Identity Mappings in Deep Residual Networks

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun

Microsoft Research

Abstract Deep residual networks [1] have emerged as a family of extremely deep architectures showing compelling accuracy and nice convergence behaviors. In this paper, we analyze the propagation formulations behind the residual building blocks, which suggest that the forward and backward signals can be directly propagated from one block to any other block, when using identity mappings as the skip connections and after-addition activation. A series of ablation experiments support the importance of these identity mappings. This motivates us to propose a new residual unit, which makes training easier and improves generalization. We report improved results using a 1001-layer ResNet on CIFAR-10 (4.62% error) and CIFAR-100, and a 200-layer ResNet on ImageNet. Code is available at: https://github.com/KaimingHe/resnet-1k-layers.

1 Introduction

Deep residual networks (ResNets) [1] consist of many stacked "Residual Units". Each unit (Fig. 1 (a)) can be expressed in a general form:

$$\mathbf{y}_l = h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l),$$
$$\mathbf{x}_{l+1} = f(\mathbf{y}_l),$$

where $\mathbf{x}_l$ and $\mathbf{x}_{l+1}$ are input and output of the l-th unit, and $\mathcal{F}$ is a residual function. In [1], $h(\mathbf{x}_l) = \mathbf{x}_l$ is an identity mapping and f is a ReLU [2] function.

ResNets that are over 100-layer deep have shown state-of-the-art accuracy for several challenging recognition tasks on ImageNet [3] and MS COCO [4] competitions. The central idea of ResNets is to learn the additive residual function $\mathcal{F}$ with respect to $h(\mathbf{x}_l)$, with a key choice of using an identity mapping $h(\mathbf{x}_l) = \mathbf{x}_l$. This is realized by attaching an identity skip connection ("shortcut").

In this paper, we analyze deep residual networks by focusing on creating a "direct" path for propagating information — not only within a residual unit, but through the entire network. Our derivations reveal that *if both* $h(\mathbf{x}_l)$ *and* $f(\mathbf{y}_l)$ *are identity mappings*, the signal could be *directly* propagated from one unit to any other units, in both forward and backward passes. Our experiments empirically show that training in general becomes easier when the architecture is closer to the above two conditions.

To understand the role of skip connections, we analyze and compare various types of $h(\mathbf{x}_l)$. We find that the identity mapping $h(\mathbf{x}_l) = \mathbf{x}_l$ chosen in [1]

Identity Mappings in Deep Residual Networks

何恺明、张翔宇、任少卿和孙剑

微软研究院

Abstract 深度残差网络[1]已成为一类极深架构，展现出卓越的准确性和良好的收敛特性。本文分析了残差构建模块背后的传播公式，结果表明：当使用恒等映射作为跳跃连接并采用加法后激活时，前向与反向信号可直接从一个模块传播至任意其他模块。一系列消融实验证实了这些恒等映射的重要性。这促使我们提出一种新的残差单元，其不仅使训练更易进行，还提升了泛化能力。我们在CIFAR-10（错误率4.62%）和CIFAR-100上使用1001层ResNet，在ImageNet上使用200层ResNet，均取得了改进结果。代码发布于：`https://github.com/KaimingHe/ resnet-1k-layers`。

1 Introduction

D深度残差网络（ResNets）[1] 由许多堆叠的“残差单元”组成。
E每个单元（图1（a））可以用一般形式表示：

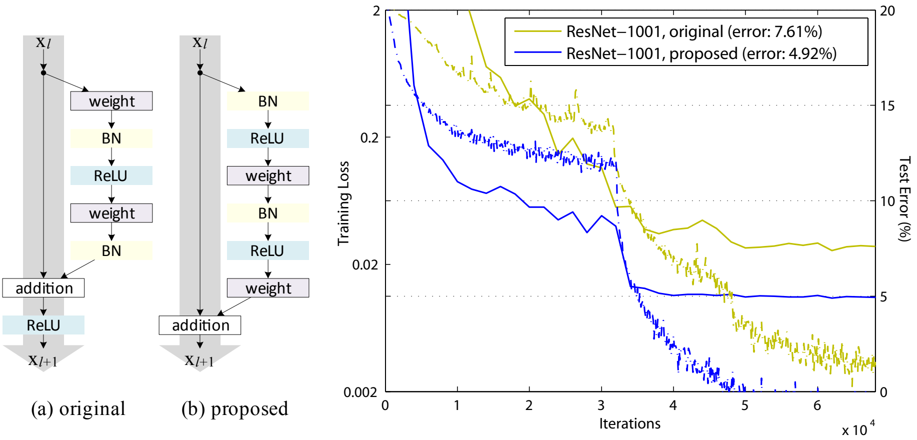
$$\mathbf{y}_l = h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l),$$
$$\mathbf{x}_{l+1} = f(\mathbf{y}_l),$$

其中 $\mathbf{x}_l$ 和 $\mathbf{x}_{l+1}$ 是第 l 个单元的输入和输出，$\mathcal{F}$ 是残差函数。在 [1] 中，$h(\mathbf{x}_l) = \mathbf{x}_l$ 是恒等映射，f 是 ReLU [2] 函数。

超过100层的深度残差网络在ImageNet [3]和MS COCO [4]竞赛的多个挑战性识别任务中展现了最先进的准确率。ResNets的核心思想是学习关于$h(\mathbf{x}_l)$的加性残差函数$\mathcal{F}$，其关键选择是使用恒等映射$h(\mathbf{x}_l) = \mathbf{x}_l$。这一思想通过添加恒等跳跃连接（“捷径”）来实现。

在本文中，我们通过专注于创建信息传播的“直接”路径来分析深度残差网络——不仅在残差单元内部，而且贯穿整个网络。我们的推导表明，*if both* $h(\mathbf{x}_l)$ *and* $f(\mathbf{y}_l)$ *are identity mappings*，信号可以在前向传播和反向传播过程中*directly*从一个单元传播到任何其他单元。我们的实验经验表明，当网络架构更接近上述两个条件时，训练通常变得更容易。

为了理解跳跃连接的作用，我们分析并比较了多种类型的$h(\mathbf{x}_l)$。我们发现，[1]中选用的恒等映射$h(\mathbf{x}_l) = \mathbf{x}_l$



(a) original (b) proposed

Figure 1. Left: (a) original Residual Unit in [1]; (b) proposed Residual Unit. The grey arrows indicate the easiest paths for the information to propagate, corresponding to the additive term "$\mathbf{x}_l$" in Eqn.(4) (forward propagation) and the additive term "1" in Eqn.(5) (backward propagation). **Right**: training curves on CIFAR-10 of **1001-layer** ResNets. Solid lines denote test error (y-axis on the right), and dashed lines denote training loss (y-axis on the left). The proposed unit makes ResNet-1001 easier to train.

achieves the fastest error reduction and lowest training loss among all variants we investigated, whereas skip connections of scaling, gating [5,6,7], and 1×1 convolutions all lead to higher training loss and error. These experiments suggest that keeping a "clean" information path (indicated by the grey arrows in Fig. 1, 2, and 4) is helpful for easing optimization.

To construct an identity mapping $f(\mathbf{y}_l) = \mathbf{y}_l$, we view the activation functions (ReLU and BN [8]) as "*pre-activation*" of the weight layers, in contrast to conventional wisdom of "post-activation". This point of view leads to a new residual unit design, shown in (Fig. 1(b)). Based on this unit, we present competitive results on CIFAR-10/100 with a 1001-layer ResNet, which is much easier to train and generalizes better than the original ResNet in [1]. We further report improved results on ImageNet using a 200-layer ResNet, for which the counterpart of [1] starts to overfit. These results suggest that there is much room to exploit the dimension of *network depth*, a key to the success of modern deep learning.

2 Analysis of Deep Residual Networks

The ResNets developed in [1] are *modularized* architectures that stack building blocks of the same connecting shape. In this paper we call these blocks "*Residual*

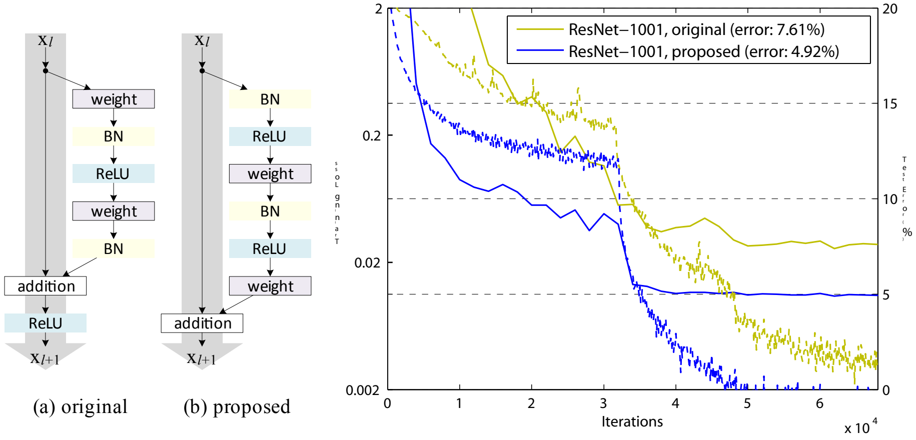


Figure 1. Left(a) 原始残差单元[1]；(b) 所提出的残差单元。灰色箭头表示信息传播的最简易路径，分别对应式(4)中的加性项“$\mathbf{x}_l$”（前向传播）与式(5)中的加性项“1”（反向传播）。**Right**：**1001-layer** ResNets在CIFAR-10上的训练曲线。实线表示测试误差（右侧y轴），虚线表示训练损失（左侧y轴）。所提出的单元使ResNet-1001更易于训练。

在我们研究的所有变体中，它实现了最快的误差降低和最低的训练损失，而缩放、门控[5,6,7]以及1{v*}1卷积的跳跃连接均会导致更高的训练损失和误差。这些实验表明，保持“纯净”的信息路径（由图1、2和4中的灰色箭头指示）有助于优化过程的简化。

为了构建恒等映射$f(\mathbf{y}_l) = \mathbf{y}_l$，我们将激活函数（ReLU和BN[8]）视为权重层的“*pre-activation*”，这与“后激活”的传统观点形成对比。这一视角催生了新的残差单元设计，如图1(b)所示。基于此单元，我们使用1001层ResNet在CIFAR-10/100上取得了具有竞争力的结果，该网络比文献[1]中的原始ResNet更易训练且泛化能力更强。我们进一步报告了使用200层ResNet在ImageNet上获得的改进结果，而文献[1]中的对应模型已开始过拟合。这些结果表明，在挖掘*network depth*维度方面仍有很大空间，而这是现代深度学习成功的关键。

