Deep Residual Learning for Image Recognition

2025-07-08

CVPR 2015

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun

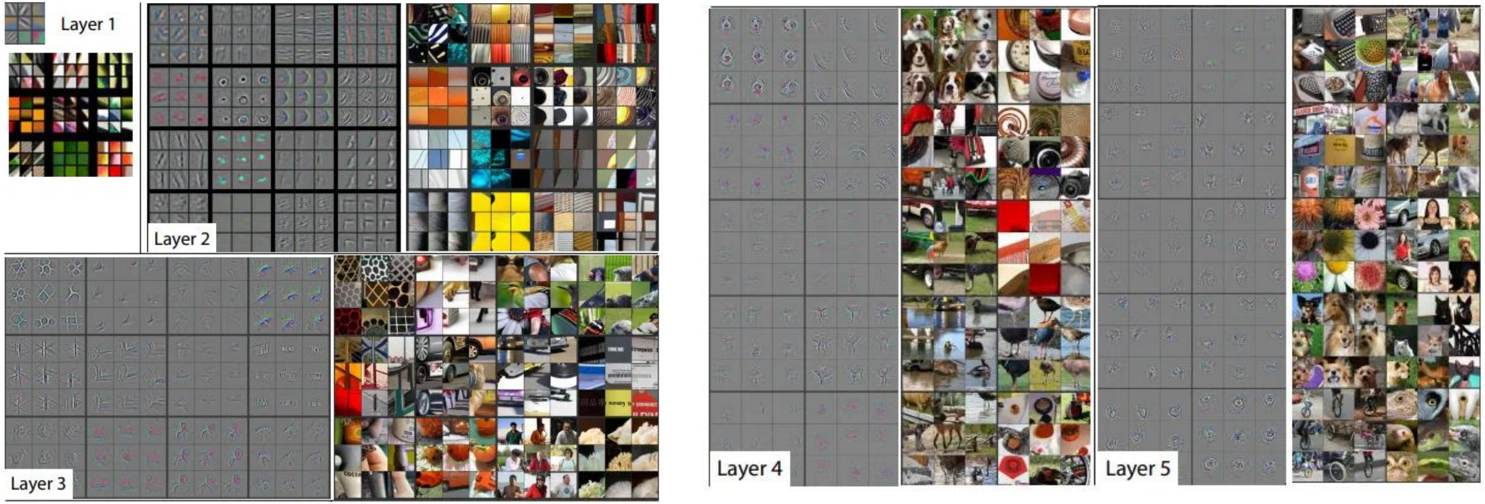
Content

- Introduction
  - Stacking Layers: Solution or Illusion?
  - Problems with adding layers
- Deep Residual Learning
- Experiments
- Conclusion

Stacking Layers: Solution or Illusion?

☐ Deep networks

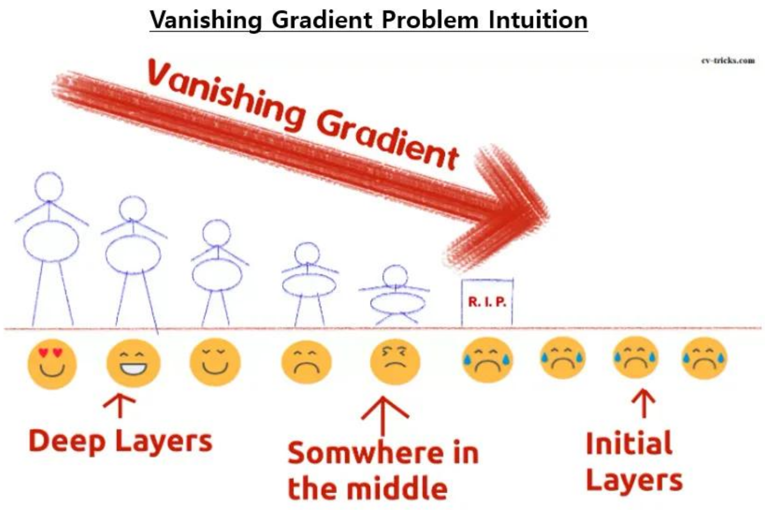
- Naturally integrate low/mid/high-level features



