Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun
Microsoft Research
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

论文地址：https://arxiv.org/abs/1512.03385

代码：https://github.com/KaimingHe/deep-residual-networks

译者：彭博（初译） 王行凤（复译） 范诗剑（审校）
组织名称：七月在线DL翻译组

Abstract

神经网络的训练因其层次加深而变得愈加困难。本文提出了一种残差学习框架，能够简化使那些非常深的网络的训练。相较之前网络所学习的是无参考的函数*(unreferenced functions)*，我们显著改进的网络结构可以根据网络的输入对其残差函数进行学习。我们的提供了详实的实验证明，这些残差网络的优化更简单，而且能通过大大增加的深度获得更高的精度。（译者注：残差网络的深度可以做的更深，从而提高更好的精度、网络能力）。在*ImageNet*数据集上评估了一个拥有*152*层，相当于*VGG [40]*网络的*8*倍深度但是仍然具有较低复杂度的残差网络。这些残差网络的一个组合模型（*ensemble*）在*ImageNet*测试集上的错误率仅为*3.57%*。这个结果在*2015*年的*ILSVRC*分类任务上获得了第一名的成绩。我们在*CIFAR-10*上对*100*层和*1000*层的残差网络也进行了分析。

表达的深度在很多视觉识别任务中具有是核心影响力的。所以仅仅因为我们非同一般的深度表示，便在*COCO*目标检测数据集上获得了*28%* 的相对提升。深度残差网络是我们参加*ILSVRC & COCO 2015* 竞赛上所使用模型的基础[1]，并且我们在*ImageNet*检测、*ImageNet*定位、*COCO*检测以及*COCO*分割上均获得了第一名的成绩。

1. Introduction

深度卷积神经网络[22, 21] 在图像分类领域取得了一系列的突破[21, 50, 39]。深度网络以一个端到端的多层方式自然的集成了低、中、高级特征[50]与分类器，多层特征可以通过堆叠的层的数量来丰富其表达。研究结果[40, 44]表明，网络的深度具有至关重要的作用，这样导致了ImageNet竞赛[34]的参赛模型[40, 44, 13, 16] 都趋向于“非常深” [40] ——从16 层[40] 到30层[16]。许多其它重要的视觉识别任务[8, 12, 7, 31, 26] 的都得益于非常深的模型。

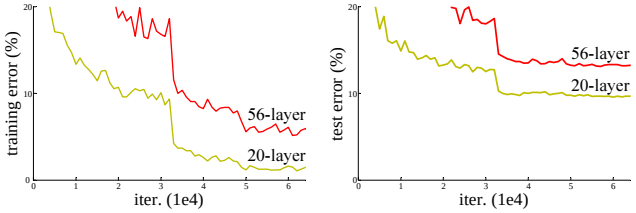

Figure 1. 20层和56层的“平铺”网络在CIFAR-10上的训练错误率（左）和测试错误率（右）。越深的网络在训练和测试上都具有越高的错误率。Fig. 4中ImageNet数据集上也出现了类似的现象。

在深度重要性的驱使下，出现了一个新的问题：是不是堆叠的层越多越容易学习到更好的网络模型？ 回答这个问题之前首先面对的一个障碍便是困扰人们很久的梯度消失/梯度爆炸[1, 9]，这从一开始便阻碍了模型的收敛。然而，规范初始化[23, 9, 35, 13] 和中值规范化层[16]在很大程度上解决了这一问题，它使得数十层的网络在利用随机梯度下降（SGD）的反向传播[22]求解上能够收敛。

当深层网络能够开始收敛时，一个退化 问题又出现了：随着网络深度的增加，准确率达到饱和（不足为奇）然后迅速退化。意外的是，这种退化并不是由过拟合造成的，并且在一个合理的深度模型中增加更多的层却导致了更高的错误率 [11, 41]，我们的实验也证明了这点。Fig. 1展示了典型的案列。

退化的出现（训练精度/准确率）表明了并非所有的系统都是很容易优化的。让我们来比较一个浅层的框架和它的深层版本。对于更深的模型，这有一种通过构建的解决方案：恒等映射（identity mapping）来构建新添加的层，而其它层直接从浅层模型中复制而来。这个构建的解决方案也表明了，一个更深的模型不应当产生比它的浅层版本更高的训练错误率。实验表明，我们目前无法找到一个与这种构建的解决方案

[1] http://image-net.org/challenges/LSVRC/2015/ and http://mscoco.org/dataset/#detections-challenge2015.