2 Analysis of Deep Residual Networks

[1]中提出的ResNet是一种*modularized*架构，它堆叠了具有相同连接形状的构建模块。在本文中，我们称这些模块为“*Residual*

Units". The original Residual Unit in [1] performs the following computation:

$$\mathbf{y}_l = h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l), \tag{1}$$

$$\mathbf{x}_{l+1} = f(\mathbf{y}_l). \tag{2}$$

Here $\mathbf{x}_l$ is the input feature to the l-th Residual Unit. $\mathcal{W}_l = \{\mathrm{W}_{l,k}|_{1\leq k\leq K}\}$ is a set of weights (and biases) associated with the l-th Residual Unit, and K is the number of layers in a Residual Unit (K is 2 or 3 in [1]). $\mathcal{F}$ denotes the residual function, *e.g.*, a stack of two 3×3 convolutional layers in [1]. The function f is the operation after element-wise addition, and in [1] f is ReLU. The function h is set as an identity mapping: $h(\mathbf{x}_l) = \mathbf{x}_l$.[1]

If f is also an identity mapping: $\mathbf{x}_{l+1} \equiv \mathbf{y}_l$, we can put Eqn.(2) into Eqn.(1) and obtain:

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l). \tag{3}$$

Recursively ($\mathbf{x}_{l+2} = \mathbf{x}_{l+1} + \mathcal{F}(\mathbf{x}_{l+1}, \mathcal{W}_{l+1}) = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l) + \mathcal{F}(\mathbf{x}_{l+1}, \mathcal{W}_{l+1})$, etc.) we will have:

$$\mathbf{x}_L = \mathbf{x}_l + \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i), \tag{4}$$

for *any deeper unit* L and *any shallower unit* l. Eqn.(4) exhibits some nice properties. **(i)** The feature $\mathbf{x}_L$ of any deeper unit L can be represented as the feature $\mathbf{x}_l$ of any shallower unit l plus a residual function in a form of $\sum_{i=l}^{L-1}\mathcal{F}$, indicating that the model is in a *residual* fashion between any units L and l. **(ii)** The feature $\mathbf{x}_L = \mathbf{x}_0 + \sum_{i=0}^{L-1}\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)$, of any deep unit L, is the *summation* of the outputs of all preceding residual functions (plus $\mathbf{x}_0$). This is in contrast to a "plain network" where a feature $\mathbf{x}_L$ is a series of matrix-vector *products*, say, $\prod_{i=0}^{L-1} W_i \mathbf{x}_0$ (ignoring BN and ReLU).

Eqn.(4) also leads to nice backward propagation properties. Denoting the loss function as $\mathcal{E}$, from the chain rule of backpropagation [9] we have:

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\frac{\partial \mathbf{x}_L}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left(1 + \frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)\right). \tag{5}$$

Eqn.(5) indicates that the gradient $\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l}$ can be decomposed into two additive terms: a term of $\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}$ that propagates information directly without concerning any weight layers, and another term of $\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left(\frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}\right)$ that propagates through the weight layers. The additive term of $\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}$ ensures that information is directly propagated back to *any shallower unit* l. Eqn.(5) also suggests that it

[1] It is noteworthy that there are Residual Units for increasing dimensions and reducing feature map sizes [1] in which h is not identity. In this case the following derivations do not hold strictly. But as there are only a very few such units (two on CIFAR and three on ImageNet, depending on image sizes [1]), we expect that they do not have the exponential impact as we present in Sec. 3. One may also think of our derivations as applied to all Residual Units within the same feature map size.

Units”[1]中的原始残差单元执行以下计算:

$$\mathbf{y}_l = h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l), \tag{1}$$

$$\mathbf{x}_{l+1} = f(\mathbf{y}_l). \tag{2}$$

这里 $\mathbf{x}_l$ 是第 l 个残差单元的输入特征。$\mathcal{W}_l = \{\mathrm{W}_{l,k}|_{1\le k\le K}\}$ 是与第 l 个残差单元相关联的一组权重（和偏置），而 K 是残差单元中的层数（在[1]中 K 为 2 或 3）。$\mathcal{F}$ 表示残差函数，在[1]中 *e.g.* 是一个由两个 3×3 卷积层组成的堆叠。函数 f 是逐元素相加后的操作，在[1]中 f 是 ReLU。函数 h 被设置为恒等映射：$h(\mathbf{x}_l) = \mathbf{x}_l$.[1]

如果f也是一个恒等映射：$\mathbf{x}_{l+1} \equiv \mathbf{y}_l$，我们可以将公式(2)代入公式(1)并得到：

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l). \tag{3}$$

递归地（$\mathbf{x}_{l+2} = \mathbf{x}_{l+1} + \mathcal{F}(\mathbf{x}_{l+1}, \mathcal{W}_{l+1}) = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l) + \mathcal{F}(\mathbf{x}_{l+1}, \mathcal{W}_{l+1})$等）我们将得到:

$$\mathbf{x}_L = \mathbf{x}_l + \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i), \tag{4}$$

对于*any deeper unit L*和*any shallower unit l*，公式(4)展现出一些优良特性。**(i)** 任意深层单元L的特征$\mathbf{x}_L$，均可表示为任意浅层单元l的特征$\mathbf{x}_l$加上形如$\sum_{i=l}^{L-1}\mathcal{F}$的残差函数，这表明模型在任意单元$L$和$l$之间呈*residual*方式运作。**(ii)** 任意深层单元L的特征$\mathbf{x}_L = \mathbf{x}_0 + \sum_{i=0}^{L-1}\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)$，是所有前序残差函数输出（加上$\mathbf{x}_0$）的*summation*。这与“普通网络”形成对比——在普通网络中，特征$\mathbf{x}_L$是一系列矩阵-向量*products*的运算，例如$\prod^{L-1}$。

$_{i=0} W_i \mathbf{x}_0$（忽略BN和ReLU）。

方程（4）也带来了良好的反向传播特性。将损失函数记为$\mathcal{E}$，根据反向传播的链式法则[9]，我们可得:

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\frac{\partial \mathbf{x}_L}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left(1 + \frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)\right). \tag{5}$$

公式(5)表明梯度$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l}$可分解为两个相加项：一项是$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}$，它直接传播信息而不涉及任何权重层；另一项是$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left(\frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}\right)$，它通过权重层传播。$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}$的相加项确保信息能直接传播回*any shallower unit l*。公式(5)还表明它

[1] It is noteworthy that there are Residual Units for increasing dimensions and reducing feature map sizes [1] in which h is not identity. In this case the following derivations do not hold strictly. But as there are only a very few such units (two on CIFAR and three on ImageNet, depending on image sizes [1]), we expect that they do not have the exponential impact as we present in Sec. 3. One may also think of our derivations as applied to all Residual Units within the same feature map size.

is unlikely for the gradient $\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l}$ to be canceled out for a mini-batch, because in general the term $\frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}$ cannot be always -1 for all samples in a mini-batch. This implies that the gradient of a layer does not vanish even when the weights are arbitrarily small.

Discussions

Eqn.(4) and Eqn.(5) suggest that the signal can be directly propagated from any unit to another, both forward and backward. The foundation of Eqn.(4) is two identity mappings: (i) the identity skip connection $h(\mathbf{x}_l) = \mathbf{x}_l$, and (ii) the condition that f is an identity mapping.

These directly propagated information flows are represented by the grey arrows in Fig. 1, 2, and 4. And the above two conditions are true when these grey arrows cover no operations (expect addition) and thus are "clean". In the following two sections we separately investigate the impacts of the two conditions.

3 On the Importance of Identity Skip Connections

Let's consider a simple modification, $h(\mathbf{x}_l) = \lambda_l \mathbf{x}_l$, to break the identity shortcut:

$$\mathbf{x}_{l+1} = \lambda_l \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l), \tag{6}$$

where λ_l is a modulating scalar (for simplicity we still assume f is identity). Recursively applying this formulation we obtain an equation similar to Eqn. (4): $\mathbf{x}_L = (\prod_{i=l}^{L-1}\lambda_i)\mathbf{x}_l + \sum_{i=l}^{L-1}(\prod_{j=i+1}^{L-1}\lambda_j)\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)$, or simply:

$$\mathbf{x}_L = (\prod_{i=l}^{L-1}\lambda_i)\mathbf{x}_l + \sum_{i=l}^{L-1}\hat{\mathcal{F}}(\mathbf{x}_i, \mathcal{W}_i), \tag{7}$$

where the notation $\hat{\mathcal{F}}$ absorbs the scalars into the residual functions. Similar to Eqn.(5), we have backpropagation of the following form:

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left((\prod_{i=l}^{L-1}\lambda_i) + \frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\hat{\mathcal{F}}(\mathbf{x}_i, \mathcal{W}_i)\right). \tag{8}$$

Unlike Eqn.(5), in Eqn.(8) the first additive term is modulated by a factor $\prod_{i=l}^{L-1}\lambda_i$. For an extremely deep network (L is large), if $\lambda_i > 1$ for all i, this factor can be exponentially large; if $\lambda_i < 1$ for all i, this factor can be exponentially small and vanish, which blocks the backpropagated signal from the shortcut and forces it to flow through the weight layers. This results in optimization difficulties as we show by experiments.

In the above analysis, the original identity skip connection in Eqn.(3) is replaced with a simple scaling $h(\mathbf{x}_l) = \lambda_l \mathbf{x}_l$. If the skip connection $h(\mathbf{x}_l)$ represents more complicated transforms (such as gating and 1×1 convolutions), in Eqn.(8) the first term becomes $\prod_{i=l}^{L-1} h'_i$ where h' is the derivative of h. This product may also impede information propagation and hamper the training procedure as witnessed in the following experiments.

