ResNet

Intro: *Is learning better networks as easy as stacking more layers?*

Vanishing gradients?

Deep models with higher training error

But not overfitting

Solution by construction: identity mapping

Deeper model should produce no higher training error than its shallower counterpart

Not easy?

Intro: *Identity mapping*

Problem: Vanishing gradients?

Solution: “shortcut connections”

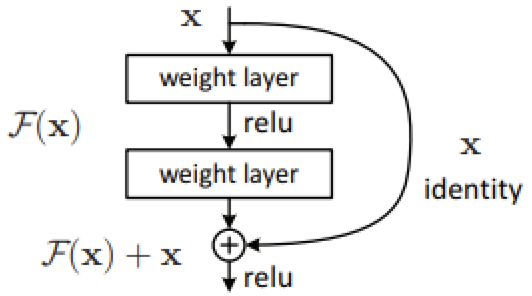


Figure 2. Residual learning: a building block.

Residual block ensures gradient of at least 1

“Easier to push residual to zero”

Q why push residual to zero? What is the benefit from it?

No extra parameter or significant computational complexity

Intro: *Identity mapping*

Problem: Vanishing gradients?

Solution: “shortcut connections”

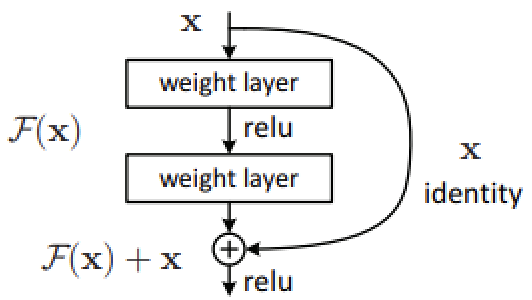


Figure 2. Residual learning: a building block.

“Easier to push residual to zero”

Q why push residual to zero? What is the benefit from it?

A “If the optimal function is closer to an identity mapping than to a zero mapping, it should be easier for the solver to find the perturbations with reference to an identity mapping, than to learn the function as a new one”

Q 무슨 소린지..

Network Architectures

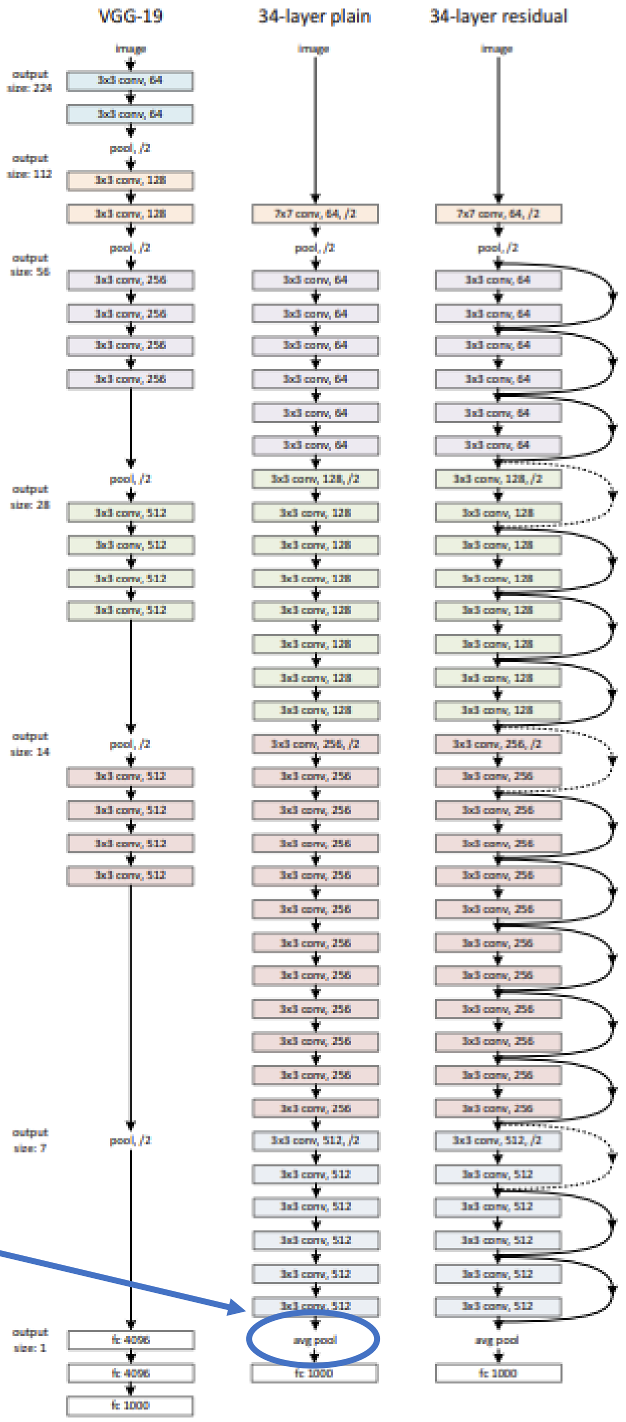
Global Average Pooling:

Commonly used to replace fc layers in convolution + fc type models

(c, h, w) → (c, 1, 1)

Faster training

Prevents overfitting

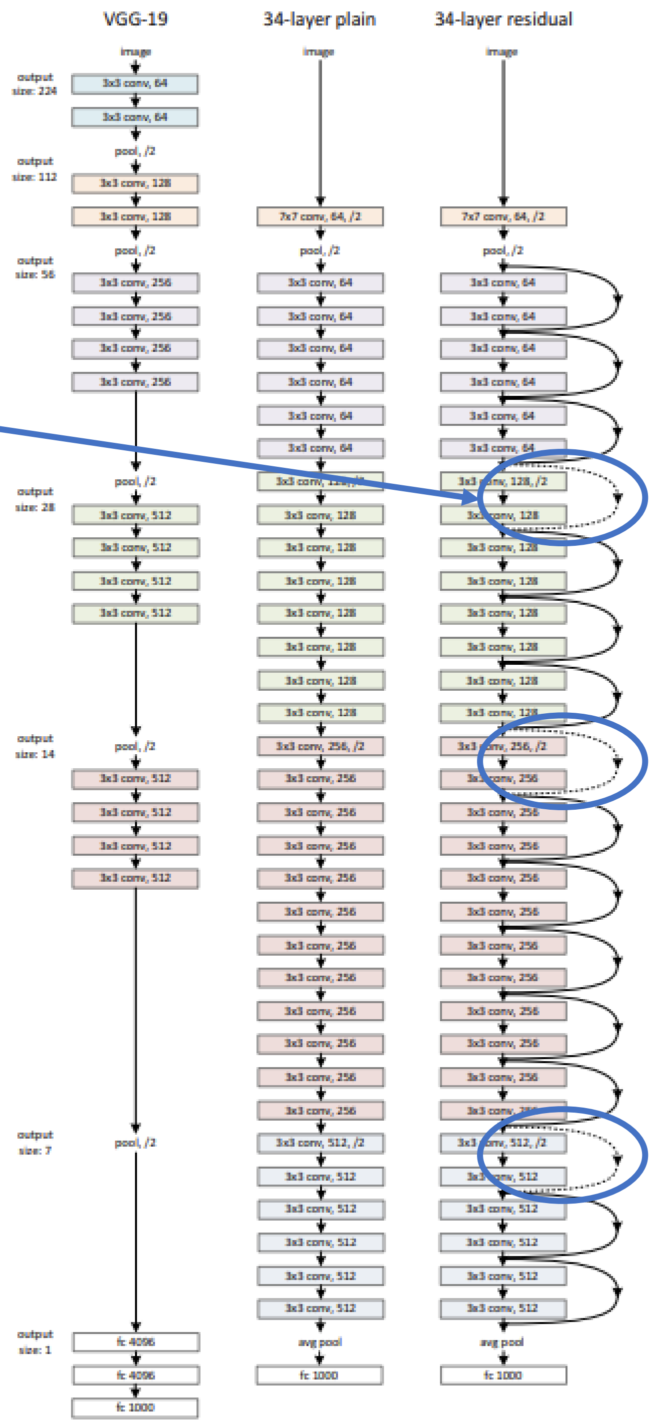


Network Architectures

Shortcut Connections when dimension increase:

(A) Extra zero entries padded for increasing dimensions

(B) Projection – 1x1 convolution to increase dimension



Network Architectures

Additionally… network details

Batch normalization in between conv and ReLU

SGD + weight decay 0.00001 momentum 0.9

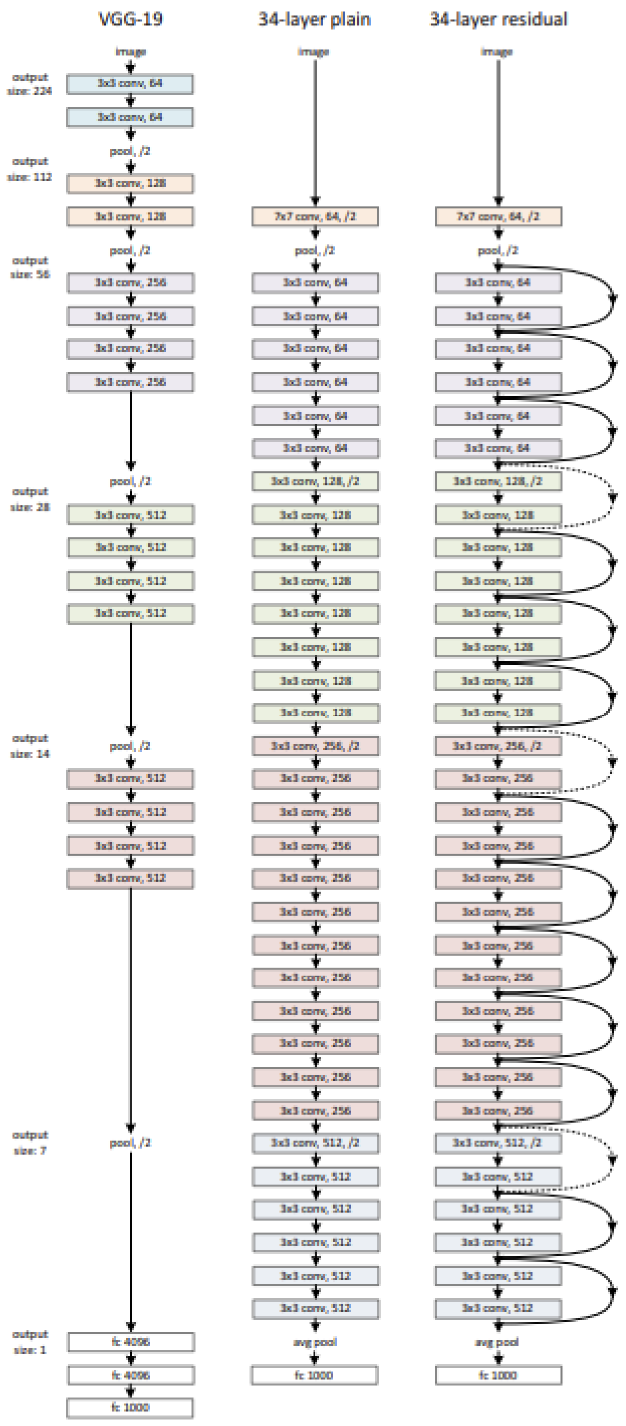
Batch size 256

Initial learning rate 0.1 with ReduceLROnPlateau

1/10 learning rate when plateau

10-crop testing

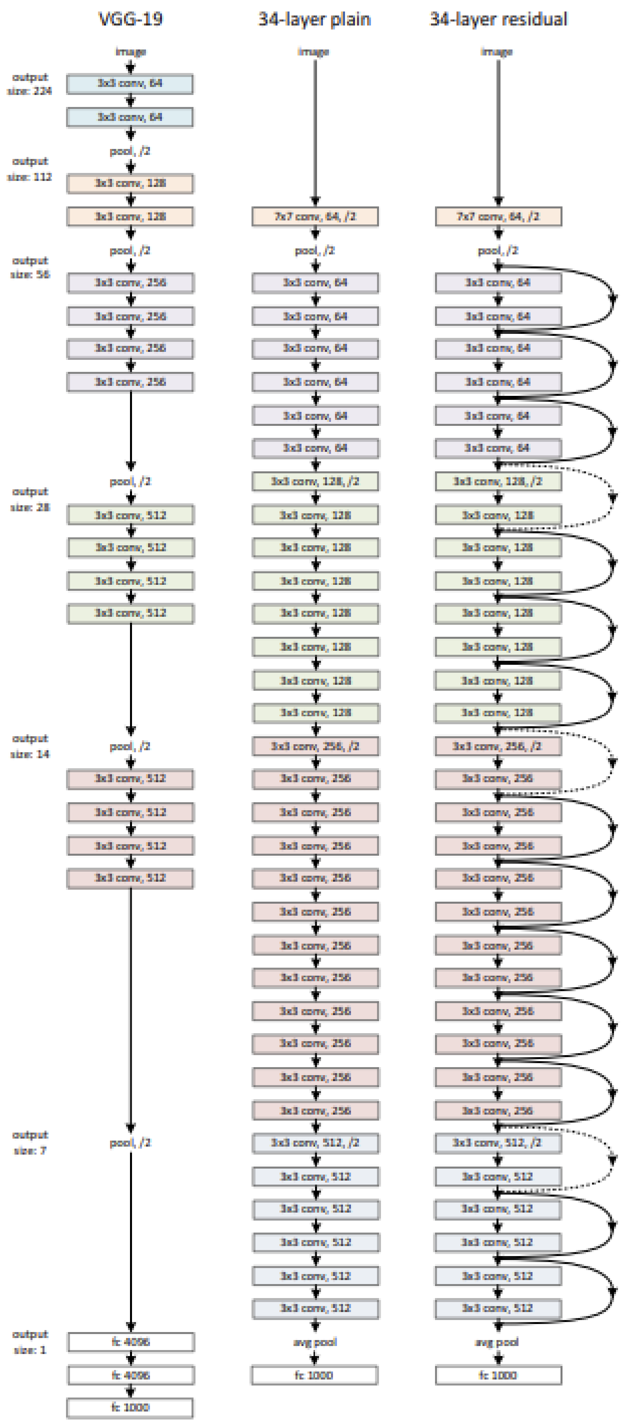
