Deep Residual Networks and Identity Mappings

Pengfei Yin

September 3, 2019

References

[1]He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016a.

[2]He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pp. 630–645. Springer, 2016b.

Problems of the previous CNN models

1. Deeper neural networks are more difficult to train. VGG has at most 19 layers.
2. Optimization becomes very hard due to the gradient vanishing problems.
3. Current architectures are not suitable for very deep nets.

Cannt go deeper! Bottleneck is training and optimization!

Degradation problem

The notorious problem of **vanishing/exploding gradients** hampers convergence from the beginning. This problem, however, has been largely addressed by **normalized initialization** and **intermediate normalization layers**, which enable networks with tens of layers to start converging for stochastic gradient descent (SGD).
However, degradation problems emerge when network goes deeper: **Accuracy gets saturated and then degrades rapidly!**

Such degradation is not caused by overfitting. Not all systems are similarly easy to optimize.

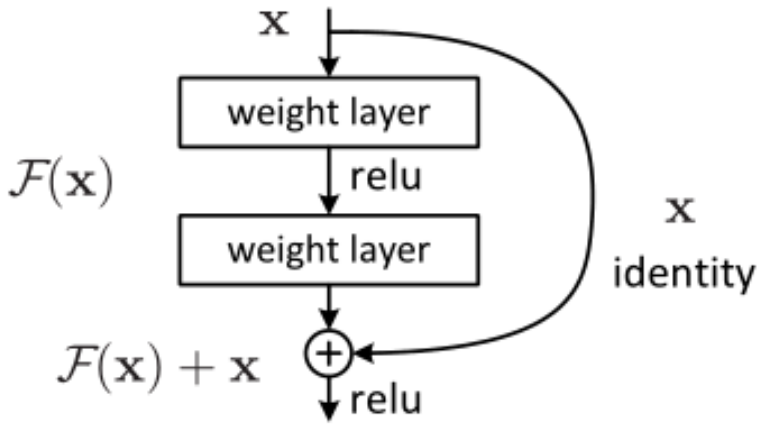


Figure: Residual learning: a building block

Instead of hoping each few stacked layers directly fit a desired underlying mapping $\mathcal{H}(x)$, we explicitly let these layers fit a residual mapping: $\mathcal{F}(x)$.

$$\mathcal{F}(x) := \mathcal{H}(x) - x \tag{1}$$

It would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

Key ideas of ResNet:

1. Residual Representations.
2. Shortcut connections.

Advantages:

1. Easy to optimize.
2. Can easily enjoy accuracy gains from greatly increased depth.
3. Identity shortcuts introduce neither extra parameter nor computation complexity.

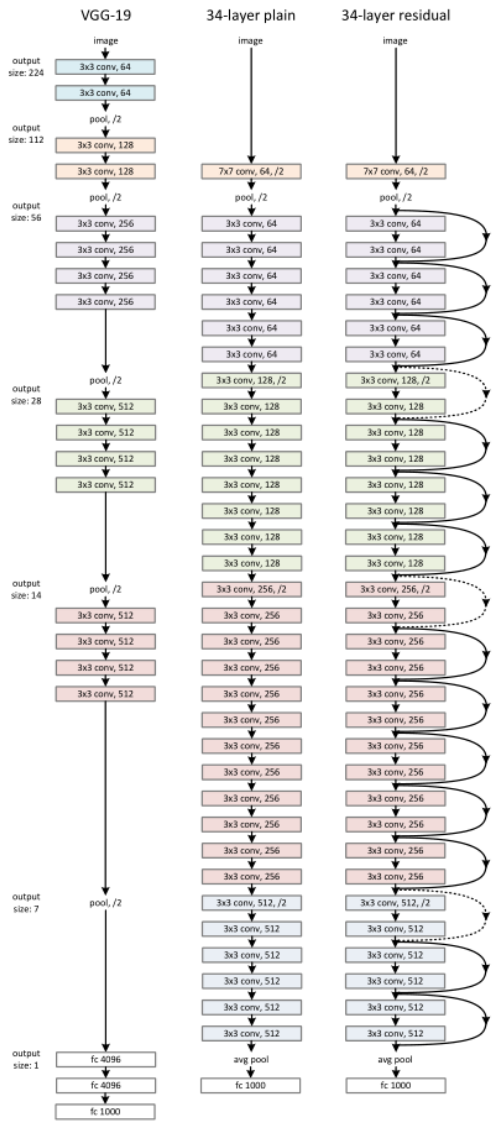


Figure: 34-layer ResNet

The buiding block:

$$y = \mathcal{F}(x, \{W_i\}) + x \tag{2}$$

$\mathcal{F}$ is the residual mapping to be learnd. For two layers,
$\mathcal{F} = W_2\sigma(W_1 x)$.
If the input and output are of different dimension (dotted line).
We perform a linear projection W_s.