对于一个小批量来说，梯度$\frac{\partial\mathcal{E}}{\partial\mathbf{x}_l}$不太可能被抵消，因为通常来说，对于一个小批量中的所有样本，项$\frac{\partial}{\partial\mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}$不可能总是-1。这意味着即使权重任意小，某一层的梯度也不会消失。

Discussions

式(4)和式(5)表明信号可以直接在任何单元之间前向和后向传播。式(4)的基础是两个恒等映射：(i) 恒等跳跃连接 $h(\mathbf{x}_l) = \mathbf{x}_l$，以及 (ii) f 是恒等映射的条件。

这些直接传播的信息流由图1、2和4中的灰色箭头表示。当这些灰色箭头不覆盖任何操作（加法除外）且因此保持“纯净”时，上述两个条件成立。在接下来的两节中，我们将分别探讨这两个条件的影响。

3 On the Importance of Identity Skip Connections

让我们考虑一个简单的修改，$h(\mathbf{x}_l) = \lambda_l\mathbf{x}_l$，以打破恒等快捷方式：

$$\mathbf{x}_{l+1} = \lambda_l\mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l), \tag{6}$$

其中λ_l是一个调制标量（为简化起见，我们仍假设f是单位矩阵）。递归应用此公式，我们得到一个类似于公式(4)的方程：
$\mathbf{x}_L = (\prod_{i=l}^{L-1}\lambda_i)\mathbf{x}_l + \sum_{i=l}^{L-1}(\prod_{j=i+1}^{L-1}\lambda_j)\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)$，或简写为：

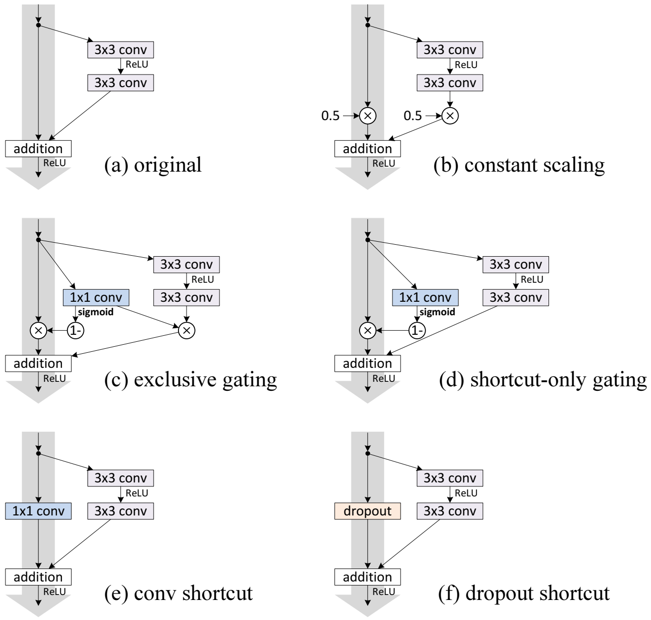
$$\mathbf{x}_L = (\prod_{i=l}^{L-1}\lambda_i)\mathbf{x}_l + \sum_{i=l}^{L-1}\hat{\mathcal{F}}(\mathbf{x}_i, \mathcal{W}_i), \tag{7}$$

其中，符号$\hat{\mathcal{F}}$将标量吸收到残差函数中。类似于方程(5)，我们得到以下形式的反向传播：

$$\frac{\partial\mathcal{E}}{\partial\mathbf{x}_l} = \frac{\partial\mathcal{E}}{\partial\mathbf{x}_L}\left((\prod_{i=l}^{L-1}\lambda_i) + \frac{\partial}{\partial\mathbf{x}_l}\sum_{i=l}^{L-1}\hat{\mathcal{F}}(\mathbf{x}_i, \mathcal{W}_i)\right). \tag{8}$$

与公式(5)不同，在公式(8)中，第一个加法项受到因子$\prod_{i=l}^{L-1}\lambda_i$的调制。对于一个极深的网络（$L$较大），如果所有$i$对应的$\lambda_i >$都大于1，该因子可能呈指数级增大；如果所有$i$对应的$\lambda_i <$都小于1，该因子可能呈指数级减小直至消失，这会阻断来自捷径的反向传播信号，迫使信号流经权重层。正如我们通过实验所展示的，这将导致优化困难。

在上述分析中，式(3)中的原始恒等跳跃连接被替换为简单的缩放
$h(\mathbf{x}_l) = \lambda_l\mathbf{x}_l$。若跳跃连接$h(\mathbf{x}_l)$代表更复杂的变换（例如门控和1×1卷积），则式(8)中的第一项变为$\prod_{i=l}^{L-1}h'_i$，其中h'是h的导数。如后续实验所示，该乘积同样可能阻碍信息传播并干扰训练过程。



(a) original

(b) constant scaling

(c) exclusive gating

(d) shortcut-only gating

(e) conv shortcut

(f) dropout shortcut

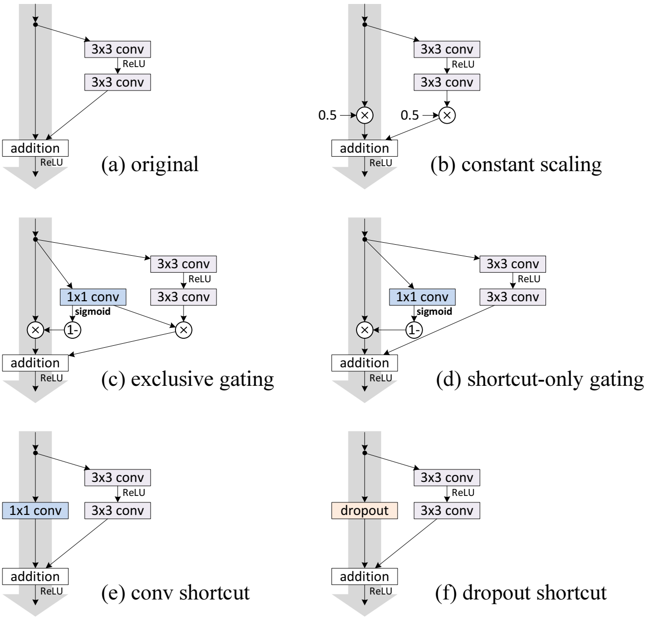
Figure 2. Various types of shortcut connections used in Table 1. The grey arrows indicate the easiest paths for the information to propagate. The shortcut connections in (b-f) are impeded by different components. For simplifying illustrations we do not display the BN layers, which are adopted right after the weight layers for all units here.

3.1 Experiments on Skip Connections

We experiment with the 110-layer ResNet as presented in [1] on CIFAR-10 [10]. This extremely deep ResNet-110 has 54 two-layer Residual Units (consisting of 3×3 convolutional layers) and is challenging for optimization. Our implementation details (see appendix) are the same as [1]. Throughout this paper we report the median accuracy of **5 runs** for each architecture on CIFAR, reducing the impacts of random variations.

Though our above analysis is driven by identity f, the experiments in this section are all based on $f = \text{ReLU}$ as in [1]; we address identity f in the next section. Our baseline ResNet-110 has 6.61% error on the test set. The comparisons of other variants (Fig. 2 and Table 1) are summarized as follows:

Constant scaling. We set $\lambda = 0.5$ for all shortcuts (Fig. 2(b)). We further study two cases of scaling $\mathcal{F}$: (i) $\mathcal{F}$ is not scaled; or (ii) $\mathcal{F}$ is scaled by a constant scalar of $1 - \lambda = 0.5$, which is similar to the highway gating [6,7] but with frozen gates. The former case does not converge well; the latter is able to converge, but the test error (Table 1, 12.35%) is substantially higher than the original ResNet-110. Fig 3(a) shows that the training error is higher than that of the original ResNet-110, suggesting that the optimization has difficulties when the shortcut signal is scaled down.



(a) original (b) constant scaling

(c) exclusive gating (d) shortcut-only gating

(e) conv shortcut (f) dropout shortcut

Figure 2. 表1中使用的各种类型的快捷连接。灰色箭头表示信息传播的最简单路径。(b-f)中的快捷连接受到不同组件的阻碍。为简化图示，我们未显示BN层，此处所有单元在权重层之后均采用了BN层。

3.1 Experiments on Skip Connections

我们在CIFAR-10数据集上对文献[1]提出的110层ResNet进行了实验。这种极深的ResNet-110包含54个双层残差单元（由3×3卷积层构成），对优化过程具有挑战性。我们的实现细节（见附录）与文献[1]保持一致。本文中所有CIFAR架构的准确率均采用**5 runs**次运行的中位数进行报告，以减少随机波动的影响。

尽管我们上述分析基于恒等映射f，但本节实验均采用如[1]中所述的$f =$ ReLU；我们将在下一节探讨恒等映射f。我们的基线ResNet-110在测试集上的错误率为6.61%。其他变体的比较（图2和表1）总结如下：

Constant scaling我们将所有快捷连接（图2(b)）的$\lambda =$设为0.5。我们进一步研究了两种缩放$\mathcal{F}$的情况：（i）$\mathcal{F}$不进行缩放；或（ii）$\mathcal{F}$按$1-\lambda =$0.5的常数标量进行缩放，这与高速公路门控[6,7]类似，但门控是冻结的。前一种情况收敛效果不佳；后一种情况能够收敛，但测试误差（表1，12.35%）明显高于原始ResNet-110。图3(a)显示其训练误差高于原始ResNet-110，表明当快捷信号被缩小时，优化过程会遇到困难。

Table 1. Classification error on the CIFAR-10 test set using ResNet-110 [1], with different types of shortcut connections applied to all Residual Units. We report "fail" when the test error is higher than 20%.

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	

Exclusive gating. Following the Highway Networks [6,7] that adopt a gating mechanism [5], we consider a gating function $g(\mathbf{x}) = \sigma(\mathrm{W}_g\mathbf{x} + b_g)$ where a transform is represented by weights W_g and biases b_g followed by the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. In a convolutional network $g(\mathbf{x})$ is realized by a 1×1 convolutional layer. The gating function modulates the signal by element-wise multiplication.

We investigate the "exclusive" gates as used in [6,7] — the $\mathcal{F}$ path is scaled by $g(\mathbf{x})$ and the shortcut path is scaled by $1-g(\mathbf{x})$. See Fig 2(c). We find that the initialization of the biases b_g is critical for training gated models, and following the guidelines[2] in [6,7], we conduct hyper-parameter search on the initial value of b_g in the range of 0 to -10 with a decrement step of -1 on the training set by cross-validation. The best value (-6 here) is then used for training on the training set, leading to a test result of 8.70% (Table 1), which still lags far behind the ResNet-110 baseline. Fig 3(b) shows the training curves. Table 1 also reports the results of using other initialized values, noting that the exclusive gating network does not converge to a good solution when b_g is not appropriately initialized.

The impact of the exclusive gating mechanism is two-fold. When $1 - g(\mathbf{x})$ approaches 1, the gated shortcut connections are closer to identity which helps information propagation; but in this case $g(\mathbf{x})$ approaches 0 and suppresses the function $\mathcal{F}$. To isolate the effects of the gating functions on the shortcut path alone, we investigate a non-exclusive gating mechanism in the next.

Shortcut-only gating. In this case the function $\mathcal{F}$ is not scaled; only the shortcut path is gated by $1-g(\mathbf{x})$. See Fig 2(d). The initialized value of b_g is still essential in this case. When the initialized b_g is 0 (so initially the expectation of $1 - g(\mathbf{x})$ is 0.5), the network converges to a poor result of 12.86% (Table 1). This is also caused by higher training error (Fig 3(c)).

[2] See also: `people.idsia.ch/~rupesh/very_deep_learning/` by [6,7].

