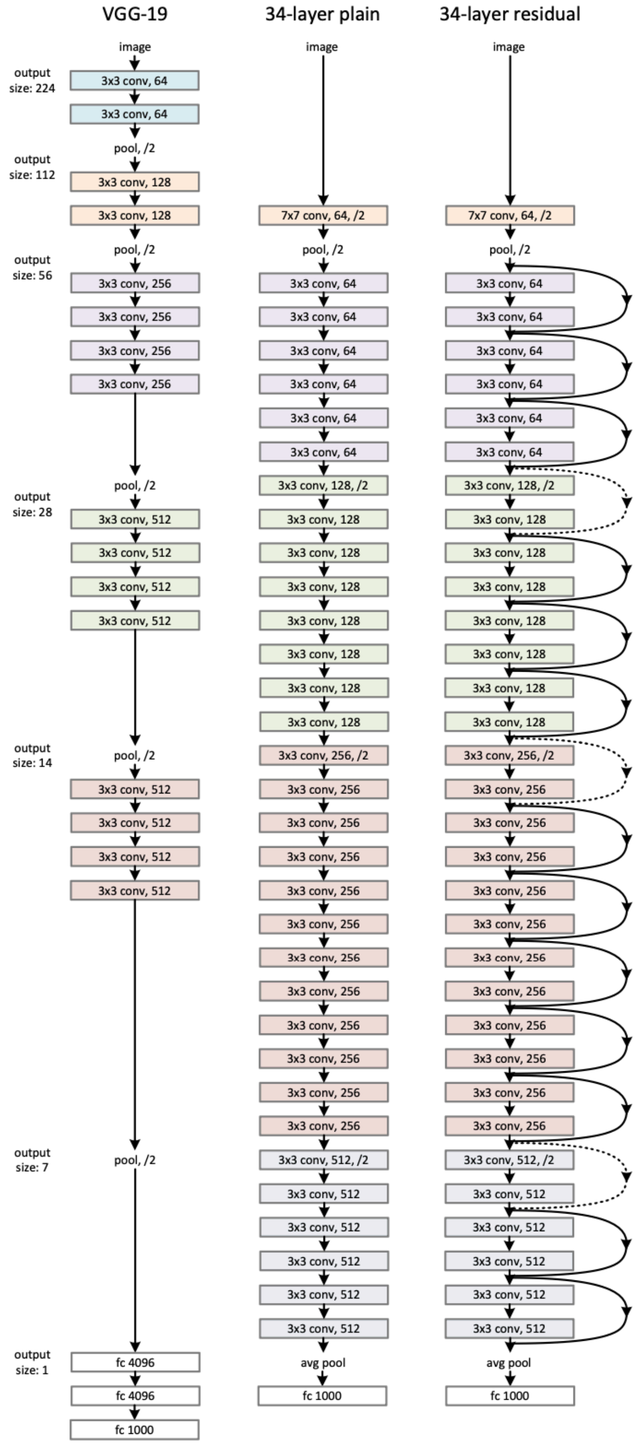
  - Convolutional layer은 3x3 filters를 갖는다.
  - 각 layer들은 같은 크기의 output feature map size와 filter 수를 가진다.
  - Feature map size가 반으로 줄어들면, filter의 수는 시간 복잡도를 유지하기 위해 두 배로 늘어난다.

  -> Fewer filter and lower complexity than VGG nets.

- **Residual Network**

  - Plain network를 기반으로 만들어졌다.