→ *Is learning better networks as easy as stacking more layers?*

Problems with adding layers

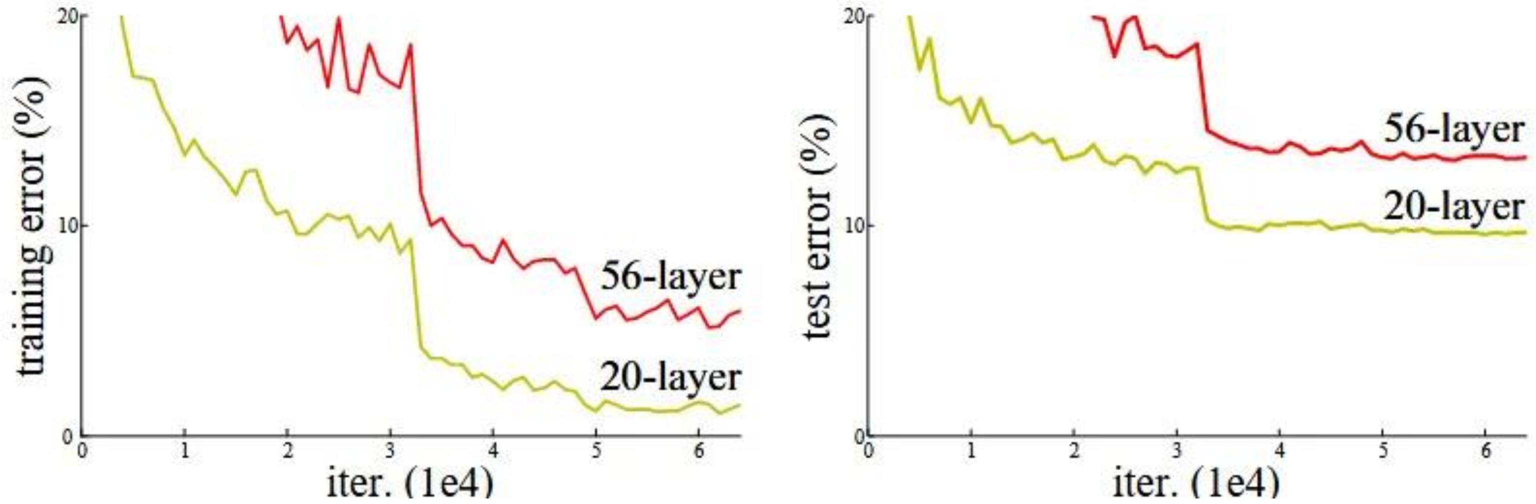
- **Vanishing/Exploding gradients**
  - Hamper convergence from the beginning