Average the scores at multiple scales



Network Architectures

Additionally… augmentations

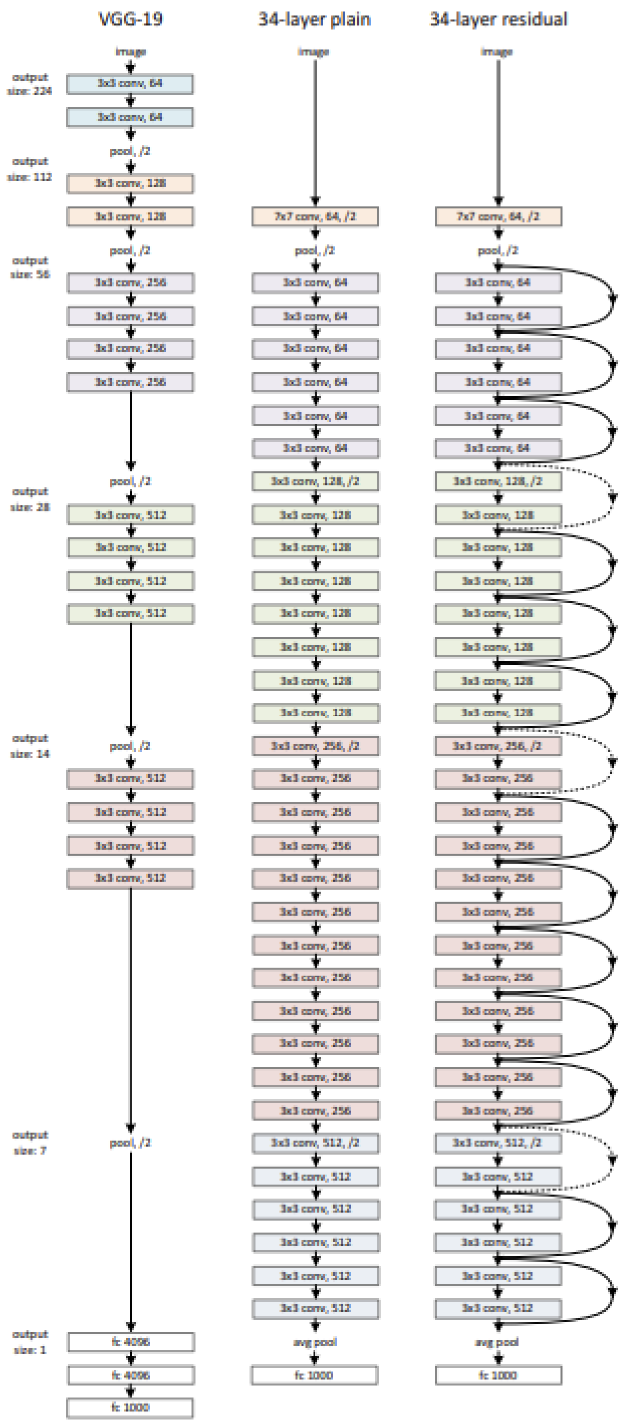
1. Resize shorter side to range of [256, 480]
2. RandomCrop 224x224
3. HorizontalFlip
4. Normalization
5. Standard Color Augmentation



Network Architectures

Additionally… augmentations

1. Resize shorter side to range of [256, 480]
2. RandomCrop 224x224
3. HorizontalFlip
4. Normalization
5. Standard Color Augmentation