Table 1. 在CIFAR-10测试集上使用ResNet-110[1]的分类错误率，其中所有残差单元均应用了不同类型的快捷连接。当测试错误率高于20%时，我们标记为“失败”。

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	

Exclusive gating继采用门控机制[5]的高速网络[6,7]之后，我们考虑一个门控函数$g(\mathbf{x}) = \sigma(\mathrm{W}_g\mathbf{x} + b_g)$，其中变换由权重$\mathrm{W}_g$和偏置$b_g$表示，后接sigmoid函数$\sigma(x) = \frac{1}{1+e^{-x}}$。在卷积网络中，$g(\mathbf{x})$通过1×1卷积层实现。该门控函数通过逐元素乘法调制信号。

我们研究了[6,7]中使用的“独占”门——$\mathcal{F}$路径由$g(\mathbf{x})$缩放，而捷径路径由$1-g(\mathbf{x})$缩放。参见图2(c)。我们发现偏置项b_g的初始化对于训练门控模型至关重要，依据[6,7]中的指导原则[2]，我们通过交叉验证在训练集上对b_g的初始值在0到-10范围内以-1为递减步长进行超参数搜索。最佳值（此处为-6）随后被用于训练集上的训练，得到8.70%的测试结果（表1），但仍远落后于ResNet-110基线。图3(b)展示了训练曲线。表1还报告了使用其他初始化值的结果，值得注意的是当b_g未适当初始化时，独占门控网络无法收敛到良好解。

排他性门控机制的影响是双重的。当$1-g(\mathbf{x})$趋近于1时，门控捷径连接更接近恒等映射，这有助于信息传播；但此时$g(\mathbf{x})$趋近于0并抑制了函数$\mathcal{F}$的作用。为了单独研究门控函数在捷径路径上的影响，我们将在下一节探讨非排他性门控机制。

Shortcut-only gating在这种情况下，函数$\mathcal{F}$未被缩放；仅快捷路径被$1-g(\mathbf{x})$门控。参见图2(d)。此时b_g的初始化值仍然至关重要。当初始化的b_g为0时（因此初始时期望值$1-g(\mathbf{x})$为0.5），网络会收敛至12.86%的较差结果（表1）。这也由较高的训练误差导致（图3(c)）。

[2] See also: `people.idsia.ch/~rupesh/very_deep_learning/` by [6,7].

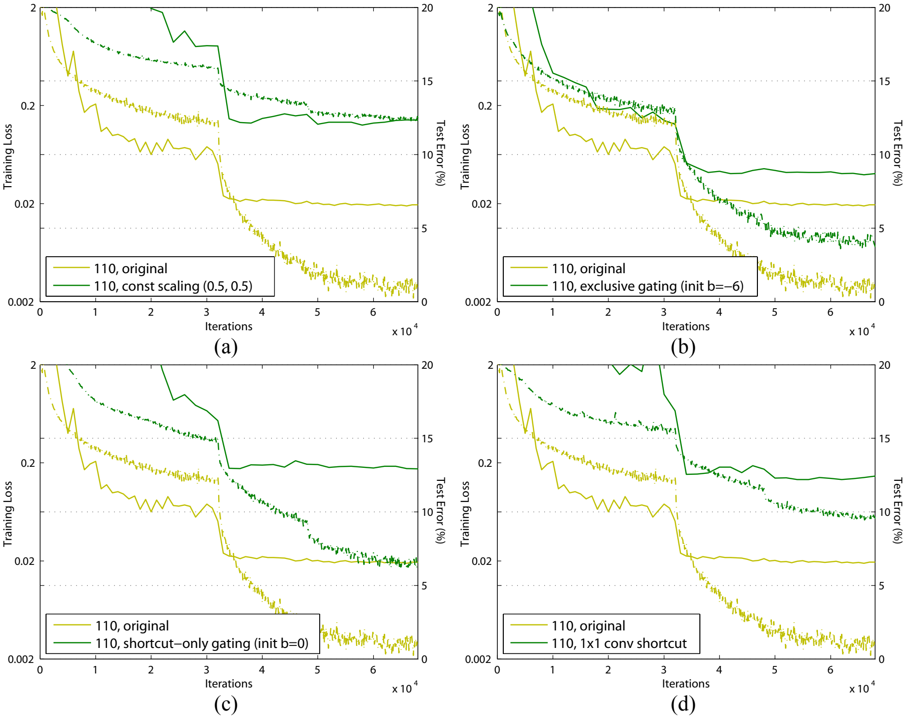
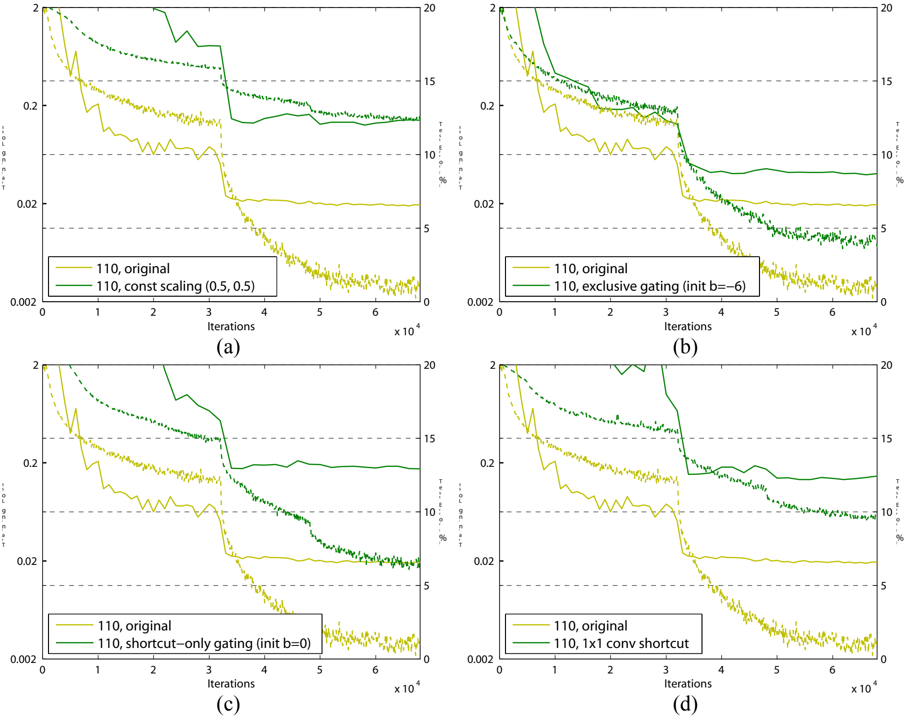


Figure 3. Training curves on CIFAR-10 of various shortcuts. Solid lines denote test error (y-axis on the right), and dashed lines denote training loss (y-axis on the left).

When the initialized b_g is very negatively biased (*e.g.*, -6), the value of $1-g(\mathbf{x})$ is closer to 1 and the shortcut connection is nearly an identity mapping. Therefore, the result (6.91%, Table 1) is much closer to the ResNet-110 baseline.

1×1 convolutional shortcut. Next we experiment with 1×1 convolutional shortcut connections that replace the identity. This option has been investigated in [1] (known as option C) on a 34-layer ResNet (16 Residual Units) and shows good results, suggesting that 1×1 shortcut connections could be useful. But we find that this is not the case when there are many Residual Units. The 110-layer ResNet has a poorer result (12.22%, Table 1) when using 1×1 convolutional shortcuts. Again, the training error becomes higher (Fig 3(d)). When stacking so many Residual Units (54 for ResNet-110), even the shortest path may still impede signal propagation. We witnessed similar phenomena on ImageNet with ResNet-101 when using 1×1 convolutional shortcuts.

Dropout shortcut. Last we experiment with dropout [11] (at a ratio of 0.5) which we adopt on the output of the identity shortcut (Fig. 2(f)). The network fails to converge to a good solution. 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.



(a) (b) (c) (d)

Figure 3. 多种捷径在CIFAR-10数据集上的训练曲线。实线表示测试误差（右侧y轴），虚线表示训练损失（左侧y轴）。

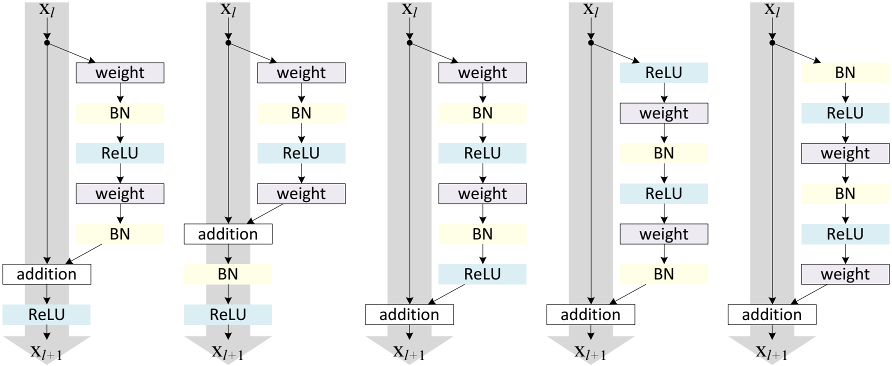
当初始化的b_g具有很大的负偏置($e.g.$, -6)时，$1-g(\mathbf{x})$的值更接近1，此时捷径连接几乎是一个恒等映射。因此，其结果(6.91%，表1)非常接近ResNet-110的基线。

1×1 convolutional shortcut接下来，我们尝试使用1×1卷积捷径连接来替代恒等映射。这一方案已在[1]（称为选项C）中基于34层ResNet（16个残差单元）进行过研究，并显示出良好效果，表明1×1捷径连接可能具有实用价值。但我们发现，当残差单元数量较多时，情况并非如此。在使用1×1卷积捷径时，110层ResNet的结果更差（12.22%，表1）。同样，训练误差也变得更高（图3(d)）。当堆叠如此多的残差单元（ResNet-110为54个）时，即使是最短路径仍可能阻碍信号传播。我们在ImageNet上使用ResNet-101并采用1×1卷积捷径时，也观察到了类似现象。

Dropout shortcut最后，我们尝试了在恒等捷径的输出上采用丢弃法[11]（比例为0.5）（图2(f)）。网络未能收敛到一个良好的解。丢弃法在统计上对捷径施加了一个期望值为0.5的λ尺度，类似于常数缩放0.5，它阻碍了信号的传播。

Table 2. Classification error (%) on the CIFAR-10 test set using different activation functions.

case	Fig.	ResNet-110	ResNet-164
original Residual Unit [1]	Fig. 4(a)	6.61	5.93
BN after addition	Fig. 4(b)	8.17	6.50
ReLU before addition	Fig. 4(c)	7.84	6.14
ReLU-only pre-activation	Fig. 4(d)	6.71	5.91
full pre-activation	Fig. 4(e)	**6.37**	**5.46**



(a) original

(b) BN after addition

(c) ReLU before addition

(d) ReLU-only pre-activation

(e) **full pre-activation**

Figure 4. Various usages of activation in Table 2. All these units consist of the same components — only the orders are different.