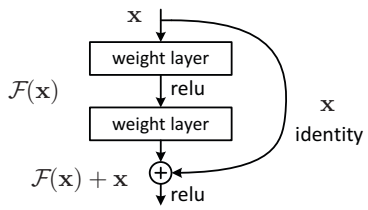

Figure 2. 残差学习：一个构建块。

相当或者更好的方案（或者说无法在合适的时间内实现）。

本文中，我们提出了一种深度残差学习框架来解决退化问题。与其希望让每组少数的几个层就直接拟合出期望的实际映射关系，我们明确的让这些层拟合残差映射。假设期望的底层映射为$\mathcal{H}(\mathbf{x})$，我们让堆叠的非线性层来拟合另一个映射：$\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$。因此原来的映射转化为：$\mathcal{F}(\mathbf{x}) + \mathbf{x}$。我们推断残差映射比原始未参考的映射（unreferenced mapping）更容易优化。（译者注：常规的，原先需要使用layers映射desired underlying mapping，即$\mathcal{H}(\mathbf{x})$，现在把这个映射拆分成$\mathcal{F}(\mathbf{x}) + \mathbf{x}$来处理，原先的original 直接需要映射到$\mathcal{H}(\mathbf{x})$，现在original 成了$\mathcal{F}(\mathbf{x})+\mathbf{x}$，这样优化$\mathcal{F}(\mathbf{x})$比优化$\mathcal{H}(\mathbf{x})$ 要简单容易些）。在极端的情况下，如果某个恒等映射是最优的，那么将残差置为0 比用非线性层的堆叠来拟合恒等映射更简单。(复译注：在一些极端情况下，恒等映射已经优化了，这个时候置需要把残差设置为0可以了，这比原先的使用堆叠的层拟合恒等映射简单很多。)

公式$\mathcal{F}(\mathbf{x}) + \mathbf{x}$ 可以通过前馈神经网络的“捷径连接(捷径connections)”来实现(Fig. 2)。捷径连接[2, 32, 49] 就是跳过一个或者多个层。在我们的例子中，捷径连接只是简单的执行恒等映射，再将它们的输出和堆叠层的输出叠加在一起(Fig. 2)。恒等的捷径连接并不增加额外的参数和计算复杂度。整个的网络仍然可以通过结合反向传播的SGD端到端训练，并且能够简单的通过公共库(例如，Caffe [19])来实现而无需修改求解器（solvers）。

我们在ImageNet数据集上[34]提供了全面的实验演示了退化问题并评估了我们提出的方法。本文表明了：1) 我们极深的残差网络是很容易优化的，但是对应的“平铺”网络(仅是堆叠了层)随着深度的增加表现出更高的错误率; 2) 我们的深度残差网络能够轻易的由增加层来提高准确率，并且结果也大大优于以前的网络。

CIFAR-10数据集上[20]也出现了类似的现象，这表明了优化难度和我们的方法的效果并不仅仅是针对某个特定的数据集。我们在这个数据集上成功的提出了超过100层的训练模型，并探索了超过1000层的模型。

在ImageNet分类数据集上[34]，极深的残差网络获得了优异的成绩。我们的152层残差网路是迄今为止使用在ImageNet上最深的网络，并且比VGG网络[40]的复杂度还要低。在ImageNet测试集上，我们的组合模型(ensemble)的top-5 错误率仅为**3.57%**，这个结果赢得了ILSVRC 2015分类竞赛的第一名。这个极深的模型在其他识别任务上同样也具有非常好的泛化性能，这让我们在ILSVRC & COCO 2015 竞赛的ImageNet 检测、ImageNet定位、COCO检测以及COCO分割上均获得了第一名的成绩。这强有力的证明了残差学习法则的通用性，因此我们将把它应用到其他视觉甚至非视觉问题上。