Experiments

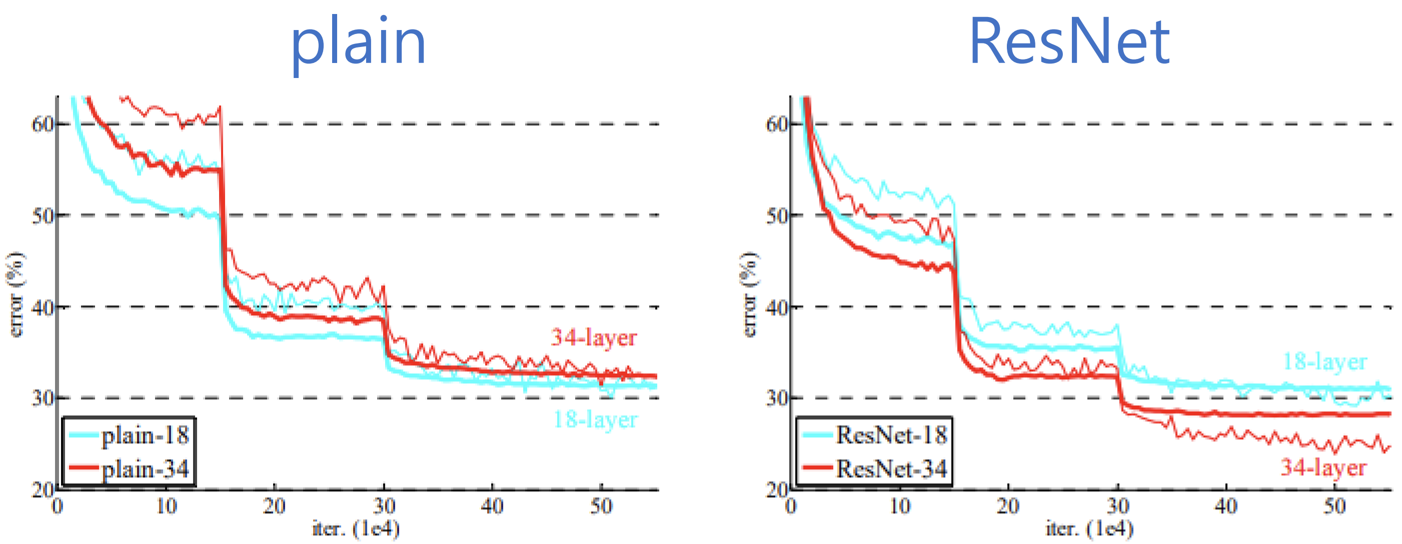


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

Vanishing gradient is not a problem because batch