3.2 Discussions

As indicated by the grey arrows in Fig. 2, the shortcut connections are the most direct paths for the information to propagate. *Multiplicative* manipulations (scaling, gating, 1×1 convolutions, and dropout) on the shortcuts can hamper information propagation and lead to optimization problems.

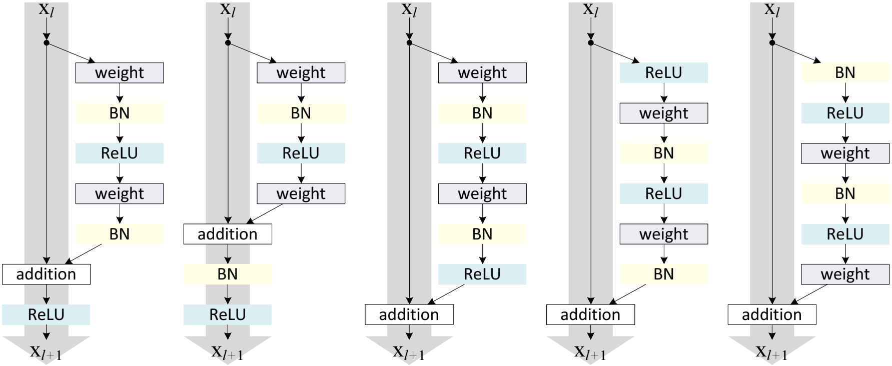
It is noteworthy that the gating and 1×1 convolutional shortcuts introduce more parameters, and should have stronger *representational* abilities than identity shortcuts. In fact, the shortcut-only gating and 1×1 convolution cover the solution space of identity shortcuts (*i.e.*, they could be optimized as 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.

4 On the Usage of Activation Functions

Experiments in the above section support the analysis in Eqn.(5) and Eqn.(8), both being derived under the assumption that the after-addition activation f

Table 2. 在CIFAR-10测试集上使用不同激活函数的分类错误率（%）。

case	Fig.	ResNet-110	ResNet-164
original Residual Unit [1]	Fig. 4(a)	6.61	5.93
BN after addition	Fig. 4(b)	8.17	6.50
ReLU before addition	Fig. 4(c)	7.84	6.14
ReLU-only pre-activation	Fig. 4(d)	6.71	5.91
full pre-activation	Fig. 4(e)	**6.37**	**5.46**



(a) original

(b) BN after addition

(c) ReLU before addition

(d) ReLU-only pre-activation

(e) **full pre-activation**

Figure 4. 表2中激活功能的各种用法。所有这些单元都由相同的组件构成——仅仅是顺序不同。

3.2 Discussions

如图2中的灰色箭头所示，快捷连接是信息传播的最直接路径。对快捷连接进行*Multiplicative*操作（缩放、门控、1×1卷积和丢弃）可能阻碍信息传播并导致优化问题。

值得注意的是，门控和1×1卷积捷径引入了更多参数，应比恒等捷径具备更强的*representational*能力。实际上，仅含捷径的门控和1×1卷积覆盖了恒等捷径的解空间（*i.e.*，它们可被优化为恒等捷径）。然而，其训练误差高于恒等捷径，表明这些模型的性能退化是由优化问题而非表征能力所导致。

4 On the Usage of Activation Functions

上一节的实验支持了公式(5)和公式(8)的分析，这两者都是在假设加法后激活值*f*的前提下推导得出的。

is the identity mapping. But in the above experiments f is ReLU as designed in [1], so Eqn.(5) and (8) are approximate in the above experiments. Next we investigate the impact of f.

We want to make f an identity mapping, which is done by re-arranging the activation functions (ReLU and/or BN). The original Residual Unit in [1] has a shape in Fig. 4(a) — BN is used after each weight layer, and ReLU is adopted after BN except that the last ReLU in a Residual Unit is after element-wise addition (f = ReLU). Fig. 4(b-e) show the alternatives we investigated, explained as following.

4.1 Experiments on Activation

In this section we experiment with ResNet-110 and a 164-layer *Bottleneck* [1] architecture (denoted as ResNet-164). A bottleneck Residual Unit consist of a 1×1 layer for reducing dimension, a 3×3 layer, and a 1×1 layer for restoring dimension. As designed in [1], its computational complexity is similar to the two-3×3 Residual Unit. More details are in the appendix. The baseline ResNet-164 has a competitive result of 5.93% on CIFAR-10 (Table 2).

BN after addition. Before turning f into an identity mapping, we go the opposite way by adopting BN after addition (Fig. 4(b)). In this case f involves BN and ReLU. The results become considerably worse than the baseline (Table 2). Unlike the original design, now the BN layer alters the signal that passes through the shortcut and impedes information propagation, as reflected by the difficulties on reducing training loss at the beginning of training (Fib. 6 left).

ReLU before addition. A naïve choice of making f into an identity mapping is to move the ReLU before addition (Fig. 4(c)). However, this leads to a *non-negative* output from the transform $\mathcal{F}$, while intuitively a "residual" function should take values in $(-\infty, +\infty)$. As a result, the forward propagated signal is monotonically increasing. This may impact the representational ability, and the result is worse (7.84%, Table 2) than the baseline. We expect to have a residual function taking values in $(-\infty, +\infty)$. This condition is satisfied by other Residual Units including the following ones.

Post-activation or pre-activation? In the original design (Eqn.(1) and Eqn.(2)), the activation $\mathbf{x}_{l+1} = f(\mathbf{y}_l)$ affects *both paths* in the *next* Residual Unit: $\mathbf{y}_{l+1} = f(\mathbf{y}_l) + \mathcal{F}(f(\mathbf{y}_l), \mathcal{W}_{l+1})$. Next we develop an *asymmetric* form where an activation $\hat{f}$ only affects the $\mathcal{F}$ path: $\mathbf{y}_{l+1} = \mathbf{y}_l + \mathcal{F}(\hat{f}(\mathbf{y}_l), \mathcal{W}_{l+1})$, for any l (Fig. 5 (a) to (b)). By renaming the notations, we have the following form:

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\hat{f}(\mathbf{x}_l), \mathcal{W}_l), . \tag{9}$$

It is easy to see that Eqn.(9) is similar to Eqn.(4), and can enable a backward formulation similar to Eqn.(5). For this new Residual Unit as in Eqn.(9), the new after-addition activation becomes an identity mapping. This design means that if a new after-addition activation $\hat{f}$ is asymmetrically adopted, it is equivalent to recasting $\hat{f}$ as the *pre-activation* of the next Residual Unit. This is illustrated in Fig. 5.

是恒等映射。但在上述实验中，f按照[1]的设计采用ReLU，因此公式(5)和(8)在上述实验中为近似结果。接下来我们将探究f的影响。

我们希望将f变为恒等映射，这通过重新排列激活函数（ReLU和/或BN）来实现。原始残差单元[1]的结构如图4(a)所示——每个权重层后都使用BN，且BN后采用ReLU，但残差单元中最后一个ReLU位于逐元素相加之后（f = ReLU）。图4(b-e)展示了我们研究的替代方案，解释如下。

4.1 Experiments on Activation

在本节中，我们使用ResNet-110和一个164层的$Bottleneck$架构（称为ResNet-164）进行实验。一个瓶颈残差单元包含一个用于降维的1×1层、一个3×3层和一个用于恢复维度的1×1层。按照[1]的设计，其计算复杂度与两个3×3残差单元相近。更多细节见附录。基线ResNet-164在CIFAR-10上取得了5.93%的竞争性结果（表2）。

BN after addition在将f转变为恒等映射之前，我们采取了相反的做法，即在加法后采用BN（图4(b)）。在这种情况下，f包含了BN和ReLU。其结果明显差于基线（表2）。与原始设计不同，此时BN层改变了通过捷径的信号，阻碍了信息传播，这反映在训练初期降低训练损失的困难上（图6左）。

ReLU before addition一种将f设为恒等映射的简单方法是将ReLU移到加法之前（图4(c)）。然而，这会导致变换$\mathcal{F}$的输出为$non\text{-}negative$，而直观上“残差”函数应在$(-\infty, +\infty)$范围内取值。因此，前向传播的信号会单调递增。这可能影响模型的表达能力，导致结果（7.84%，表2）比基准更差。我们希望残差函数能在$(-\infty, +\infty)$范围内取值。其他残差单元（包括后续单元）均满足这一条件。

Post-activation or pre-activation? 在原始设计（公式(1)和公式(2)）中，激活$\mathbf{x}_{l+1} = f(\mathbf{y}_l)$会影响$next$残差单元中的$both\ paths$: $\mathbf{y}_{l+1} = f(\mathbf{y}_l) + \mathcal{F}(f(\mathbf{y}_l), \mathcal{W}_{l+1})$。接下来我们推导一种$asymmetric$形式，其中激活$\hat{f}$仅影响$\mathcal{F}$路径：$\mathbf{y}_{l+1} = \mathbf{y}_l + \mathcal{F}(\hat{f}(\mathbf{y}_l), \mathcal{W}_{l+1})$，适用于任意$l$(（图5从(a))到(b)）。通过重命名符号，我们得到以下形式：

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\hat{f}(\mathbf{x}_l), \mathcal{W}_l), . \tag{9}$$

很容易看出，式(9)与式(4)相似，并且能够实现类似于式(5)的反向公式。对于如式(9)所示的新残差单元，新的加法后激活变为恒等映射。这一设计意味着，如果非对称地采用新的加法后激活$\hat{f}$，则相当于将$\hat{f}$重新定义为下一个残差单元的$pre\text{-}activation$。如图5所示。

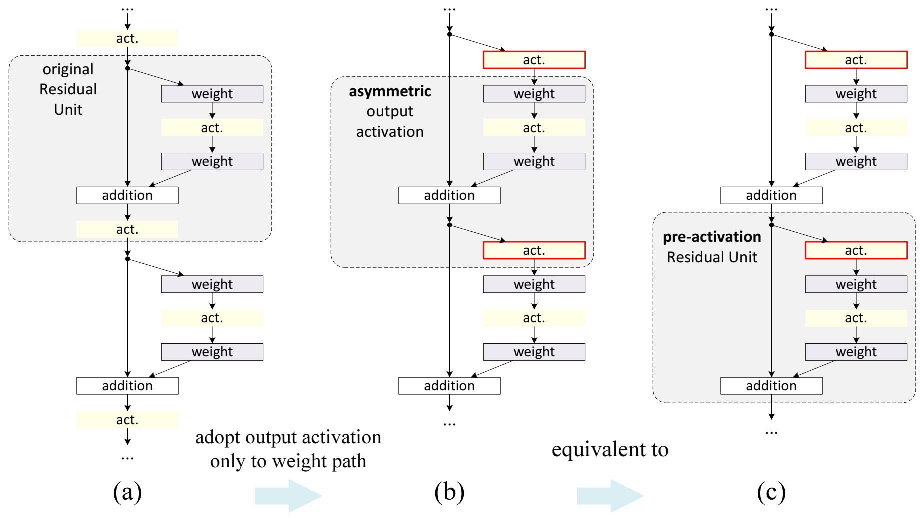


Figure 5. Using asymmetric after-addition activation is equivalent to constructing a *pre-activation* Residual Unit.