2. Related Work

残差表达。 在图像识别中，VLAD [18]是残差向量对应于字典进行编码的一种表达形式，Fisher Vector [29]可以看做是VLAD 的一个概率版本[18]。对于图像检索和分类[4, 48]它们都是强力的浅层表达。对于向量量化，残差向量编码[17]比原始向量编码更加有效。

在底层视觉和计算机图形学中，为了求解偏微分方程（PDEs），通常使用多重网格法(Multigrid) [3]将系统重新表达成多尺度的子问题来解决，每一个子问题就是解决粗细尺度之间的残差问题。多重网格法的另外一种方式是分层基预处理[45, 46]，它依赖于代表着两个尺度之间残差向量的变量。实验证明[3, 45, 46] 这些求解器收敛速度比那些没有意识到残差特性的标准求解器要快得多。这些方法表明了一个好的再表达式或者预处理能够简化优化问题。

捷径连接。 捷径连接[2, 32, 49] 已经经过了很长的一段实践和理论研究过程。里有一个最早的实践就是在训练多层感知器(MLPs)的时候从输入到输出之间添加一个线性层[32, 49]。在[43, 24]中，将一些中间层直接与辅助分类器相连接可以解决梯度消失/爆炸问题。[36, 37, 30, 47] 中提出了利用捷径连接将层响应、梯度以及传播误差中心化的方法。在[43]中，一个“inception”层由一个捷径分支和一些更深的分支组合而成。

和我们同期的工作也有一些，“高速公路网络(highway networks)”[41, 42] 将捷径连接与门控函数[15] 结合起来。这些门是数据相关并且是有额外参数的，而我们的恒等捷径是无参数的。当门控捷径是“关闭”（接近于0）时，高速公路网络中的层表示非残差函数。相反的，我们对残差函数的学习贯穿了我们的方法;我们的恒等捷径永远不会关闭的，所有的信息总是透传过去的，并藉此学习残差函数。此外，高速公路网络并不能由增加层的深度（例如超过100 层）来提高准确率。

3. Deep Residual Learning

3.1. Residual Learning

我们将$\mathcal{H}(\mathbf{x})$看作一个由部分堆叠的层（不需要一定是整个网络）来拟合的底层映射，其中$\mathbf{x}$是这些层的输入。如果多个非线性层能够逼近复杂的函数的假设[2]是成立的，那么就等价于这些层能够逼近复杂的残差函数，比如，$\mathcal{H}(\mathbf{x}) - \mathbf{x}$（假设输入和输出的维度

[2]该假设仍然是一个未解决的问题。详见[27].

相同）。所以我们也可以显式的用它们来逼近残差函数：$\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$ 而不是$\mathcal{H}(\mathbf{x})$。因此原始函数变成了：$\mathcal{F}(\mathbf{x}) + \mathbf{x}$。尽管这两个形式应该都能够逼近目标函数（根据上述假设），然而学习的难度可能存在差异的。

此番重构的动因由于退化问题中所表现出的反常现象(Fig. 1，左)。正如我们在introduction中对这个问题的所做出的说明那样，如果能够以恒等映射的方式来构建所增加的层，一个加深的模型的训练误差就不会大于它所基于的较浅的模型。退化问题表明利用多个非线性网络层对于恒等映射作逼近可能会存在求解上的困难。而通过残差学习的办法重构这个方法，如果恒等映射是最优的，那么求解器驱使多个非线性层的权重趋向于零来逼近恒等映射。

在实际情况下，恒等映射不大可能一开始就已经达到最优了，然而我们重构可能有助于问题的预处理。如果最优函数更趋近于恒等映射而不是零值映射，求解器更可能参考一个恒等映射的办法来确定扰动，而不是将其作为一个全新的函数来学习。通过实验(Fig. 7)表明，学习到的残差函数通常响应较小，这表明恒等映射是一种合理的预处理手段。