normalization ensures forward propagated signals to have non-zero variances

Experiments

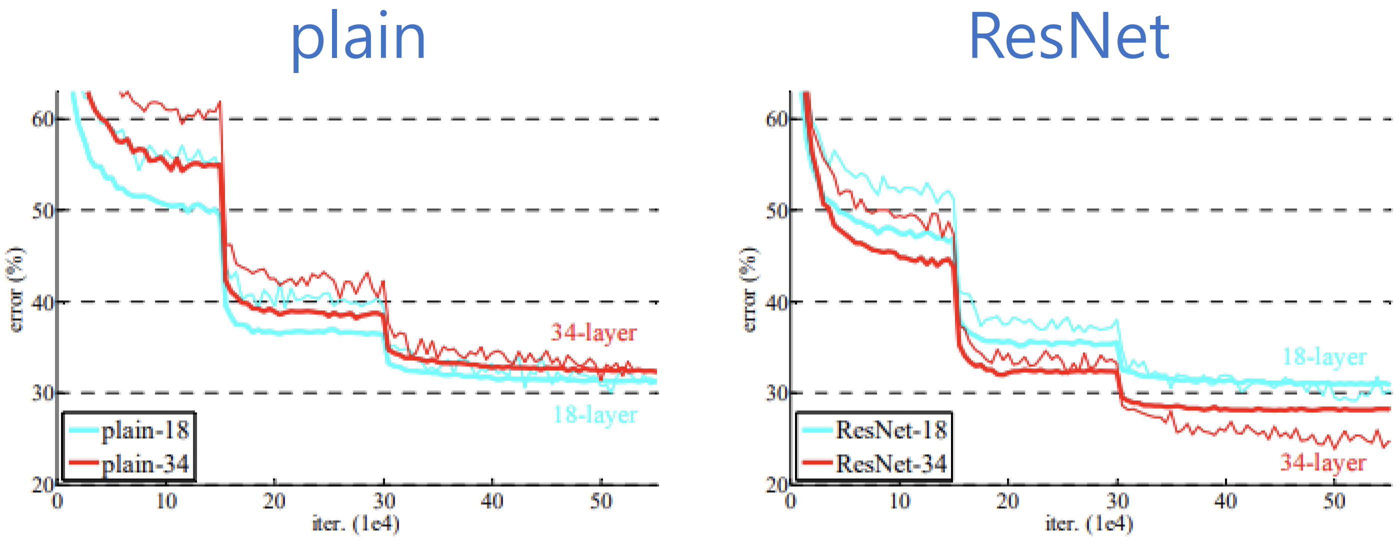
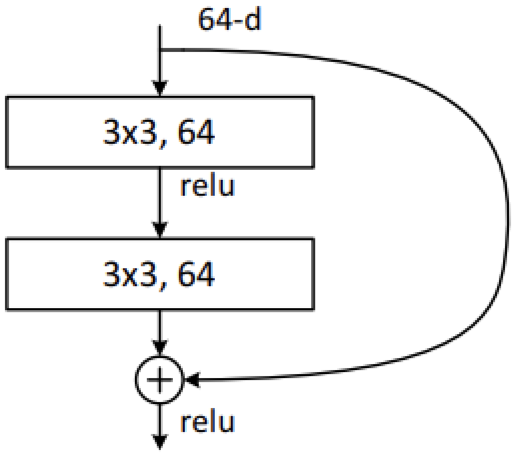


