Deep Residual Learning for Image Recognition

한현수

Problem : Deeper neural networks are more difficult to train , gradient vanishing/exploding , highway network

Introduction

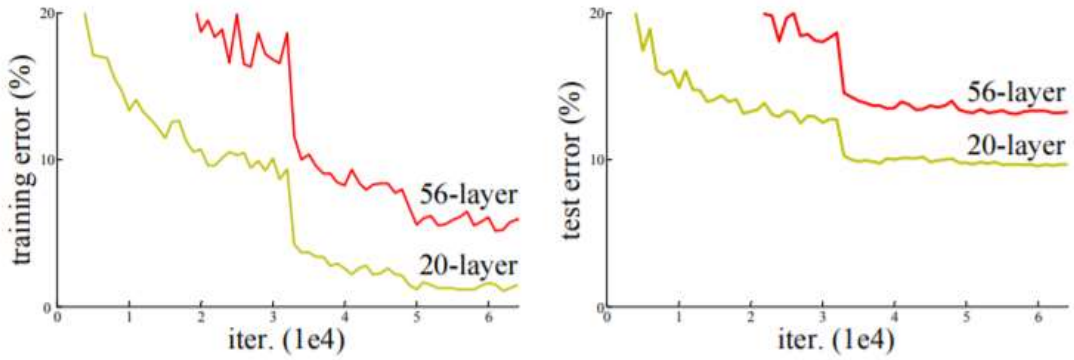


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Neural networks에서 layer가 많아질수록 좋을 것이라 생각하지만, 깊어질수록 성능은 더 좋아진다. 하지만 Gradient vanishing/exploding, degradation problem이란 현상 때문에 layer가 깊어질수록 성능이 나빠지는 현상이 나타난다. Test error와 training error 둘 다 낮기 때문에 overfitting이라고 볼 수 없고 이러한 현상들 중 degradation(layer가 깊어지면서 gradient descent가 실제 output에 반영이 잘 안되는 것)현상을 위의 그래프에서 발견할 수 있다. 이처럼 네트워크의 정확도를 높이려면 더 깊은 층을 쌓아야 하지만 특정 깊이를 넘어서면서부터 기존의 네트워크들은 잘 작동을 하지 않는다는 문제점이 있다는 것을 알 수 있었다.

Deep Residual Learning

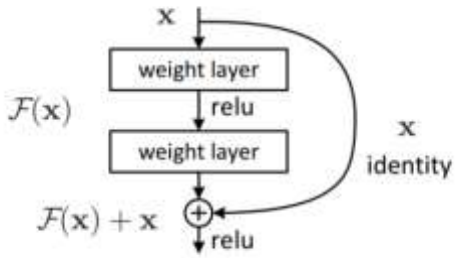
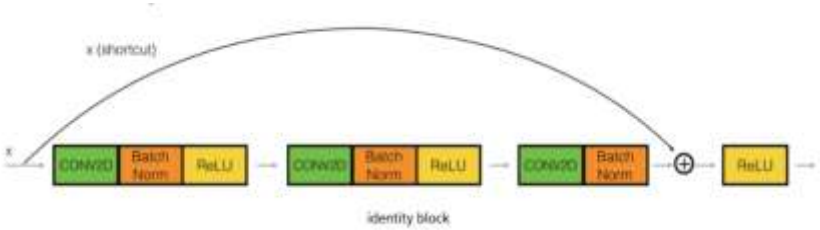


Figure 2. Residual learning: a building block.

이는 Identity shortcut connection으로 이는 H(x) = F(x) + x 구조임을 볼 수 있다. 여기서 identity를 더하는 add layer는 추가적 parameter나 computational complexity를 필요로 하지 않다는 장점이 있다. 저자는 ImageNet에서 plain net과 다르게 layer를 깊게 쌓음으로써 accuracy를 높일 수 있었다고 한다.