Table 3. Classification error (%) on the CIFAR-10/100 test set using the original Residual Units and our pre-activation Residual Units.

dataset	network	baseline unit	pre-activation unit
CIFAR-10	ResNet-110 (1layer skip)	9.90	8.91
	ResNet-110	6.61	6.37
	ResNet-164	5.93	5.46
	ResNet-1001	7.61	4.92
CIFAR-100	ResNet-164	25.16	24.33
	ResNet-1001	27.82	22.71

The distinction between post-activation/pre-activation is caused by the presence of the element-wise *addition*. For a plain network that has N layers, there are $N - 1$ activations (BN/ReLU), and it does not matter whether we think of them as post- or pre-activations. But for branched layers merged by addition, the position of activation matters.

We experiment with two such designs: (i) ReLU-only pre-activation (Fig. 4(d)), and (ii) full pre-activation (Fig. 4(e)) where BN and ReLU are both adopted before weight layers. Table 2 shows that the ReLU-only pre-activation performs very similar to the baseline on ResNet-110/164. This ReLU layer is not used in conjunction with a BN layer, and may not enjoy the benefits of BN [8].

Somehow surprisingly, when BN and ReLU are both used as pre-activation, the results are improved by healthy margins (Table 2 and Table 3). In Table 3 we report results using various architectures: (i) ResNet-110, (ii) ResNet-164, (iii) a 110-layer ResNet architecture in which each shortcut skips only 1 layer (*i.e.*,

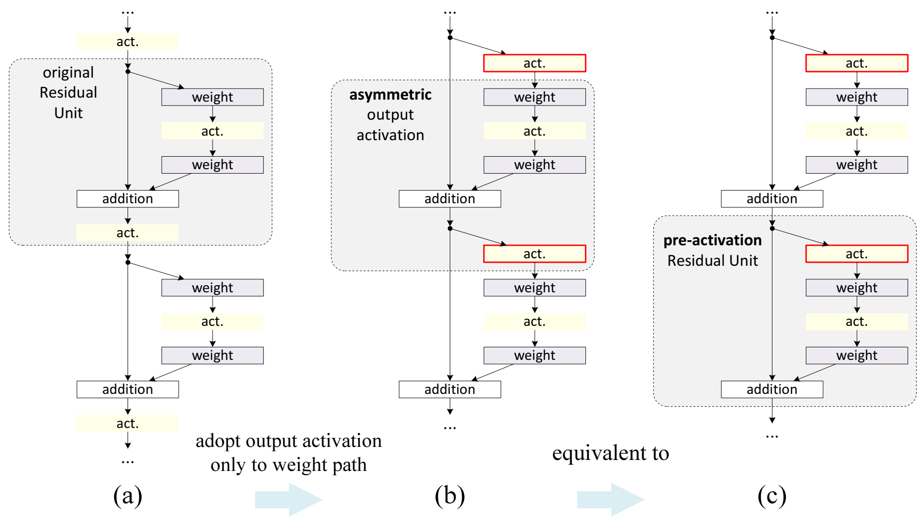


Figure 5. 使用非对称后加激活等价于构建一个*pre-activation*残差单元。

Table 3. 在CIFAR-10/100测试集上使用原始残差单元和我们的预激活残差单元的分类错误率(%)。

dataset	network	baseline unit	pre-activation unit
CIFAR-10	ResNet-110 (1layer skip)	9.90	8.91
	ResNet-110	6.61	6.37
	ResNet-164	5.93	5.46
	ResNet-1001	7.61	4.92
CIFAR-100	ResNet-164	25.16	24.33
	ResNet-1001	27.82	22.71

后激活/预激活之间的区别是由逐元素*addition*的存在引起的。对于一个具有N层的普通网络，存在N –个激活（BN/ReLU），将其视为后激活或预激活并不重要。但对于通过加法合并的分支层，激活的位置则至关重要。

我们尝试了两种此类设计：(i) 仅使用ReLU的预激活（图4(d)），以及(ii) 完整预激活（图4(e)），其中BN和ReLU均在权重层之前采用。表2显示，在ResNet-110/164上，仅ReLU的预激活效果与基线非常接近。该ReLU层未与BN层结合使用，因此可能无法受益于BN的优势[8]。

令人有些惊讶的是，当BN和ReLU都作为预激活使用时，结果得到了显著提升（表2和表3）。在表3中，我们报告了使用不同架构得到的结果：（i）ResNet-110，（ii）ResNet-164，（iii）一种110层的ResNet架构，其中每个快捷连接仅跳过1层（$i.e.$,

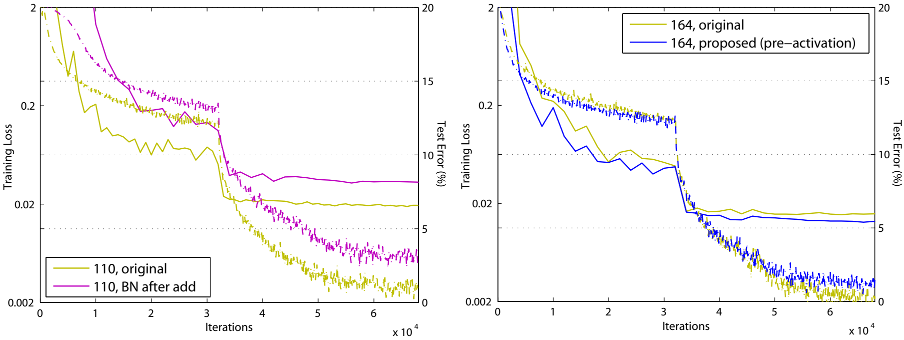


Figure 6. Training curves on CIFAR-10. **Left**: BN after addition (Fig. 4(b)) using ResNet-110. **Right**: pre-activation unit (Fig. 4(e)) on ResNet-164. Solid lines denote test error, and dashed lines denote training loss.

a Residual Unit has only 1 layer), denoted as "ResNet-110(1layer)", and (iv) a 1001-layer bottleneck architecture that has 333 Residual Units (111 on each feature map size), denoted as "ResNet-1001". We also experiment on CIFAR-100. Table 3 shows that our "pre-activation" models are consistently better than the baseline counterparts. We analyze these results in the following.

4.2 Analysis

We find the impact of pre-activation is twofold. First, the optimization is further eased (comparing with the baseline ResNet) because f is an identity mapping. Second, using BN as pre-activation improves regularization of the models.

Ease of optimization. This effect is particularly obvious when training the *1001-layer* ResNet. Fig. 1 shows the curves. Using the original design in [1], the training error is reduced very slowly at the beginning of training. For f = ReLU, the signal is impacted if it is negative, and when there are many Residual Units, this effect becomes prominent and Eqn.(3) (so Eqn.(5)) is not a good approximation. On the other hand, when f is an identity mapping, the signal can be propagated directly between any two units. Our 1001-layer network reduces the training loss very quickly (Fig. 1). It also achieves the lowest loss among all models we investigated, suggesting the success of optimization.

We also find that the impact of f = ReLU is not severe when the ResNet has fewer layers (*e.g.*, 164 in Fig. 6(right)). The training curve seems to suffer a little bit at the beginning of training, but goes into a healthy status soon. By monitoring the responses we observe that this is because after some training, the weights are adjusted into a status such that $\mathbf{y}_l$ in Eqn.(1) is more frequently above zero and f does not truncate it ($\mathbf{x}_l$ is always non-negative due to the previous ReLU, so $\mathbf{y}_l$ is below zero only when the magnitude of $\mathcal{F}$ is very negative). The truncation, however, is more frequent when there are 1000 layers.

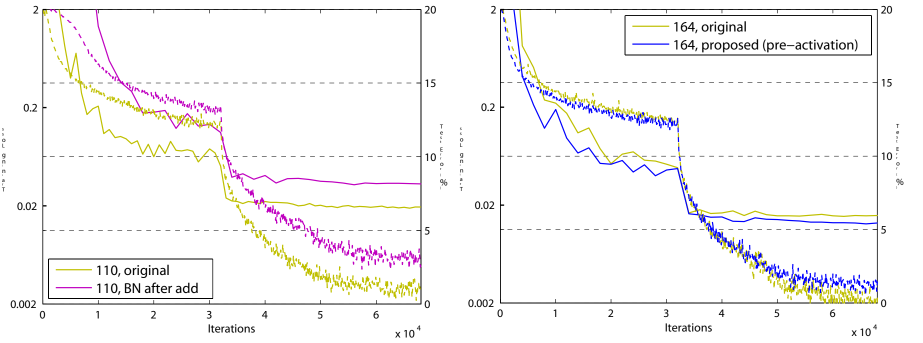


Figure 6. CIFAR-10上的训练曲线。**Left**：使用ResNet-110的加法后批归一化（图4(b)）。**Right**：在ResNet-164上的预激活单元（图4(e)）。实线表示测试误差，虚线表示训练损失。

一个残差单元只有1层），表示为“ResNet-110(1层)”，以及（iv）一个具有333个残差单元（每种特征图尺寸上111个）的1001层瓶颈架构，表示为“ResNet-1001”。我们还在CIFAR-100上进行了实验。表3显示，我们的“预激活”模型始终优于基线对应模型。我们将在下文中分析这些结果。

4.2 Analysis

我们发现预激活的影响是双重的。首先，由于f是恒等映射，优化过程进一步得到简化（与基准ResNet相比）。其次，使用BN作为预激活增强了模型的正则化效果。

Ease of optimization这种效应在训练*1001-layer* ResNet时尤为明显。图1展示了相关曲线。采用[1]中的原始设计时，训练误差在初始阶段下降非常缓慢。对于$f = \mathrm{ReLU}$而言，若信号为负值则会受到影响，当残差单元数量较多时，这种影响会变得显著，此时公式(3)（即公式(5)）不再是一个良好的近似。另一方面，当f作为恒等映射时，信号可以在任意两个单元间直接传播。我们提出的1001层网络能够极快地降低训练损失（图1），并且在所有研究模型中取得了最低的损失值，这证明了优化方案的成功性。

我们还发现，当ResNet的层数较少时（*e.g.*，图6（右）中的164层），$f = \mathrm{ReLU}$的影响并不严重。训练曲线在训练初期似乎受到一些影响，但很快便进入健康状态。通过监测响应情况，我们观察到这是因为经过一段时间的训练后，权重调整到一种状态，使得公式(1)中的$\mathbf{y}_l$更频繁地大于零，且f不会对其截断（由于之前的ReLU，$\mathbf{x}_l$始终为非负值，因此仅当$\mathcal{F}$的幅度为非常大的负值时，$\mathbf{y}_l$才会低于零）。然而，当网络有1000层时，截断现象则更为频繁。

Table 4. Comparisons with state-of-the-art methods on CIFAR-10 and CIFAR-100 using "*moderate data augmentation*" (flip/translation), except for ELU [12] with no augmentation. Better results of [13,14] have been reported using stronger data augmentation and ensembling. For the ResNets we also report the number of parameters. Our results are the median of 5 runs with mean±std in the brackets. All ResNets results are obtained with a mini-batch size of 128 except † with a mini-batch size of 64 (code available at https://github.com/KaimingHe/resnet-1k-layers).