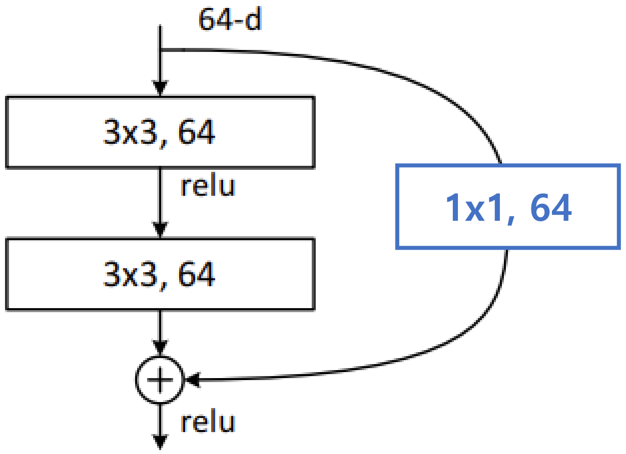
Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

Inferred that plain nets have exponentially low convergence rates

Experiments

Shortcut Types





model	top-1 err.
plain-34	28.54
ResNet-34 A	25.03
ResNet-34 B	24.52
ResNet-34 C	24.19

Best but costly

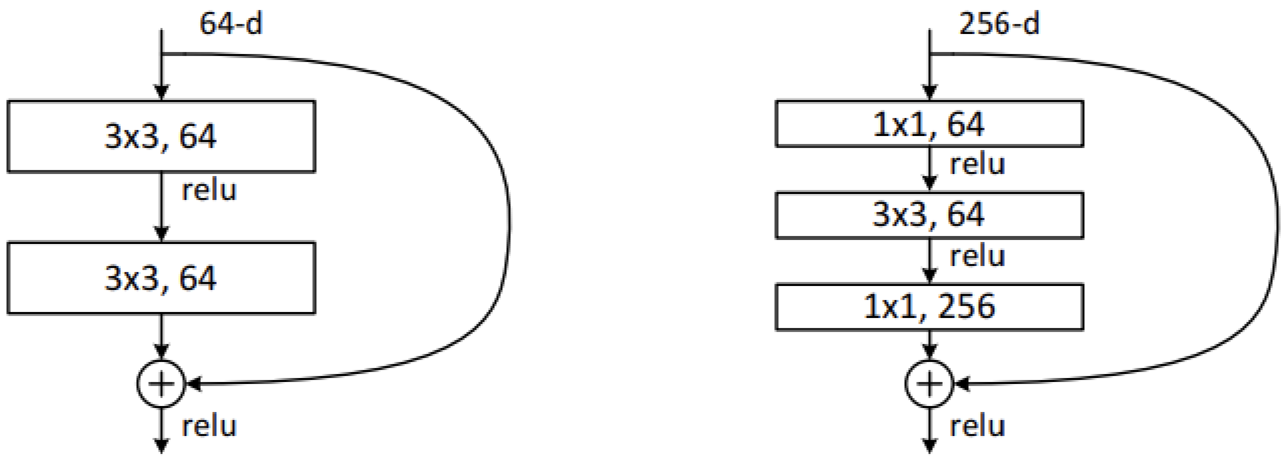
(A) zero-padding for increasing dimension / identity shortcuts

(B) Projection for increasing dimesion / identity shortcuts

(C) Projection for increasing dimesion / projection shortcuts

Experiments

Shortcut A → bottleneck



Modification from ResNet 50 upwards]

Shortcut type (B)

How about checking bottleneck on ResNet-18 / 34?