$$y = \mathcal{F}(x, \{W_i\}) + W_s x \tag{3}$$

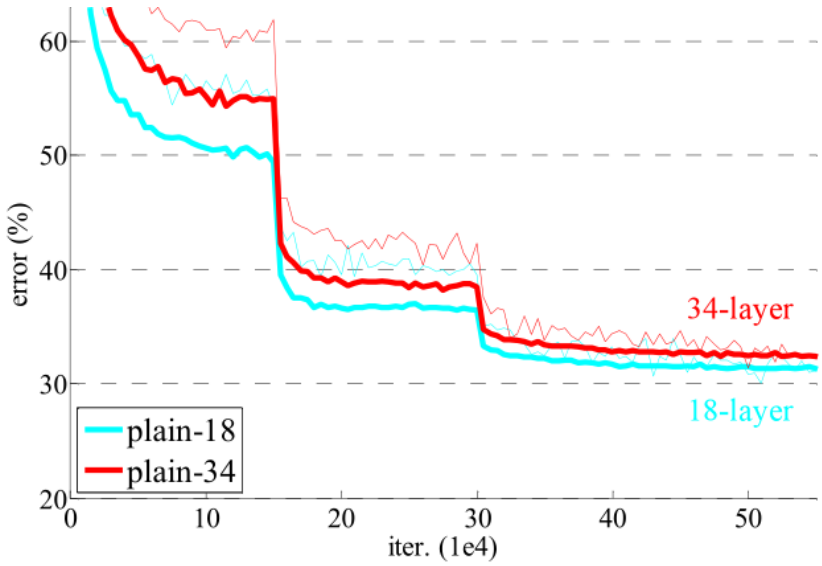
and consider two options:

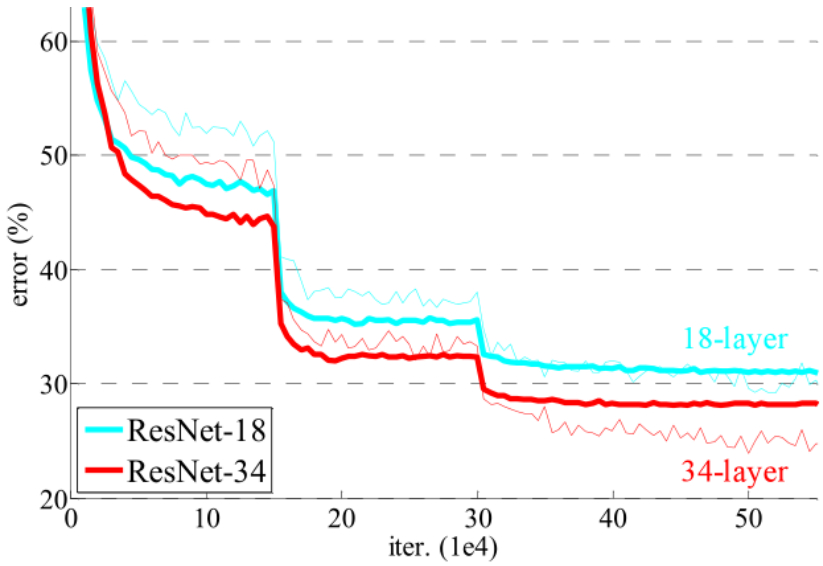
(A) The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter;

(B) The projection shortcut in Eqn.(3) is used to match dimensions (done by 1×1 convolutions).

Plain and Residual Networks

The degradation problem : **34-layer plain net has higher training error throughout the whole training procedure.**





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

We conjecture that the deep plain nets may have **exponentially low convergence rates**, which impact the reducing of the training error. The reason for such optimization difficulties will be studied in the future.

Identity vs. Projection Shortcuts

(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.

(C) All shortcuts are projections.