→ Can alleviate through SGD, Batch Normalization

Problems with adding layers

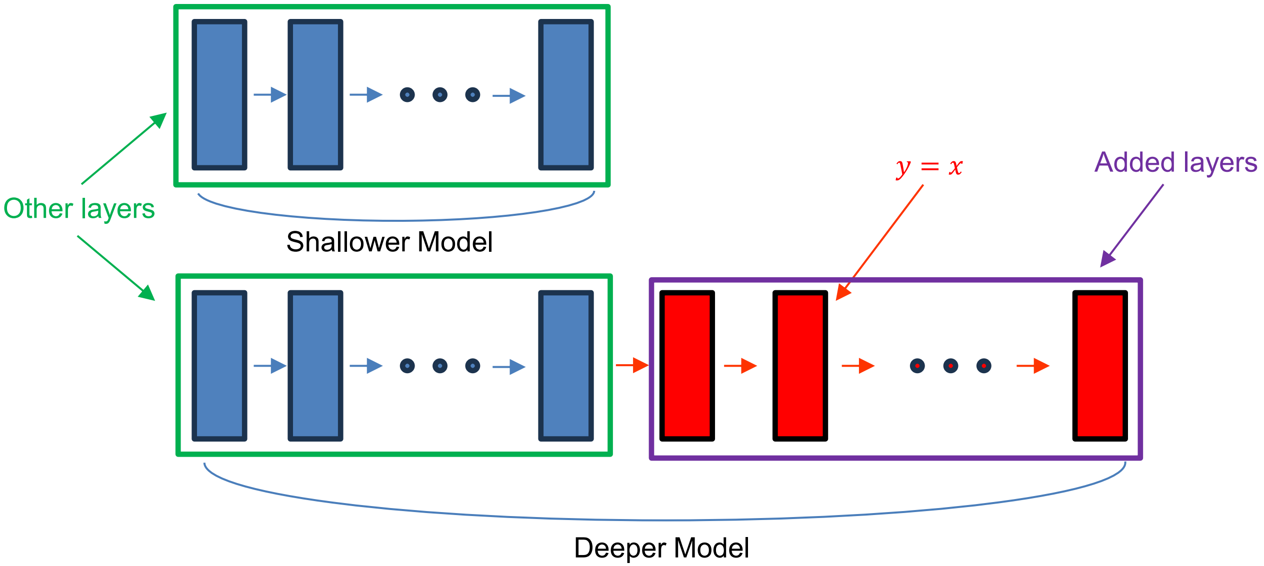
- **Degradation**
  - With the network depth increasing, accuracy gets saturated and then degrades rapidly



Problems with adding layers

☐ Degradation

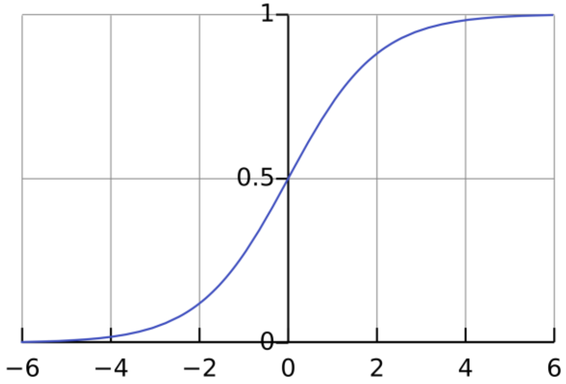
- Solution
  - ☐ Added layer : identity mapping
  - ☐ The other layer : copied from the learned shallower model



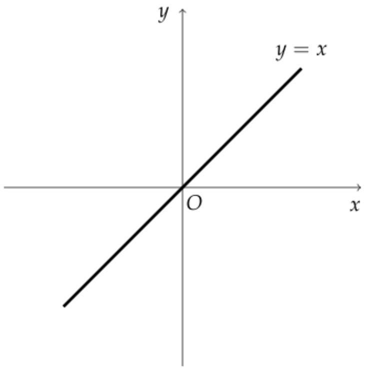
Problems with adding layers

☐ Degradation

- Difficult to approximate identity mapping by multiple nonlinear layers



$\times n \approx$



→ *When the identity mapping is optimal, wouldn't the zero-mapping approximation be easier?*

Deep Residual Learning

Residual Learning

- $\mathcal{H}(x)$
  - A mapping to be fit by a few stacked layer
- $\mathcal{F}(x)$
  - A residual function
  - $\mathcal{H}(x) - x$
  - Can involve several layers except for only one layer

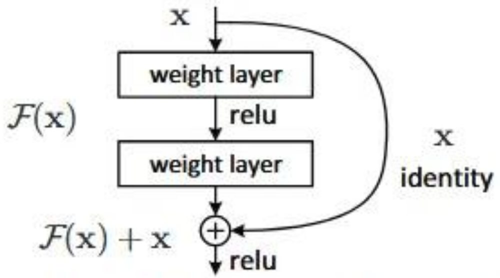


Figure 2. Residual learning: a building block.