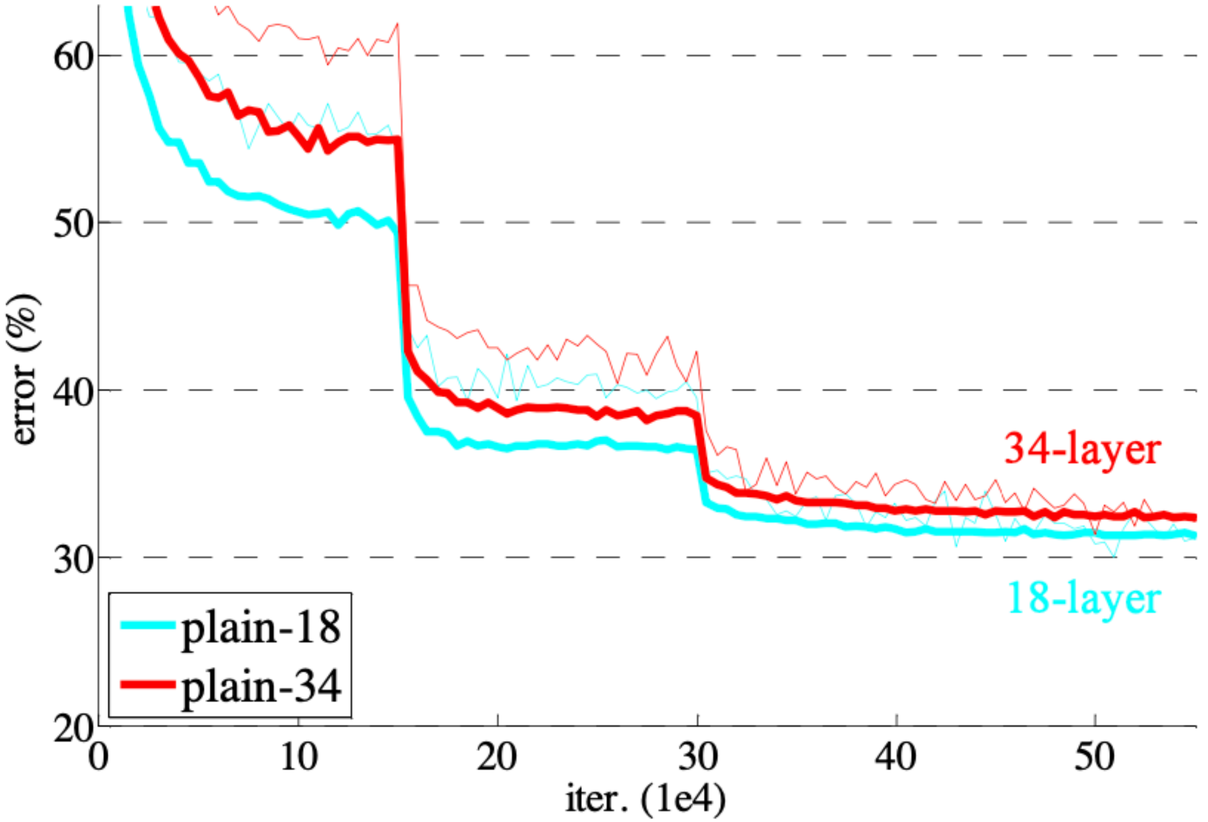
  - Shortcut connection이 추가되었다.
  - 실선 shortcut은 dimension이 같은 경우이고, 점선은 다른 경우이다.
  - dimension이 증가하면 두 가지 옵션 모두 stride는 2를 사용하며,
    (a) Identity shortcut connection -> Zero padding 수행
    (b) Projection shortcut connection -> 1x1 convolution 수행



ImageNet Classification - Plain Networks

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

Table 2. Top-1 error (%, 10-crop testing) on ImageNet validation. Here the ResNets have no extra parameter compared to their plain counterparts. Fig. 4 shows the training procedures.



*수렴율(Convergence rate)
: 최적화 기법에서 등장하는 기법으로,
수렴을 위해 필요한 에폭이나 수렴 난이도를 언급하고자 할 때 사용하는 척도