CIFAR-10	error (%)
NIN [15]	8.81
DSN [16]	8.22
FitNet [17]	8.39
Highway [7]	7.72
All-CNN [14]	7.25
ELU [12]	6.55
FitResNet, LSUV [18]	5.84
ResNet-110 [1] (1.7M)	6.61
ResNet-1202 [1] (19.4M)	7.93
ResNet-164 [ours] (1.7M)	5.46
ResNet-1001 [ours] (10.2M)	4.92 (4.89±0.14)
ResNet-1001 [ours] (10.2M)†	**4.62** (4.69±0.20)

CIFAR-100	error (%)
NIN [15]	35.68
DSN [16]	34.57
FitNet [17]	35.04
Highway [7]	32.39
All-CNN [14]	33.71
ELU [12]	24.28
FitNet, LSUV [18]	27.66
ResNet-164 [1] (1.7M)	25.16
ResNet-1001 [1] (10.2M)	27.82
ResNet-164 [ours] (1.7M)	24.33
ResNet-1001 [ours] (10.2M)	**22.71** (22.68±0.22)

Reducing overfitting. Another impact of using the proposed pre-activation unit is on regularization, as shown in Fig. 6 (right). The pre-activation version reaches slightly higher training loss at convergence, but produces lower test error. This phenomenon is observed on ResNet-110, ResNet-110(1-layer), and ResNet-164 on both CIFAR-10 and 100. This is presumably caused by BN's regularization effect [8]. In the original Residual Unit (Fig. 4(a)), although the BN normalizes the signal, this is soon added to the shortcut and thus the merged signal is not normalized. This unnormalized signal is then used as the input of the next weight layer. On the contrary, in our pre-activation version, the inputs to all weight layers have been normalized.

5 Results

Comparisons on CIFAR-10/100. Table 4 compares the state-of-the-art methods on CIFAR-10/100, where we achieve competitive results. We note that we do not specially tailor the network width or filter sizes, nor use regularization techniques (such as dropout) which are very effective for these small datasets. We obtain these results via a simple but essential concept — going deeper. These results demonstrate the potential of *pushing the limits of depth*.

Comparisons on ImageNet. Next we report experimental results on the 1000-class ImageNet dataset [3]. We have done preliminary experiments using the skip connections studied in Fig. 2 & 3 on ImageNet with ResNet-101 [1], and observed similar optimization difficulties. The training error of these non-identity shortcut networks is obviously higher than the original ResNet at the first learning rate

Table 4. 在CIFAR-10和CIFAR-100数据集上，使用“*moderate data augmentation*”（翻转/平移）数据增强与现有先进方法进行比较，但ELU[12]未使用数据增强除外。[13,14]中通过采用更强的数据增强和集成方法报告了更好的结果。对于ResNet，我们还报告了参数量。我们的结果是5次运行的中位数，括号内为均值±标准差。所有ResNet结果均使用128的小批量获得，但†使用64的小批量（代码可在 `https://github.com/KaimingHe/resnet-1k-layers`获取）。

CIFAR-10	error (%)
NIN [15]	8.81
DSN [16]	8.22
FitNet [17]	8.39
Highway [7]	7.72
All-CNN [14]	7.25
ELU [12]	6.55
FitResNet, LSUV [18]	5.84
ResNet-110 [1] (1.7M)	6.61
ResNet-1202 [1] (19.4M)	7.93
ResNet-164 [ours] (1.7M)	5.46
ResNet-1001 [ours] (10.2M)	4.92 (4.89±0.14)
ResNet-1001 [ours] (10.2M)†	**4.62** (4.69±0.20)

CIFAR-100	error (%)
NIN [15]	35.68
DSN [16]	34.57
FitNet [17]	35.04
Highway [7]	32.39
All-CNN [14]	33.71
ELU [12]	24.28
FitNet, LSUV [18]	27.66
ResNet-164 [1] (1.7M)	25.16
ResNet-1001 [1] (10.2M)	27.82
ResNet-164 [ours] (1.7M)	24.33
ResNet-1001 [ours] (10.2M)	**22.71** (22.68±0.22)

Reducing overfitting使用所提出的预激活单元的另一个影响是正则化，如图6（右）所示。预激活版本在收敛时达到的训练损失略高，但产生的测试误差更低。这一现象在CIFAR-10和100数据集上的ResNet-110、ResNet-110（单层）和ResNet-164中均有观察到。这可能是由BN的正则化效应引起的[8]。在原始残差单元（图4(a)）中，尽管BN对信号进行了归一化，但该信号很快被添加到捷径连接中，因此合并后的信号并未归一化。这种未归一化的信号随后被用作下一个权重层的输入。相反，在我们的预激活版本中，所有权重层的输入都经过了归一化处理。

5 Results

Comparisons on CIFAR-10/100. 表4比较了在CIFAR-10/100数据集上的先进方法，我们取得了具有竞争力的结果。需要说明的是，我们并未专门调整网络宽度或滤波器尺寸，也未采用针对这类小数据集非常有效的正则化技术（如dropout）。我们通过一个简单但核心的理念——增加网络深度——获得了这些结果。这些结果展现了*pushing the limits of depth*的潜力。

Comparisons on ImageNet. 接下来，我们在1000类别的ImageNet数据集[3]上报告实验结果。我们使用图2和图3中研究的跳跃连接在ImageNet上使用ResNet-101[1]进行了初步实验，并观察到类似的优化困难。这些非恒等快捷连接网络的训练误差在第一个学习率阶段明显高于原始ResNet。

Table 5. Comparisons of single-crop error on the ILSVRC 2012 validation set. All ResNets are trained using the same hyper-parameters and implementations as [1]). Our Residual Units are the full pre-activation version (Fig. 4(e)). †: code/model available at https://github.com/facebook/fb.resnet.torch/tree/master/pretrained, using scale and aspect ratio augmentation in [20].

method	augmentation	train crop	test crop	top-1	top-5
ResNet-152, original Residual Unit [1]	scale	224×224	224×224	23.0	6.7
ResNet-152, original Residual Unit [1]	scale	224×224	320×320	21.3	5.5
ResNet-152, **pre-act** Residual Unit	scale	224×224	320×320	21.1	5.5
ResNet-200, original Residual Unit [1]	scale	224×224	320×320	21.8	6.0
ResNet-200, **pre-act** Residual Unit	scale	224×224	320×320	**20.7**	**5.3**
ResNet-200, **pre-act** Residual Unit	scale+asp ratio	224×224	320×320	**20.1**†	**4.8**†
Inception v3 [19]	scale+asp ratio	299×299	299×299	21.2	5.6

(similar to Fig. 3), and we decided to halt training due to limited resources. But we did finish a "BN after addition" version (Fig. 4(b)) of ResNet-101 on ImageNet and observed higher training loss and validation error. This model's single-crop (224×224) validation error is 24.6%/7.5%, *vs.* the original ResNet-101's 23.6%/7.1%. This is in line with the results on CIFAR in Fig. 6 (left).

Table 5 shows the results of ResNet-152 [1] and ResNet-200[3], all trained from scratch. We notice that the original ResNet paper [1] trained the models using scale jittering with shorter side $s \in [256, 480]$, and so the test of a 224×224 crop on $s = 256$ (as did in [1]) is negatively biased. Instead, we test a single 320×320 crop from $s = 320$, for all original and our ResNets. Even though the ResNets are trained on smaller crops, they can be easily tested on larger crops because the ResNets are fully convolutional by design. This size is also close to 299×299 used by Inception v3 [19], allowing a fairer comparison.

The original ResNet-152 [1] has top-1 error of 21.3% on a 320×320 crop, and our pre-activation counterpart has 21.1%. The gain is not big on ResNet-152 because this model has not shown severe generalization difficulties. However, the original ResNet-200 has an error rate of 21.8%, higher than the baseline ResNet-152. But we find that the original ResNet-200 has *lower* training error than ResNet-152, suggesting that it suffers from overfitting.

Our pre-activation ResNet-200 has an error rate of 20.7%, which is **1.1%** lower than the baseline ResNet-200 and also lower than the two versions of ResNet-152. When using the scale and aspect ratio augmentation of [20,19], our ResNet-200 has a result better than Inception v3 [19] (Table 5). Concurrent with our work, an Inception-ResNet-v2 model [21] achieves a single-crop result of 19.9%/4.9%. We expect our observations and the proposed Residual Unit will help this type and generally other types of ResNets.

Computational Cost. Our models' computational complexity is linear on

[3] The ResNet-200 has 16 more 3-layer bottleneck Residual Units than ResNet-152, which are added on the feature map of 28×28.