$\rightarrow \mathcal{H}(x) = \mathcal{F}(x) + x$

$\rightarrow$ *Approximating $\mathcal{H}(x) = x$ is equal to approximating $\mathcal{F}(x) = 0$*

Deep Residual Learning

- **Shortcut**
  - Case 1. Identity mapping
    - $\dim(x) = \dim(\mathcal{F})$
    - $y = \mathcal{F}(x, \{W_i\}) + x$
    - No extra parameter
  - Case 2. Linear projection
    - $\dim(x) \neq \dim(\mathcal{F})$
    - $y = \mathcal{F}(x, \{W_i\}) + W_s x$

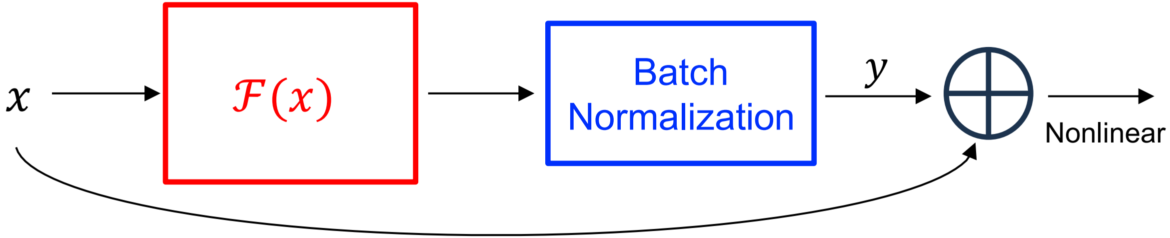
Deep Residual Learning

☐ Real case

- It is unlikely that identity mappings are optimal

  → *If $H(x)$ becomes too far from x, isn't the purpose of residual learning meaningless?*