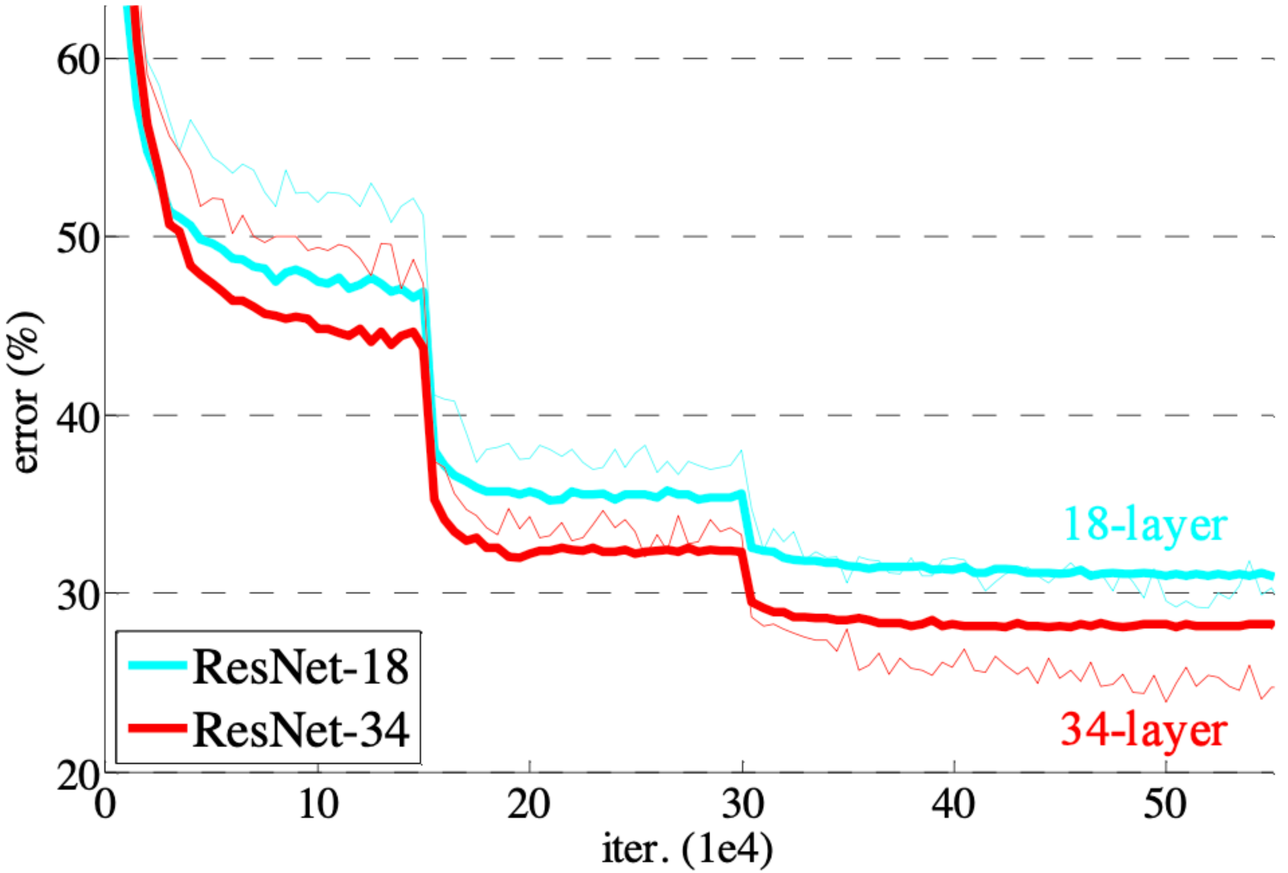
-> Plain Networks의 ImageNet Classification 결과, Degradation이 발생하고 있음을 알 수 있다.
하지만 논문에서는 원인이 vanishing gradients때문이 아니라고 언급했다.
forward와 backward를 확인했을 때, 시그널값들이 사라지는 문제가 발견되지 않았기 때문이다.
오히려 논문의 저자들은 *수렴율(convergence rates)이 기하급수적으로 낮아지는 것이 문제라고 말하고 있다.

ImageNet Classification - Residual Networks

(with zero-padding)

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

Table 2. Top-1 error (%, 10-crop testing) on ImageNet validation. Here the ResNets have no extra parameter compared to their plain counterparts. Fig. 4 shows the training procedures.



-> Residual Networks의 ImageNet Classification 결과, Degradation이 해결되었음을 알 수 있다.
다시말해, ResNet이 plain보다 빠르게 converge한다는 것이다.
ResNet은 이러한 빠른 수렴을 통해 최적화를 더 수월하게 할 수 있다.

Identity vs. Projection Shortcuts

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**

Table 3. Error rates (%, **10-crop** testing) on ImageNet validation. VGG-16 is based on our test. ResNet-50/101/152 are of option B that only uses projections for increasing dimensions.

(A) Zero-padding shortcuts are used for increasing dimensions, and all shortcuts are parameter-free

(B) Projection shortcuts are used for increasing dimensions, and other shortcuts are identity
-> Projection + Identity shortcuts

(C) All shortcuts are projections

-> All three options are considerably better than the plain counterpart.