Analysis on CIFAR10

augmentations

1. 4-padding
2. RandomCrop 32x32
3. HorizontalFlip
4. Normalization
5. Standard Color Augmentation

5. Standard Color Augmentation:
(From AlexNet paper) RGB intensity shift based on PCA, eigenvalue, random Gaussian

Analysis on CIFAR10

specifications

1. SGD weight decay 0.0001 momentum 0.9
2. He initialization
3. Batch Normalization
4. Batch size 128
5. Initial earning rate 0.1
6. 1/10 learning rate at 32k / 48k iteration
7. Training done until 64k iterations

How many epochs for CIFAR10 training dataset is it?

Roughly 164 epochs on cifar10 with 128 batch size

Analysis on CIFAR10

specifications … for 110-layer ResNet

1. SGD weight decay 0.0001 momentum 0.9
2. He initialization
3. Batch Normalization
4. Batch size 128
5. Initial earning rate 0.01 until train error is below 80%
6. Then learning rate 0.01 → 1/10 learning rate at 32k / 48k iteration
7. Training done until 64k iterations

Analysis on CIFAR10

Layer responses

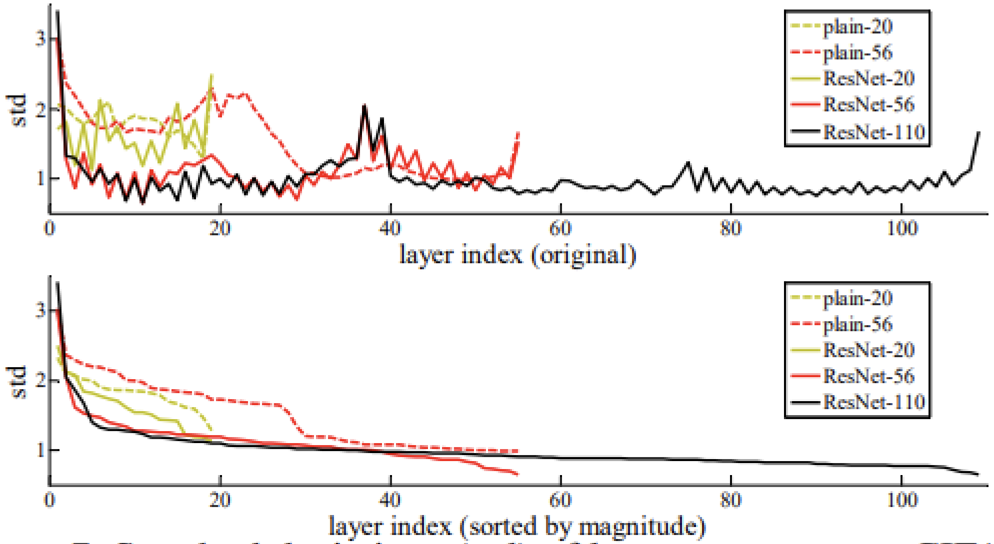


Figure 7. Standard deviations (std) of layer responses on CIFAR-10. The responses are the outputs of each 3×3 layer, after BN and before nonlinearity. **Top**: the layers are shown in their original order. **Bottom**: the responses are ranked in descending order.

Checking on output of each layer after conv and BN but before activation

ResNet has smaller responses compared to plain counterparts

→ Residual functions generally closer to zero than non-residual

Q I can't find this analysis discriminative / nor important