  → *Batch normalization helps indirectly!*

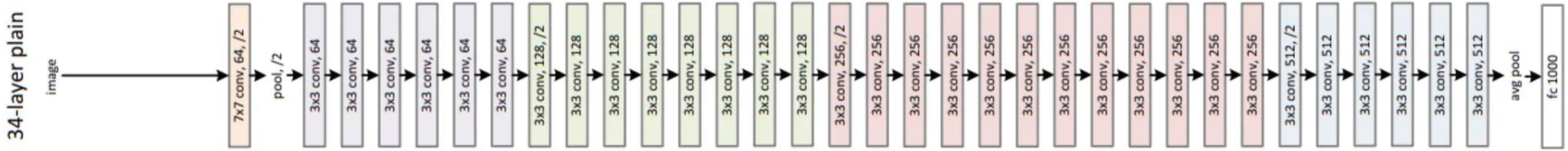


→ *Make solver to find the perturbations with reference to an identity mapping!*

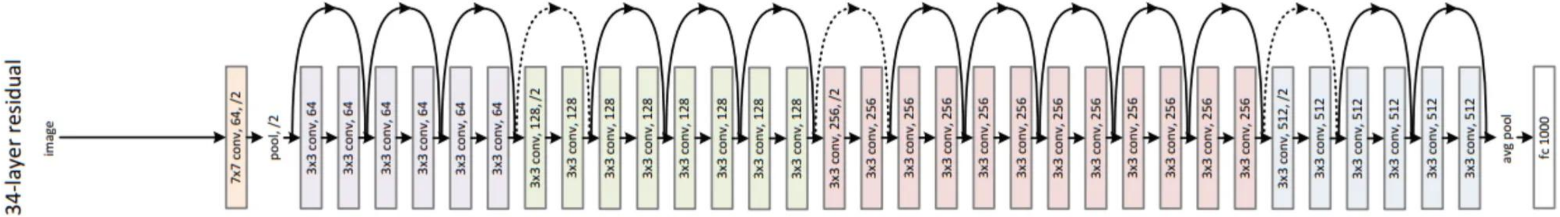
Experiments

☐ Model

- **Plain network**
  - ☐ Deploy Convolutional layers continuously



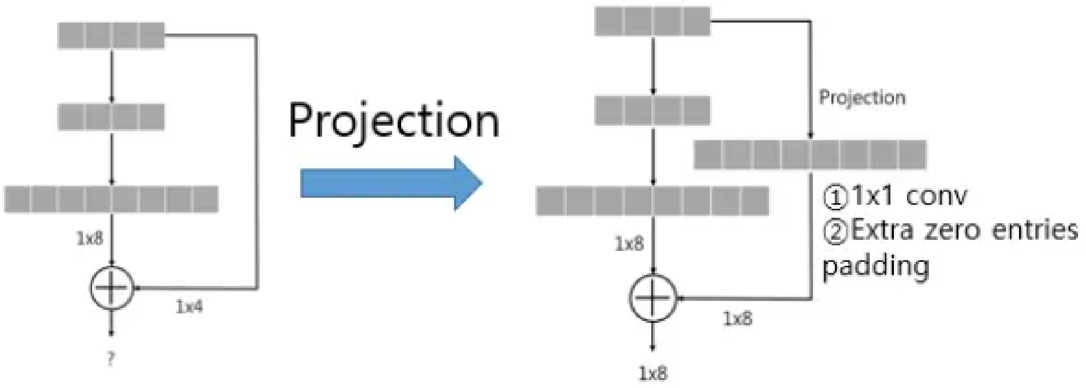
- **Residual network**
  - ☐ Add shortcut connections to the Plain network



Experiments

☐ Model

- Residual network
  - ☐ 1. Identity mapping + zero padding
  - ☐ 2. Linear projection



Experiments

☐ Model

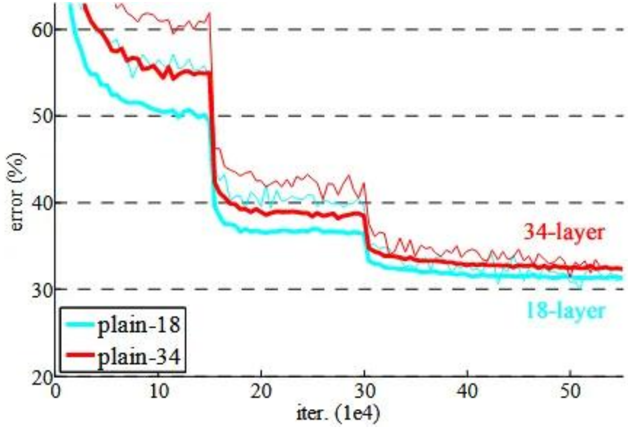
layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
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

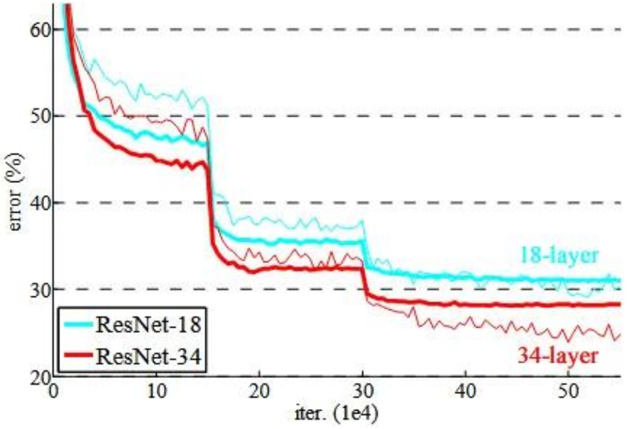
Experiments

ImageNet Classification

- Plain network (left)
  - Increase train, validation error as the number of layer grows

- ResNet (right)
  - Decrease train, validation error as the number of layer grows





Experiments

☐ Identity vs. Projection Shortcuts

- A
  - ☐ Identity + zero-padding
- B
  - ☐ Identity + Projection (when dimension increases
- C
  - ☐ Projection

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**

→ *Projection shortcuts are not essential for addressing the degradation problem!*