3.2. Identity Mapping by Shortcuts

我们在堆叠层上采取残差学习算法。一个构建块如Fig. 2所示。本文中的构建块定义如下：

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

其中$\mathbf{x}$ 和$\mathbf{y}$ 分别表示层的输入和输出。函数$\mathcal{F}(\mathbf{x}, \{W_i\})$代表着残差映射。Fig. 2中的例子包含两层，$\mathcal{F} = W_2\sigma(W_1\mathbf{x})$，其中$\sigma$代表ReLU [28]，为了简化省略了偏置项。$\mathcal{F} + \mathbf{x}$操作由一个捷径连接和元素级(element-wise)的加法来表示。在加法之后我们再执行另一个非线性操作(*i.e.*, $\sigma(\mathbf{y})$，如Fig. 2).

Eqn.(1)中的捷径连接没有增加额外的参数和计算复杂度。这不仅是一个很有吸引力的做法，同时在对平铺和残差网络进行比较时也尤其重要。这样我们可以公平的对平铺和残差网络在参数、深度、宽度和计算代价相同的情况下作比较（除了可以忽略不计的元素级的加法）。

在Eqn.(1)中，$\mathbf{x}$和$\mathcal{F}$的维度必须相同。如果不相同（例如，当输入和输出通道的数量发生改变时），我们可以通过捷径连接执行一个线性映射W_s 来匹配两者的维度：

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

在Eqn.(1)中同样可以使用方阵W_s。但是我们将通过实验展示恒等就足以经济的解决问题，因此W_s只是用来解决维度不匹配的问题。

残差函数$\mathcal{F}$的形式是灵活可变的。本文实验中涉及到的函数$\mathcal{F}$为两层或者三层的(Fig. 5)，当然更多层也是可行的。但是如果$\mathcal{F}$只含有一层，Eqn.(1)就和线性函数：$\mathbf{y} = W_1\mathbf{x} + \mathbf{x}$一致，因此并不具有任何优势。

我们还发现不仅是对于全连接层，对于卷积层也是同样适用的。函数$\mathcal{F}(\mathbf{x}, \{W_i\})$可以表示多个卷积层，可以在两个特征图的通道之间执行元素级的加法。

3.3. Network Architectures

我们在多个平铺网络和残差网络上进行了测试，并都观测到了一致的现象。为了给讨论提供实证，接下来我们将描述应用于ImageNet数据集的两种网络结构。

平铺网络。 我们的平铺网络结构(Fig. 3，中)主要受VGG网络[40] (Fig. 3，左)的启发。卷积层主要为3×3的滤波器，并遵循以下两点要求：(i) 输出特征尺寸相同的层需含有相同数量的滤波器；(ii) 如果特征尺寸减半，则滤波器的数量需要加倍以保证每层的时间复杂度相同。我们直接通过stride 为2的卷积层来进行下采样。在网络的最后是一个全局的平均pooling层和一个1000 类的包含softmax的全连接层。加权层的层数为34，如Fig. 3 (中)所示。

值得注意的是，我们的模型比VGG网络[40] (Fig. 3，左)有更少的滤波器和更低的计算复杂度。我们34层的结构含有36亿个FLOPs（乘-加），而这仅仅只有VGG-19 （196亿个FLOPs）的18%。

残差网络。 在以上平铺网络的基础上，我们插入捷径连接(Fig. 3，右)，将网络变成了对应的残差版本。如果输入和输出的维度相同，可以直接使用恒等捷径(Eqn.(1))（Fig. 3中的实线部分）。当维度增加时（Fig. 3中的虚线部分），考虑两种方案：(A) 捷径仍然使用恒等映射，在增加的维度上使用0来填充，这样做不会增加额外的参数；(B) 使用Eqn.(2)的映射捷径来使维度保持一致（通过1×1的卷积）。对于这两种方案，当捷径连接了两种尺寸的特征图时，均使用stride为2的卷积。