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

The small differences among A/B/C indicate that projection shortcuts are not essential for addressing the degradation problem.

Deeper Bottleneck Architectures

To leave the 3×3 layer a bottleneck with smaller input/output dimensions.

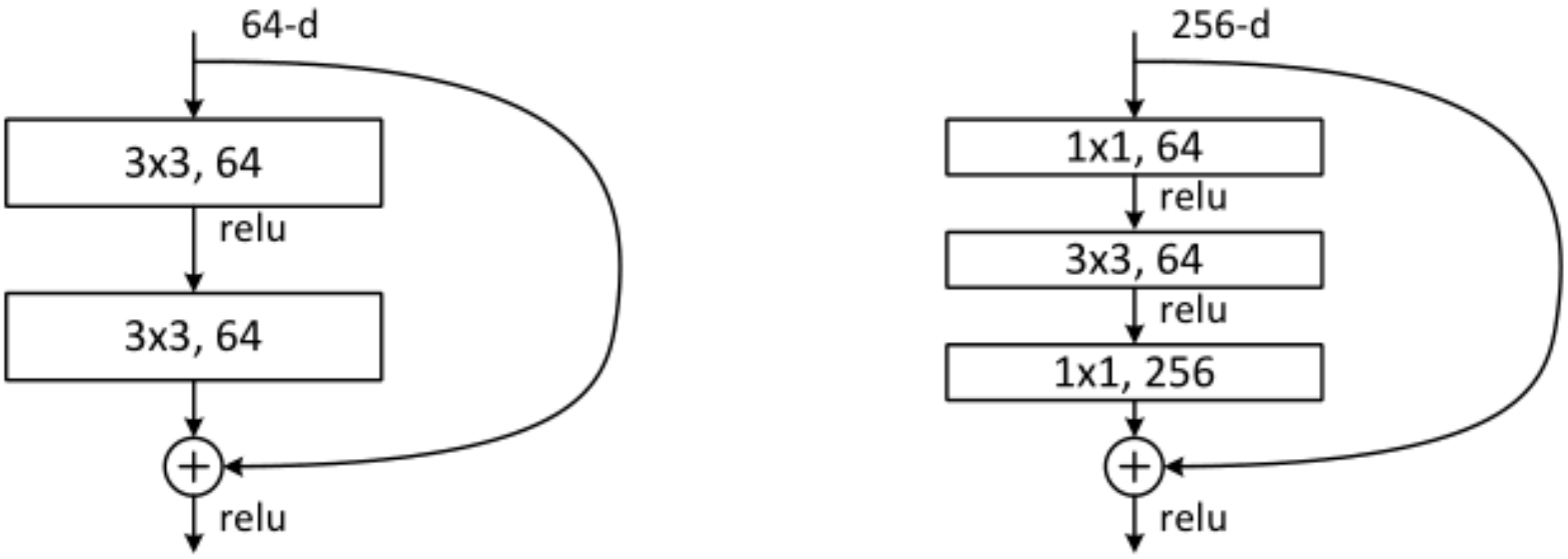


Figure: Left: a building block (on 56×56 feature maps) for ResNet- 34. Right: a "bottleneck" building block for ResNet-50/101/152.

Both designs have similar time complexity.
The parameter-free identity shortcuts are particularly important for the bottleneck architectures. If the identity shortcut in Right is replaced with projection, one can show that the time complexity and model size are doubled, as the shortcut is connected to the two high-dimensional ends. So identity shortcuts lead to more efficient models for the bottleneck designs.

ResNet-50/101/152

50-layer ResNet:
We replace each 2-layer block in the 34-layer net with this 3-layer bottleneck block, resulting in a 50-layer ResNet (Table 1). We use option B for increasing dimensions. This model has 3.8 billion FLOPs.

101-layer and 152-layer ResNets:
We construct 101-layer and 152-layer ResNets by using more 3-layer blocks. Remarkably, although the depth is significantly increased, the 152-layer ResNet (11.3 billion FLOPs) still has lower complexity than VGG-16/19 nets (15.3/19.6 billion FLOPs).

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**

The 50/101/152-layer ResNets are more accurate than the 34-layer ones by considerable margins.
We do not observe the degradation problem and thus enjoy significant accuracy gains from considerably increased depth. The benefits of depth are witnessed for all evaluation metrics.