Table 5. 在ILSVRC 2012验证集上的单作物误差比较。所有ResNet均采用与[1]相同的超参数和实现进行训练。我们的残差单元采用完整的预激活版本（图4(e)）。†：代码/模型可在`https://github.com/facebook/fb.resnet.torch/tree/master/pretrained`获取，使用了[20]中的尺度和宽高比增强方法。

method	augmentation	train crop	test crop	top-1	top-5
ResNet-152, original Residual Unit [1]	scale	224×224	224×224	23.0	6.7
ResNet-152, original Residual Unit [1]	scale	224×224	320×320	21.3	5.5
ResNet-152, **pre-act** Residual Unit	scale	224×224	320×320	21.1	5.5
ResNet-200, original Residual Unit [1]	scale	224×224	320×320	21.8	6.0
ResNet-200, **pre-act** Residual Unit	scale	224×224	320×320	**20.7**	**5.3**
ResNet-200, **pre-act** Residual Unit	scale+asp ratio	224×224	320×320	**20.1**†	**4.8**†
Inception v3 [19]	scale+asp ratio	299×299	299×299	21.2	5.6

(类似于图3)，由于资源有限，我们决定停止训练。但我们确实完成了ResNet-101在ImageNet上的“加法后批归一化”版本（图4(b)），并观察到更高的训练损失和验证误差。该模型的单次裁剪（224×224）验证误差为24.6%/7.5%，*vs.*而原始ResNet-101的误差为23.6%/7.1%。这与图6（左）中CIFAR数据集上的结果一致。

表5展示了ResNet-152[1]和ResNet-200[3]的结果，所有模型均从头开始训练。我们注意到原始ResNet论文[1]采用短边尺寸在$s \in$[256,480]范围内随机缩放的增强方式训练模型，因此在$s =$256图像上测试224×224裁剪区域（如[1]所示）会存在负向偏差。为此，我们对所有原始ResNet及我们的改进模型均采用从$s =$320图像中单次裁剪320×320区域进行测试。尽管ResNet训练时使用较小裁剪区域，但由于其设计为全卷积结构，可轻松在更大裁剪区域上测试。该尺寸也接近Inception v3[19]使用的299×299输入尺寸，使比较更公平。

原始的ResNet-152 [1]在320×320裁剪图像上的top-1错误率为21.3%，而我们采用预激活结构的对应模型为21.1%。在ResNet-152上提升并不显著，因为该模型未表现出严重的泛化困难。然而，原始ResNet-200的错误率达到21.8%，高于基准ResNet-152。但我们发现原始ResNet-200的训练误差$lower$低于ResNet-152，这表明其存在过拟合问题。

我们的预激活ResNet-200错误率为20.7%，比基准ResNet-200低**1.1%**，同时也低于两个版本的ResNet-152。当采用[20,19]中的尺度和宽高比增强方法时，我们的ResNet-200取得了优于Inception v3 [19]的结果（表5）。与我们工作同期，Inception-ResNet-v2模型[21]取得了19.9%/4.9%的单次裁剪结果。我们期望本文的观察结果及提出的残差单元能对此类及更广义的ResNet模型有所助益。

Computational Cost. 我们模型的计算复杂度在

[3] The ResNet-200 has 16 more 3-layer bottleneck Residual Units than ResNet-152, which are added on the feature map of 28×28.

depth (so a 1001-layer net is ∼10× complex of a 100-layer net). On CIFAR, ResNet-1001 takes about 27 hours to train on 2 GPUs; on ImageNet, ResNet-200 takes about 3 weeks to train on 8 GPUs (on par with VGG nets [22]).

6 Conclusions

This paper investigates the propagation formulations behind the connection mechanisms of deep residual networks. Our derivations imply that identity shortcut connections and identity after-addition activation are essential for making information propagation smooth. Ablation experiments demonstrate phenomena that are consistent with our derivations. We also present 1000-layer deep networks that can be easily trained and achieve improved accuracy.

Appendix: Implementation Details The implementation details and hyper-parameters are the same as those in [1]. On CIFAR we use only the translation and flipping augmentation in [1] for training. The learning rate starts from 0.1, and is divided by 10 at 32k and 48k iterations. Following [1], for all CIFAR experiments we warm up the training by using a smaller learning rate of 0.01 at the beginning 400 iterations and go back to 0.1 after that, although we remark that this is not necessary for our proposed Residual Unit. The mini-batch size is 128 on 2 GPUs (64 each), the weight decay is 0.0001, the momentum is 0.9, and the weights are initialized as in [23].

On ImageNet, we train the models using the same data augmentation as in [1]. The learning rate starts from 0.1 (no warming up), and is divided by 10 at 30 and 60 epochs. The mini-batch size is 256 on 8 GPUs (32 each). The weight decay, momentum, and weight initialization are the same as above.

When using the pre-activation Residual Units (Fig. 4(d)(e) and Fig. 5), we pay special attention to the first and the last Residual Units of the entire network. For the first Residual Unit (that follows a stand-alone convolutional layer, conv_1), we adopt the first activation right after conv_1 and before splitting into two paths; for the last Residual Unit (followed by average pooling and a fully-connected classifier), we adopt an extra activation right after its element-wise addition. These two special cases are the natural outcome when we obtain the pre-activation network via the modification procedure as shown in Fig. 5.

The bottleneck Residual Units (for ResNet-164/1001 on CIFAR) are constructed following [1]. For example, a $\begin{bmatrix} 3\times3,\ 16 \\ 3\times3,\ 16 \end{bmatrix}$ unit in ResNet-110 is replaced with a $\begin{bmatrix} 1\times1,\ 16 \\ 3\times3,\ 16 \\ 1\times1,\ 64 \end{bmatrix}$ unit in ResNet-164, both of which have roughly the same number of parameters. For the bottleneck ResNets, when reducing the feature map size we use projection shortcuts [1] for increasing dimensions, and when pre-activation is used, these projection shortcuts are also with pre-activation.

深度（因此一个1001层的网络是$\sim$10×复杂度的100层网络）。在CIFAR上，ResNet-1001在2个GPU上训练大约需要27小时；在ImageNet上，ResNet-200在8个GPU上训练大约需要3周（与VGG网络相当[22]）。

6 Conclusions

本文研究了深度残差网络连接机制背后的传播公式。我们的推导表明，恒等快捷连接和恒等后加激活对于实现信息平滑传播至关重要。消融实验展示了与我们的推导一致的现象。我们还提出了易于训练的1000层深度网络，并实现了更高的准确率。

Appendix: Implementation Details 实现细节和超参数与[1]中相同。在CIFAR数据集上，我们仅采用[1]中的平移和翻转增强进行训练。初始学习率为0.1，在32k和48k迭代时降至原值的十分之一。遵循[1]的方法，在所有CIFAR实验中，我们通过前400次迭代使用0.01的较小学习率进行训练预热，之后恢复至0.1——尽管需要说明这种预热对我们的残差单元并非必需。训练使用2块GPU（每块64样本），迷你批次大小为128，权重衰减为0.0001，动量为0.9，权重初始化方式与[23]一致。

在ImageNet上，我们采用与[1]相同的数据增强方法训练模型。初始学习率为0.1（无预热阶段），并在第30轮和第60轮时除以10。在8个GPU上使用的小批量大小为256（每个GPU处理32个样本）。权重衰减、动量及权重初始化设置均与上文相同。

在使用预激活残差单元（图4(d)(e)及图5）时，我们特别关注整个网络的第一个和最后一个残差单元。对于第一个残差单元（位于独立卷积层conv_1之后），我们在conv_1之后、分流为两条路径之前采用第一个激活；对于最后一个残差单元（后接平均池化和全连接分类器），我们在其逐元素相加后额外采用一次激活。这两种特殊情况是通过图5所示的修改流程得到预激活网络时的自然结果。

瓶颈残差单元（用于CIFAR上的ResNet-164/1001）按照[1]构建。例如，ResNet-110中的$\begin{bmatrix}3\times3,\ 16\\3\times3,\ 16\end{bmatrix}$单元在ResNet-164中被替换为$\begin{bmatrix}1\times1,\ 16\\3\times3,\ 16\\1\times1,\ 64\end{bmatrix}$单元，两者参数数量大致相同。对于瓶颈残差网络，在减小特征图尺寸时，我们使用投影捷径[1]来增加维度；当采用预激活时，这些投影捷径同样采用预激活处理。

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
2. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML. (2010)
3. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. IJCV (2015)
4. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: ECCV. (2014)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation (1997)
6. Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks. In: ICML workshop. (2015)
7. Srivastava, R.K., Greff, K., Schmidhuber, J.: Training very deep networks. In: NIPS. (2015)
8. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML. (2015)
9. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural computation (1989)
10. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech Report (2009)
11. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv:1207.0580 (2012)
12. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (ELUs). In: ICLR. (2016)
13. Graham, B.: Fractional max-pooling. arXiv:1412.6071 (2014)
14. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: The all convolutional net. arXiv:1412.6806 (2014)
15. Lin, M., Chen, Q., Yan, S.: Network in network. In: ICLR. (2014)
16. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. In: AISTATS. (2015)
17. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. In: ICLR. (2015)
18. Mishkin, D., Matas, J.: All you need is a good init. In: ICLR. (2016)
19. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR. (2016)
20. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR. (2015)
21. Szegedy, C., Ioffe, S., Vanhoucke, V.: Inception-v4, inception-resnet and the impact of residual connections on learning. arXiv:1602.07261 (2016)
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2015)
23. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV. (2015)

References

1. 何恺明、张祥雨、任少卿、孙剑：深度残差学习用于图像识别。载于：CVPR。（2016）2. Nair, V., Hinton, G.E.：修正线性单元改进受限玻尔兹曼机。载于：ICML。（2010）3. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.：ImageNet 大规模视觉识别挑战赛。IJCV（2015）4. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.：Microsoft COCO：上下文中的常见物体。载于：ECCV。（2014）5. Hochreiter, S., Schmidhuber, J.：长短期记忆。神经计算（1997）6. Srivastava, R.K., Greff, K., Schmidhuber, J.：高速公路网络。载于：ICML 研讨会。（2015）7. Srivastava, R.K., Greff, K., Schmidhuber, J.：训练极深网络。载于：NIPS。（2015）8. Ioffe, S., Szegedy, C.：批量归一化：通过减少内部协变量偏移加速深度网络训练。载于：ICML。（2015）9. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.：反向传播应用于手写邮政编码识别。神经计算（1989）10. Krizhevsky, A.：从微小图像中学习多层特征。技术报告（2009）11. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.：通过防止特征检测器的共适应改进神经网络。arXiv:1207.0580（2012）12. Clevert, D.A., Unterthiner, T., Hochreiter, S.：通过指数线性单元（ELU）实现快速准确的深度网络学习。载于：ICLR。（2016）13. Graham, B.：分数最大池化。arXiv:1412.6071（2014）14. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.：追求简洁：全卷积网络。arXiv:1412.6806（2014）15. Lin, M., Chen, Q., Yan, S.：网络中的网络。载于：ICLR。（2014）16. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.：深度监督网络。载于：AISTATS。（2015）17. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.：Fitnets：瘦深度网络的提示。载于：ICLR。（2015）18. Mishkin, D., Matas, J.：你只需要一个好的初始化。载于：ICLR。（2016）19. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.：重新思考计算机视觉的初始架构。载于：CVPR。（2016）20. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.：通过卷积深入探索。载于：CVPR。（2015）21. Szegedy, C., Ioffe, S., Vanhoucke, V.：Inception-v4、Inception-ResNet 以及残差连接对学习的影响。arXiv:1602.07261（2016）22. Simonyan, K., Zisserman, A.：用于大规模图像识别的极深卷积网络。载于：ICLR。（2015）23. 何恺明、张祥雨、任少卿、孙剑：深入研究整流器：超越 ImageNet 分类的人类水平性能。载于：ICCV。（2015）