3.4. Implementation

针对ImageNet的网络实现遵循了[21, 40]。图片按其短边作等比缩放后按照$[256, 480]$ 区间的尺寸随机采样进行尺度增强[40]。随机的从图像或水平镜像采样大小为224×224的裁剪图像(crop)，每个像素都减去均值[21]。图像使用标准的颜色增强[21]。我们在每一个卷积层之后，激活层之前均使用批量正规化(batch normalization, BN)[16]。我们根据[12]来初始化权值然后从零开始训练所有平铺和残差网络。我们使用的mini-batch的尺寸为256。学习率从0.1开始，每当错误率平稳时将学习率除以10，整个模型进行60×10^4次迭代训练。我们将权值衰减设置为0.0001，动量为0.9。根据[16]，我们并没有使用Dropout [14]。

在测试中，为了对结果作对比我们采用了标准的10折(10-crop)测试[21]。为了达到最佳的结果，我们使用[40, 12]中的全连接卷积形式网络，最终结果为对多个尺寸图像的实验结果得分取平均值（调整图像的大小使它的短边长度分别为$\{224, 256, 384, 480, 640\}$）。

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

Table 1. 对应于ImageNet的结构框架。括号中为构建块的参数(同样见Fig. 5)，数个构建块进行堆叠。下采样由stride为2的conv3_1、conv4_1和conv5_1 来实现。

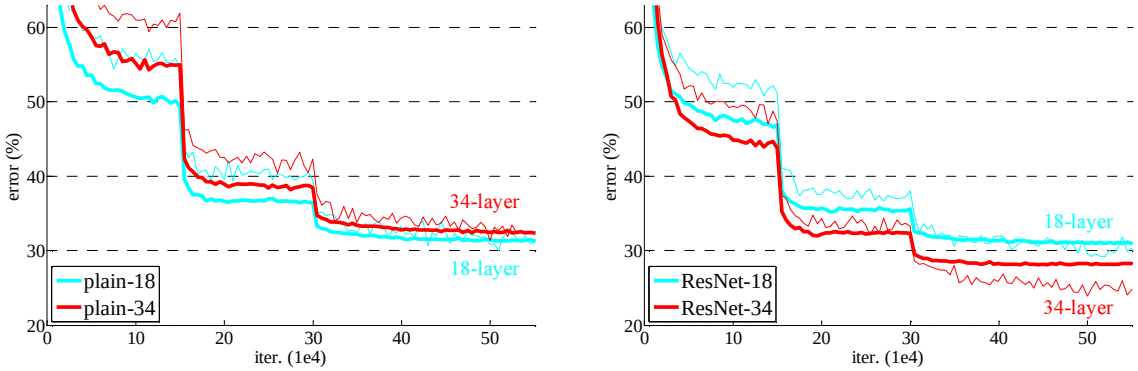

Figure 4. 在**ImageNet**上进行训练。细曲线为训练错误率，粗曲线为使用中心crop时的验证错误率。左：18和34层的平铺网络。右：18 和34层的ResNets。在这个图中，残差网络和对应的平铺网络相比并没有增加额外的参数。

4. Experiments

4.1. ImageNet Classification

本文在1000类的ImageNet2012数据集上[34]对我们的方法进行评估。训练集包含128万张图像，验证集包含5万张图像。我们在10万张测试图像上进行测试，并对top-1和top-5 的错误率进行评估。

平铺网络。 我们首先评估了18层和34层的平铺网络。34层的网络如图Fig. 3 (中)所示。18层的结构很相似，具体细节参见Table 1。

Table 2中展示的结果表明了34层的网络比18层的网络具有更高的验证错误率。为了揭示产生这种现象的原因，在Fig. 4 (左)中我们比较了整个训练过程中的训练及验证错误率。从结果中我们观测到了明显的退化问题——在整个训练过程中34 层的网络具有更高的训练错误率，即使18层网络的解空间为34层解空间的一个子空间。