Deeper Bottleneck Architectures

: 더 Deep한 nets을 만들기 위해 bottleneck architecture를 사용한다!

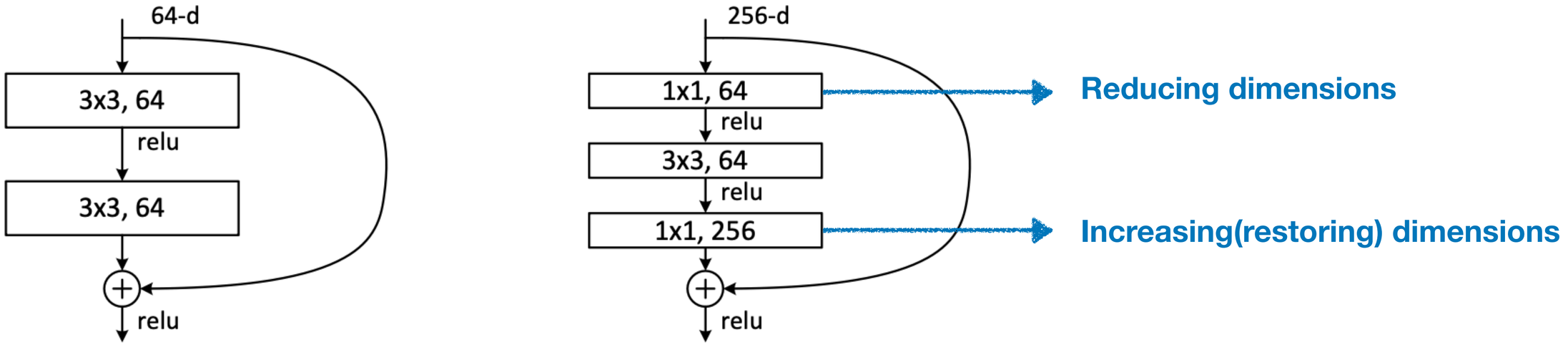


Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a "bottleneck" building block for ResNet-50/101/152.

-> 입력과 출력값의 dimension이 같은 Identity shortcut을 통해 더 deep하지만 계산량은 줄이는 결과를 얻을 수 있다.

Comparisons with State-of-the-art Methods

method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).

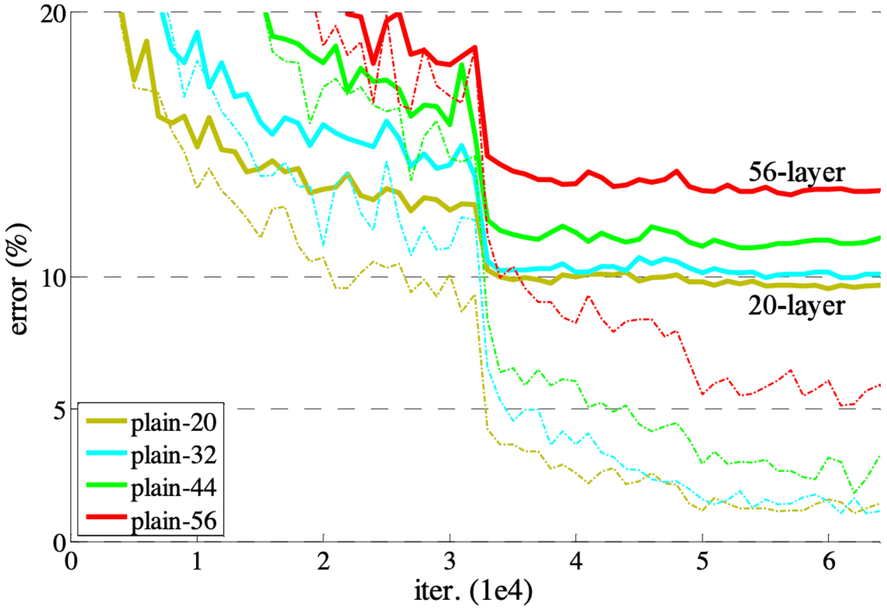
-> Bottleneck을 이용하여 더 deep한 50, 101, 152층의 ResNet을 얻어냈다. 더 deep해질수록 degradation없이 더 좋은 성능을 내고 있는 것을 확인할 수 있다.