Exploring Over 1000 layers

method			error (%)
Maxout [9]			9.38
NIN [25]			8.81
DSN [24]			8.22
	# layers	# params	
FitNet [34]	19	2.5M	8.39
Highway [41, 42]	19	2.3M	7.54 (7.72±0.16)
Highway [41, 42]	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

Figure: Classification error on the CIFAR-10 test set.

But there are still open problems on such aggressively deep models. The testing result of this 1202-layer network is worse than that of our 110-layer network, although both have similar training error. We argue that this is because of **Overfitting**.

Identity Mappings in Deep Residual Networks

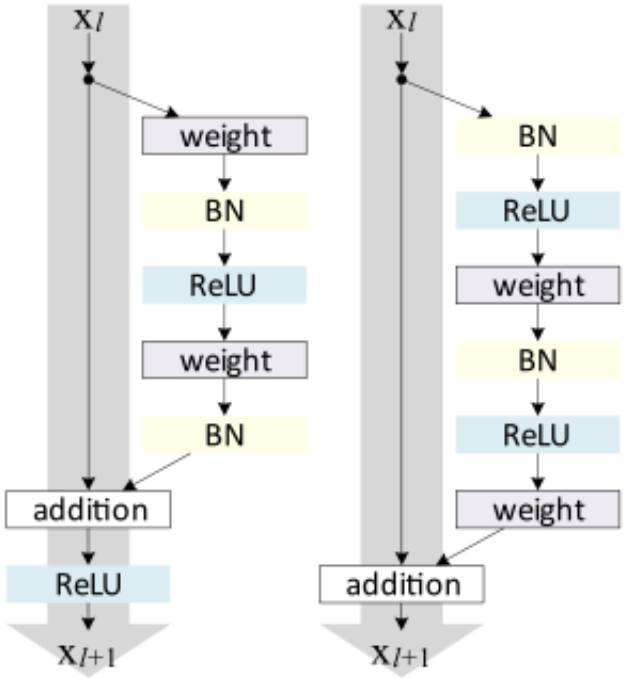
The stacked "Residual Units" in ResNets can be expressed in a general form:

$$y_l = h(x_l) + \mathcal{F}(x_l, \mathcal{W}_l) \tag{4}$$

$$x_{l+1} = f(y_l) \tag{5}$$

$\boldsymbol{x_l}$ and $\boldsymbol{x_{l+1}}$: input and output.
$\boldsymbol{\mathcal{F}}$: A residual fucntion.
$\boldsymbol{h}$: usually chosen to be identity.
$\boldsymbol{f}$: usually chosen to be ReLU.

To understand the role of skip connections, we analyze and compare various types of $h(x_l)$.



(a) original (b) proposed

To understand the role of skip connections, we analyze and compare various types of $h(x_l)$. We find that the identity mapping $h(x_l) = x_l$ achieves the fastest error reduction and lowest training loss among all variants we investigated, whereas skip connections of **scaling**, **gating**, and **1×1 convolutions** all lead to higher training loss and error.

Analysis of Deep Residual Networks

The function h is set as an identity mapping: $h(x_l) = x_l$. If f is also an identity mapping: $x_{l+1} \equiv y_l$, then (4)(5) becomes:

$$x_{l+1} = x_l + \mathcal{F}(x_l, \mathcal{W}_l) \tag{6}$$

Recursively we have:

$$x_L = x_l + \sum_{i=l}^{L-1} \mathcal{F}(x_i, \mathcal{W}_i) \tag{7}$$

Nice properties:

(i) **Residual Fashion** : Any deeper unit x_L can be represented as any shallower unit x_l plus a residual function in a form of $\sum_{i=1}^{L-1} \mathcal{F}$.

(ii) x_L is the **summation** of the outputs of all preceding residual functions(plus x_0), which is in contrast to a "plain network" where a feature x_L is a series of matrix-vector products $\prod_{i=0}^{L-1} W_i x_0$.

(iii) (7) also leads to **nice backward propagation properties**.

$$\frac{\partial \mathcal{E}}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1}\mathcal{F}(x_i, \mathcal{W}_i)\right) \tag{8}$$

The term $\frac{\partial \mathcal{E}}{\partial x_L}$ propagates information directly without concerning any weight layers, and another term $\frac{\partial \mathcal{E}}{\partial x_L}(\frac{\partial}{\partial x_l}\sum_{i=l}^{L-1}\mathcal{F}(x_i, \mathcal{W}_i))$ propagates through the weight layers.
The additive term of $\frac{\partial \mathcal{E}}{\partial x_L}$ ensures that information is directly propagated back to any shallower unit l. This implies that the gradient of a layer does not vanish even when the weights are arbitrarily small.