我们认为这种优化上的困难不太可能是由梯度消失所造成的。因为这些平铺网络的训练使用了BN [16]，这能保证前向传递的信号是具有非零方差的。我们同样验证了在反向传递阶段的梯度由于BN而具有良好的范式，所以在前向和反向阶段的信号不会存在梯度消失的问题。事实上34层的平铺网络仍然具有不错的准确率(Table 3)，这表明了求解器在某种程度上也是有效的。我们推测，深层的平铺网络的收敛率是指数衰减的，这可能会影响训练错误率的降低[3]。这种优化困难的原因我们将在以后的工作中进行研究。

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

Table 2. ImageNet验证集上的Top-1错误率(%, 10-crop testing)。这里的ResNets并没有额外增加的参数。Fig. 4展示了整个训练过程。

残差网络。 接下来我们对18层和34层的残差网络ResNets进行评估。如Fig. 3 (右)所示，ResNets的基本框架和平铺网络的基本相同，除了在每一对3×3的滤波器上添加了一个捷径连接。在Table 2以及Fig. 4(右)的比较中，所有的捷径都是恒等映射，并且使用0对增加的维度进行填充(方案A)。因此他们并没有增加额外的参数。

我们从Table 2和Fig. 4中观测到以下三点：第一，与

[3]我们使用了更多的训练迭代次数(3×) 但是仍然观测到了退化的问题，这表明了简单的增加迭代次数并不能有效的解决这个问题。

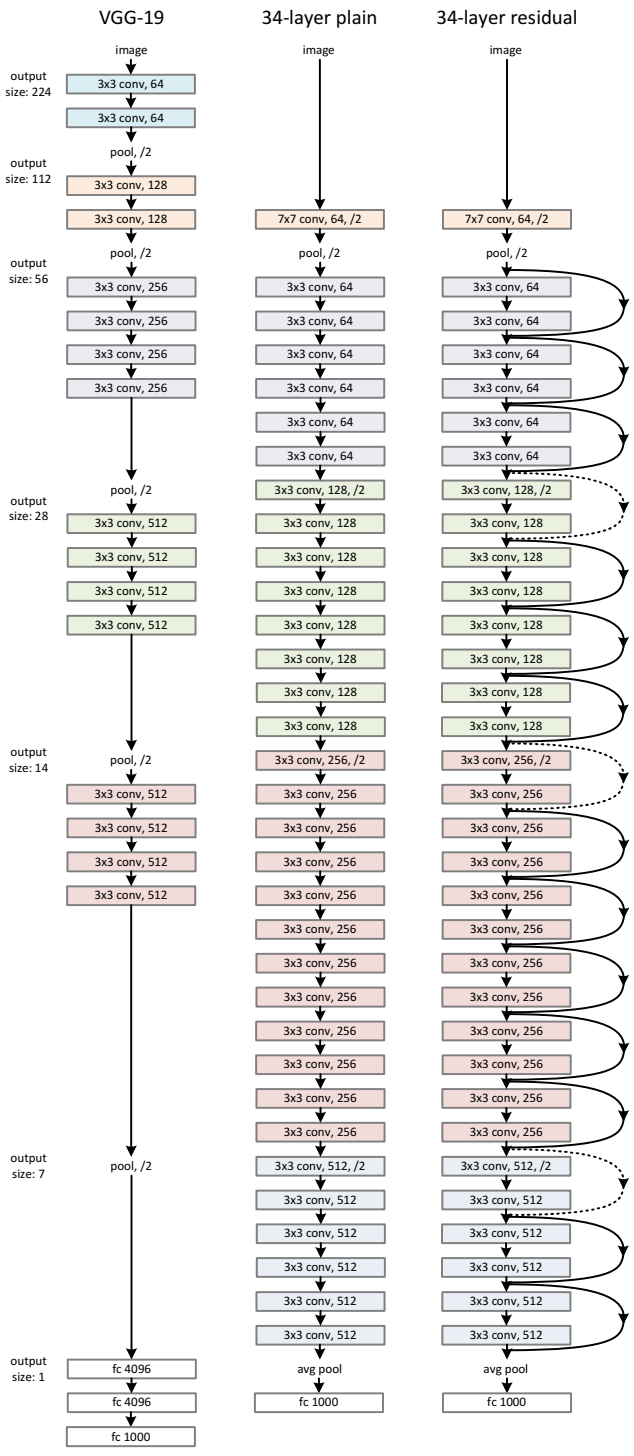

Figure 3. 对应于ImageNet的网络框架举例。**左**：VGG-19模型[40]（196亿个FLOPs）作为参考。**中**：平铺网络，含有34个参数层（36亿个FLOPs）。**右**：残差网络，含有34个参数层（36亿个FLOPs）。虚线表示的捷径增加了维度。**Table 1**展示了更多细节和其它变量。

model	top-1 err.	top-5 err.
VGG-16 [40]	28.07	9.33
GoogLeNet [44]	-	9.15
PReLU-net [13]	24.27	7.38
平铺-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. 在ImageNet验证集上的错误率(%，**10-crop** testing)。VGG-16是基于我们测试的网络。ResNet-50/101/152使用了方案B——利用映射来匹配增加的维度。

平铺网络相反，34层的ResNet比18层ResNet的结果更优(2.8%)。更重要的是，34层的ResNet在训练集和验证集上均展现出了更低的错误率。这表明了这种设置可以很好的解决退化问题，并且我们可以由增加的深度来提高准确率。