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**

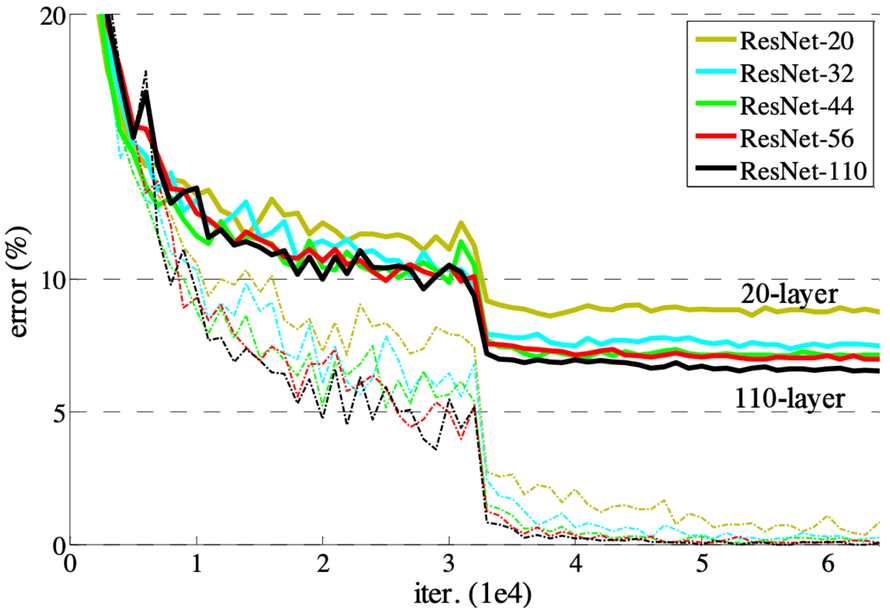
Table 5. Error rates (%) of **ensembles**. The top-5 error is on the test set of ImageNet and reported by the test server.

-> 152층의 ResNet은 다른 모델들의 ensemble보다 뛰어난 성능을 보인다. 3.57이라는 적은 error값을 확인할 수 있다.

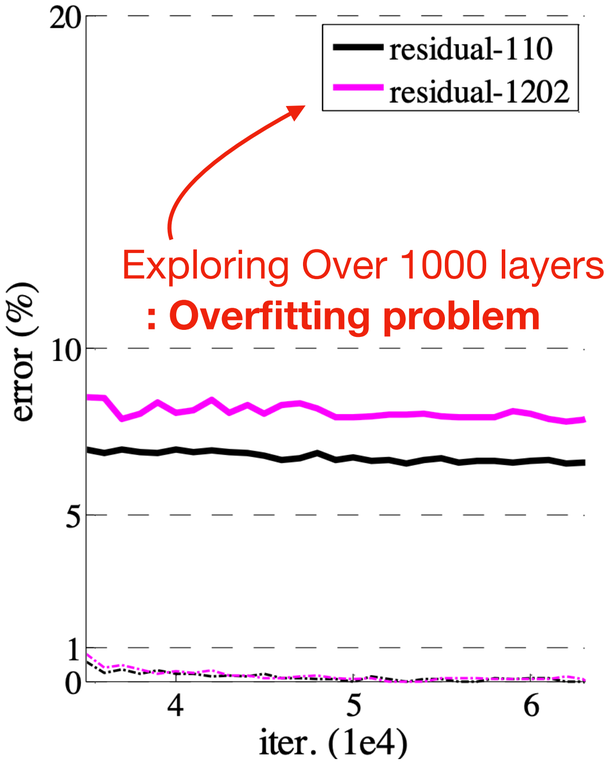
CIFAR-10 and Analysis



Plain Networks



ResNet



Residual Networks

Conclusion

ResNet은 Residual learning을 통해
Degradation problem을 해결하여 Optimization을 쉽게할 수 있도록 만들었다.

이 논문에서는 ResNet에 추가로 maxout이나 dropout등의 정규화를 사용하지 않았다고 밝혔다.
정규화를 추가한 모델로 더 좋은 성능을 낼 수 있을 것이라 언급하며 글을 마쳤다.

Thank you