Experiments

☐ CIFAR-10

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

Experiments

- CIFAR-10
  - Plain network
    - Increase train, validation error as the number of layer grows
  - ResNet
    - Train error converges to almost zero, and test error decreases

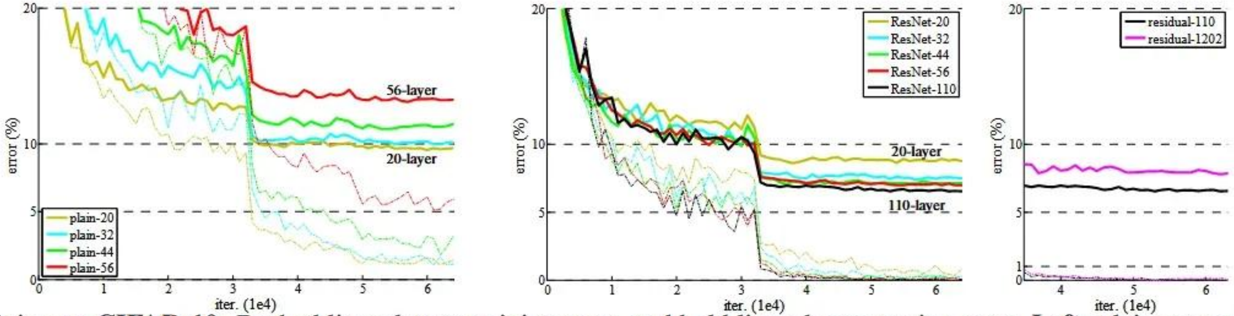
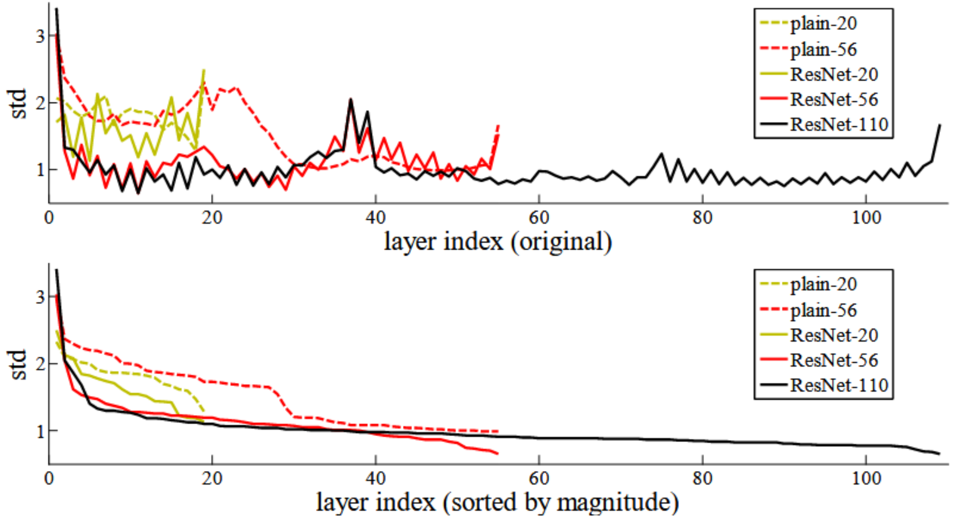


Figure 6. Training on **CIFAR-10**. Dashed lines denote training error, and bold lines denote testing error. **Left**: plain networks. The error of plain-110 is higher than 60% and not displayed. **Middle**: ResNets. **Right**: ResNets with 110 and 1202 layers.

Experiments

CIFAR-10

- Analysis of layer responses
  - ResNets have generally smaller responses than their plain counterparts



Conclusion

Problem with adding layers

- Vanishing/Exploding gradients, Degradation

Deep Residual Learning

- Learn $\mathcal{F}(x) \coloneqq \mathcal{H}(x) - x$
- Make solver to find the perturbations with reference to an identity mapping

Experiments

- Decrease train, validation error as the number of layer grows