第二，与对应的平铺网络相比，34层的ResNet在top-1错误率上降低了3.5% (Table 2)，这得益于训练错误率的降低(Fig. 4 右*vs.* 左)。这也验证了在极深的网络中残差学习的有效性。

最后，我们同样注意到，18层的平铺网络和残差网络的准确率很接近(Table 2)，但是ResNet 的收敛速度要快得多。(Fig. 4 右*vs.* 左)。如果网络"并不是特别深"(如18层)，现有的SGD能够很好的对平铺网络进行求解，而ResNet能够使优化得到更快的收敛。

恒等捷径连接*vs.* 映射捷径连接。 我们已经展示了无参数的恒等捷径连接是有助于训练的。接下来我们研究映射捷径连接(Eqn.(2))。在Table 3中，我们比较了三种方案：(A) 对增加的维度使用0填充，所有的捷径是无参数的(与Table 2 和Fig. 4 (右)相同)；(B) 对增加的维度使用映射捷径，其它使用恒等捷径；(C) 所有的都是映射捷径。

Table 3表明了三种方案的模型都比对于的平铺模型要好。B略好于A，我们认为这是因为A中的补零方法所获得的维数并没有残差学习。C略好于B，我们把这个归结于更多的（13个）映射捷径所引入的参数。而在A、B、C三个结果中细小的差距也表明了映射捷径连接对于解决退化问题并不是必需的。所以我们在本文接下来的内容中，为了减少复杂度和模型尺寸，并不使用方案C的模型。恒等捷径连接因其无额外复杂度而对以下介绍的瓶颈结构尤为重要。

深度瓶颈结构。 接下来我们介绍针对ImageNet数据集而涉及的更深的模型。考虑到训练时间的限制，我们将构建块修改成瓶颈的设计[4]. 对于每一个残差函数$\mathcal{F}$，

[4]更深的非瓶颈ResNets(*e.g.*, Fig. 5 左)同样随着深度增加，准确率得到了提高(如CIFAR-10上所示)，但是相比于瓶颈的ResNets来言并不是那么实惠。因此瓶颈设计主要是针对实用性的考虑。我们还注意到，在瓶颈设计中同样出现了退化问题的迹象。

method	top-1 err.	top-5 err.
VGG [40] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [40] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. **单一模型**在ImageNet验证集上的错误率(%)(除了† 是在验证集上的结果)。

method	top-5 err. (**test**)
VGG [40] (ILSVRC'14)	7.32
GoogLeNet [44] (ILSVRC'14)	6.66
VGG [40] (v5)	6.8
PReLU-net [13]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