On the Importance of Identity Skip Connections

A simple modification to break the identity shortcut : $h(x_l) = \lambda_l x_l + \mathcal{F}(x_l, \mathcal{W}_l)$. Similarly,

$$x_L = (\prod_{i=l}^{L-1} \lambda_i) x_l + \sum_{i=l}^{L-1} \hat{\mathcal{F}}(x_i, \mathcal{W}_i) \tag{9}$$

where the notation $\hat{\mathcal{F}}$ absorbs the scalars into the residual functions. And

$$\frac{\partial \mathcal{E}}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L} \left((\prod_{i=l}^{L-1} \lambda_i) + \frac{\partial}{\partial x_l} \sum_{i=l}^{L-1} \hat{\mathcal{F}}(x_i, \mathcal{W}_i) \right) \tag{10}$$

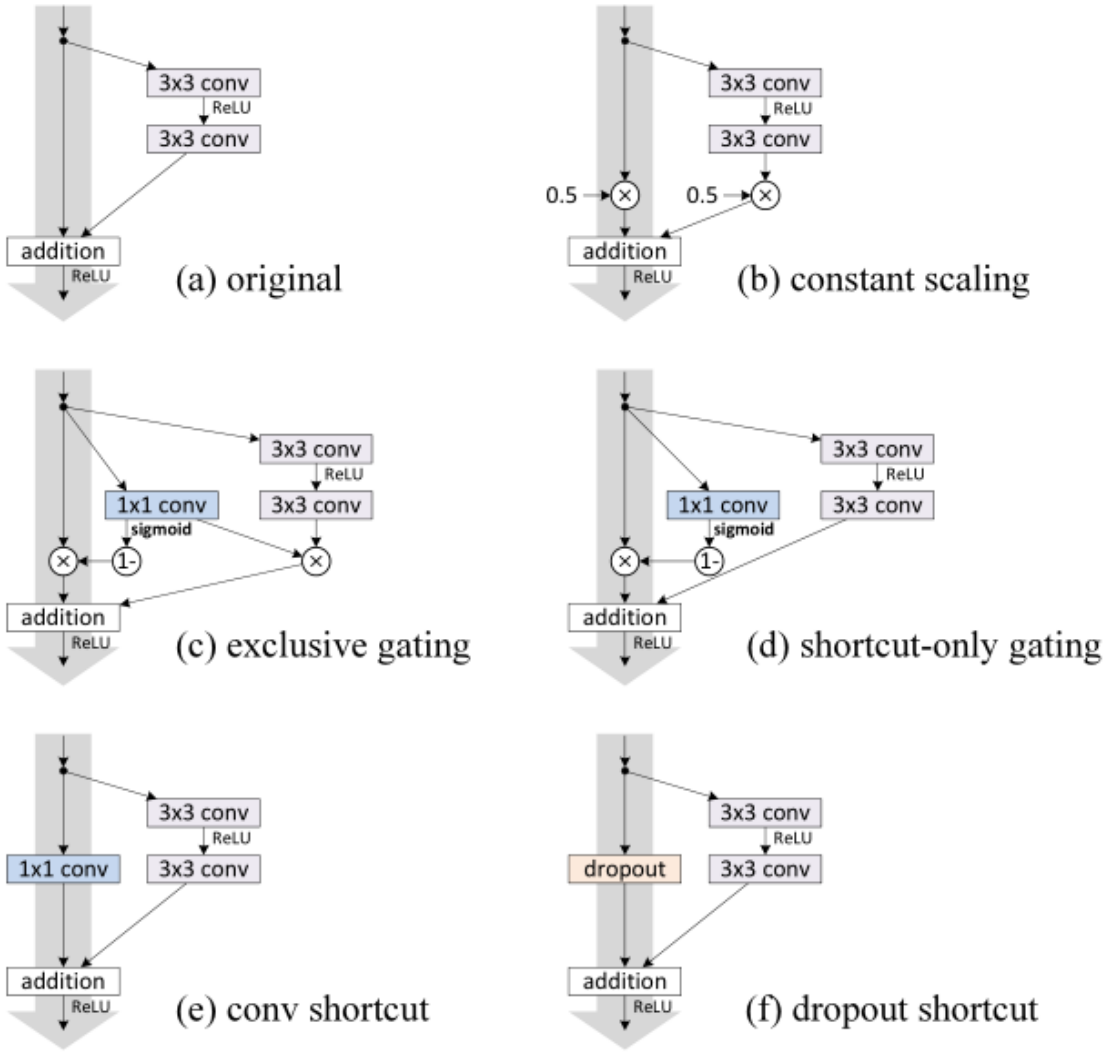


Figure: Various types of shortcut connections. (omitting the BN layers, right after the weight layers for all units here.)

case	Fig.	on shortcut	on $\mathcal{F}$	error (%)	remark
original [1]	Fig. 2(a)	1	1	**6.61**	
constant scaling	Fig. 2(b)	0	1	fail	This is a plain net
		0.5	1	fail	
		0.5	0.5	12.35	frozen gating
exclusive gating	Fig. 2(c)	$1-g(\mathbf{x})$	$g(\mathbf{x})$	fail	init b_g=0 to −5
		$1-g(\mathbf{x})$	$g(\mathbf{x})$	8.70	init b_g=-6
		$1-g(\mathbf{x})$	$g(\mathbf{x})$	9.81	init b_g=-7
shortcut-only gating	Fig. 2(d)	$1-g(\mathbf{x})$	1	12.86	init b_g=0
		$1-g(\mathbf{x})$	1	6.91	init b_g=-6
1×1 conv shortcut	Fig. 2(e)	1×1 conv	1	12.22	
dropout shortcut	Fig. 2(f)	dropout 0.5	1	fail	

Figure: Classification error on the CIFAR-10 test set using ResNet-110, with different types of shortcut connections. We report "fail" when the test error is higher than 20%.

Experiments on Skip Connections

(b) **Constant scaling.** We set $\lambda = 0.5$ for all shortcuts and $\mathcal{F}$ is scalared by $1 - \lambda = 0.5$.

(c) **Exclusive gating.** Consider a gating function $g(x) = \sigma(W_g x + b_g)$, where σ is sigmoid $\sigma(x) = \frac{1}{1+\exp(-x)}$ and $g(x)$ is realized by a 1×1 convolutional layer. The "exclusive" gates : the $\mathcal{F}$ path is scaled by $g(x)$ and the shortcut path is scaled by $1 - g(x)$. We find that *the initialization of the biases b_g is critical for training gated models.*