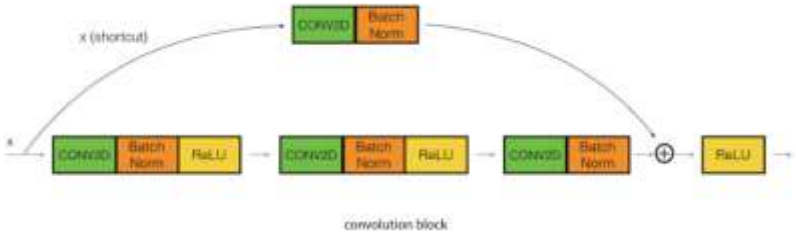
Identity shortcut

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}. \qquad (1)$$



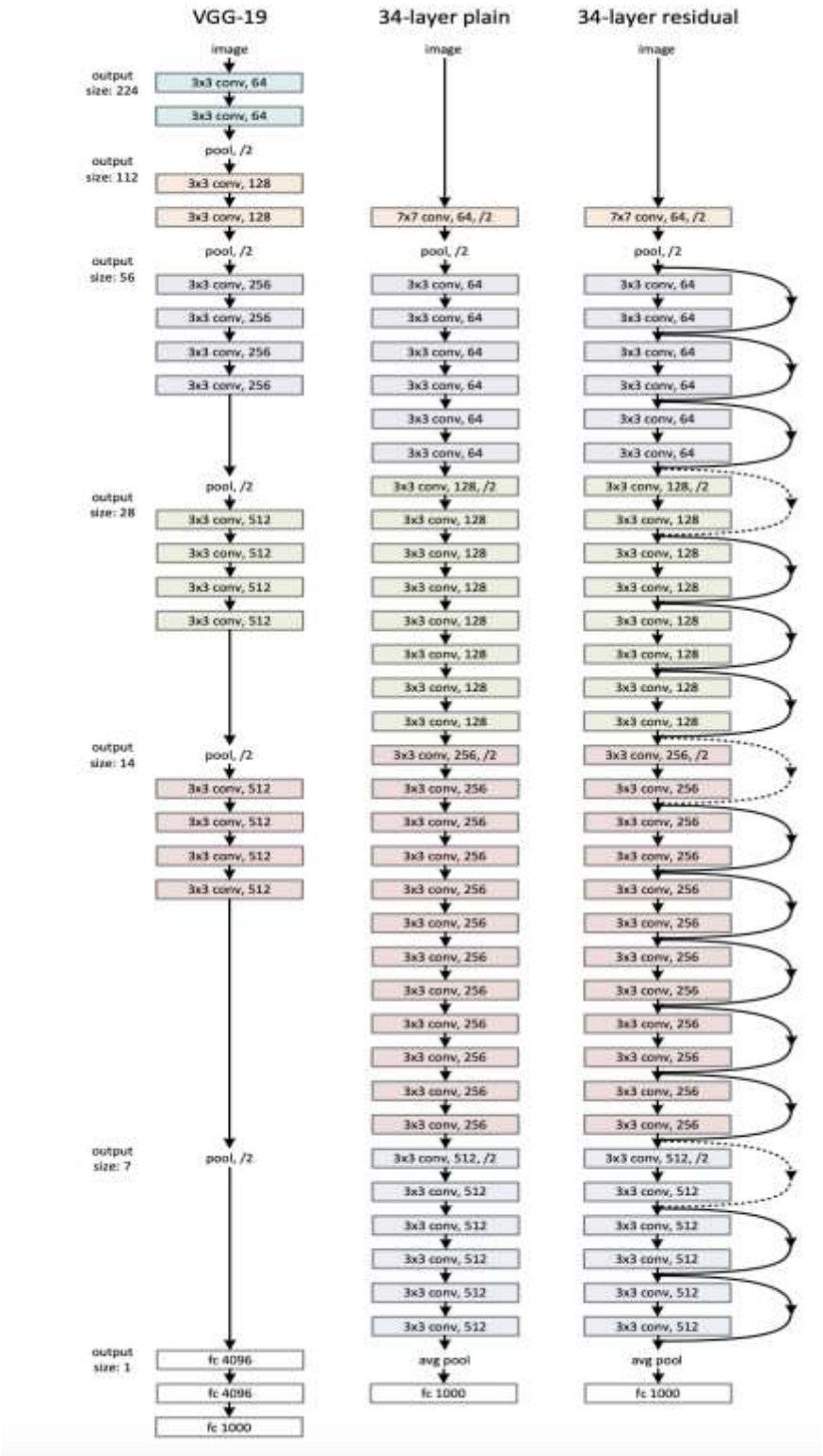
Projection shortcut

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}. \qquad (2)$$



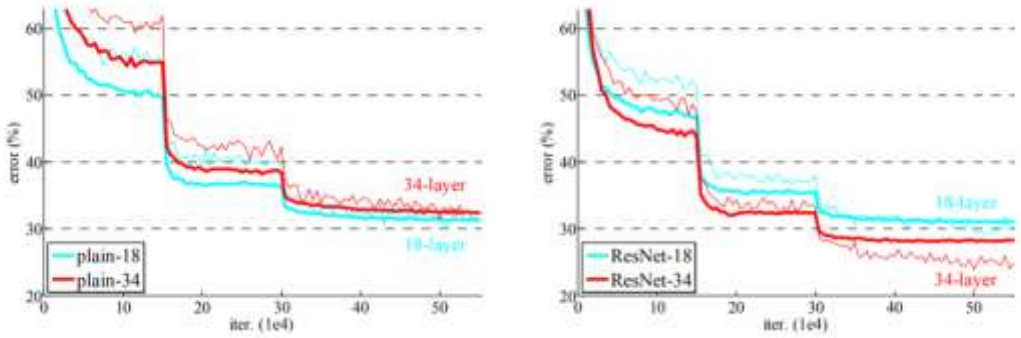
이는 x와 F를 더할 때 dimension이 다르다면 W_s(위의 그림에선 convolution layer)을 이용하여 dimension을 맞춰줄 수 있다.

Network Architectures / Implementation



기본적으로 ResNet의 구조는 VGGNET-34를 따른다. 위의 그림과 같이 ResNet은 Layer 2개마다 skip connection을 해주는 것을 볼 수 있다.

Experiment



위의 그림은 1000개의 class를 가진 ImageNet 2012데이터 셋으로 plain Network와 ResNet의 각각 layer가 18,34개인 경우로 훈련한 결과이다. 그래프에서 알 수 있다시피 plain Network는 34-layer가 18-layer보다 error가 높아 Degradation problem을 볼 수 있지만, ResNet의 경우 34-layer가 18-layer보다 error이 낮으므로 Degradation problem을 해결했다고 볼 수 있다. 논문저자는 이 실험을 통해 3가지 사실을 발견했는데, 첫번째는 ResNet이 degradation problem을 피하여 더 깊은 34layer의 성능이 좋게 나왔다고 한다. 두번째는 위의 residual learning을 적용하면 더 깊게 layer를 쌓고 좋은 성능을 낼 수 있다고 한다. 마지막으로 plain network와 ResNet과 비교했을 때 ResNet이 좀더 빨리 최고 성능에 도달하게 된다는 것이다.