(d) **Shortcut-only gating.** $\mathcal{F}$ is not scaled. The initialized b_g is more negatively biased, the value of $1 - g(x)$ is closer to 1 and the shortcut connection is nearly an identity mapping.

(e) 1×1 **convolutional shortcut.** We use 1×1 convolutional shortcut connections that replace the identity. This option has showed good results on a 34-layer ResNet (16 Residual Units), but has a poorer result on 110-layer ResNet.

(f) **Dropout shortcut.** Dropout statistically imposes a scale of λ with an expectation of 0.5 on the shortcut, and similar to constant scaling by 0.5, it impedes signal propagation. The network fails to converge to a good solution.

All impede signal propagation!!

Note that the gating and 1×1 convolutional shortcuts introduce more parameters, and should have stronger representational abilities than identity shortcuts. However, their training error is higher than that of identity shortcuts, indicating that the degradation of these models is caused by **optimization issues**, instead of **representational abilities**.

The previous CNN models cannot go deeper, such as VGG which has at most 19-layers. Because the current architectures are not suitable for very deep nets and the bottleneck is training and optimization.
A serious problem is degradation problem, which observed from the experiments of ResNet-18 and ResNet-34 : accuracy gets saturated and then degrades rapidly when networks goes deeper. This talk mainly foucus on why ResNet can make networks go deeper and easily enjoy accuracy gains, and show some analysis results of the advantage of identity shortcuts by comparison with other various types of shortcut connections.
Questions asked by Juncai and Huang huang: The 152-layer ResNet (11.3 billion FLOPs) still has lower complexity than VGG-16/19 nets (15.3/19.6 billion FLOPs) and the reason is that the bottleneck designs leave the 3×3 layer a bottleneck with smaller input and output dimensions so that have less FLOPs.