Identity vs projection shortcut

model	top-1 err.	top-5 err.
VGG-16 [40]	28.07	9.33
GoogLeNet [43]	-	9.15
PReLU-net [12]	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.

위의 표에선 ResNet의 3가지 옵션(A,B,C)을 비교할 수 있다.

(A) Dimension을 증가시키기 위해 zero padding shortcut가 사용되어 추가 parameter가 없다.

(B) Dimension을 증가시키기 위해 projection shortcut을 사용하고,다른 shortcut은 identity를 사용

(C) 모든 shortcut은 projection shortcut

여기서 err의 차이는 A,B,C간의 추가적인 parameter 차이임을 알 수 있다. 하지만 차이가 근소하고 parameter가 많을수록 model이 복잡해진다는 단점이 있다.

Deeper Bottleneck Architectures

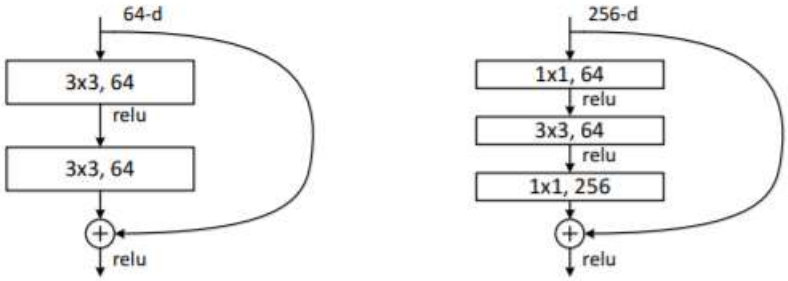


Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a "bottleneck" building block for ResNet-50/101/152.

Training time을 단축시키기 위해 building block 대신 bottleneck구조를 사용하였다. Bottleneck 구조는 위의 그림과 같이 1x1 3x3 1x1의 3-layer를 사용한다. 1x1은 차원의 증가, 감소시켜 identity shortcut을 사용할 수 있도록 한다. 만약 identity shortcut이 아닌 projection shortcut을 사용한다면 time complexity와 model size가 증가한다. 따라서 identity shortcut이 bottleneck design을 효율적으로 만든다.