Table 5. **组合模型**在ImageNet测试集上的top-5错误率。

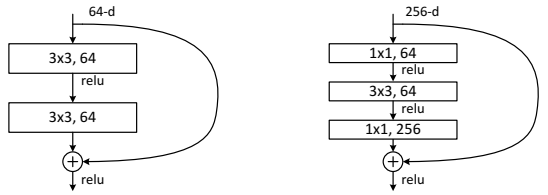

Figure 5. 对于ImageNet的一个更深的残差函数$\mathcal{F}$。左：对于ResNet-34的如Fig. 3描述的构建块。右：对于ResNet-50/101/152的"瓶颈"构建块。

我们使用了三个叠加层而不是两个(Fig. 5)。这三层分别是1×1、3×3 和1×1 的卷积，1×1 的层主要负责减少然后增加（恢复）维度，剩下的3×3的层来减少输入和输出的维度。Fig. 5展示了一个例子，这两种设计具有相近的时间复杂度。

无参数的恒等捷径对于瓶颈结构尤为重要。如果使用映射捷径连接来替代Fig. 5 (右)中的恒等捷径连接，将会发现时间复杂度和模型尺寸都会增加一倍，因为捷径连接了两个高维端，所以恒等捷径连接对于瓶颈设计是更加有效的。

50层ResNet： 我们将34层网络中2层的模块替换成3层的瓶颈模块，整个模型也就变成了50层的ResNet (Table 1)。我们使用方案B来做升维。整个模型含有38亿个FLOPs。

101层和152层ResNets： 我们使用更多的3层模块来构建101层和152层的ResNets (Table 1)。值得注意的是，虽然层的深度明显增加了，但是152层ResNet的计算复杂度(113亿个FLOPs)仍然比VGG-16(153 亿个FLOPs)和VGG-19(196亿个FLOPs）的小很多。

50/101/152层ResNets比34层ResNet的准确率要高得多(Table 3 和 4)。而且我们并没有观测到退化问题。所有的指标都证实了深度带来的好处。(Table 3 和 4)。

与最优秀方法的比较。 在Table 4中我们比较了目前最好的单模型结果。我们的34层ResNets取得了非常好的结果，152层的ResNet的单模型top-5验证错误率仅为4.49%，甚至比先前组合模型的结果还要好(Table 5)。我们将6个不同深度的ResNets合成一个组合模型(在提交结果时只用到2个152层的模型)。这在测试集上的top-5错误率仅为**3.57%** (Table 5)，这一项在ILSVRC 2015 上获得了第一名的成绩。

4.2. CIFAR-10 and Analysis

我们在包含5万张训练图像和1万张测试图像的10类CIFAR-10数据集[20]上进行了进一步的研究。我们在训练集上进行训练，在测试集上进行验证。我们关注的是极深模型的效果，而不是追求最好的结果，因此我们只使用简单的框架如下。

平铺网络和残差网络的框架如Fig. 3 (中/右)所示。网络的输入是32×32的减掉像素均值的图像。第一层是3×3的卷积层。然后我们使用$6n$个3×3的卷积层的堆叠，卷积层对应的特征图有三种尺寸：$\{32, 16, 8\}$，每个尺寸卷积层的数量为$2n$ 个，对应的滤波器数量分别为$\{16, 32, 64\}$。使用strde为2的卷积层进行下采样。在网络的最后是一个全局的平均pooling层和一个10类的包含softmax的全连接层。一共有$6n$+2个堆叠的权重层。具体的结构见下表：

output map size	32×32	16×16	8×8
# layers	1+2n	2n	2n
# filters	16	32	64

使用捷径连接的为3×3的卷积层对(共有$3n$ 个捷径连接)。在这个数据集上我们所有的模型都使用恒等捷径连接(*i.e.*, 方案A)，因此我们的残差模型和对应的平铺模型具有相同的深度、宽度和参数量。

权重的衰减设置为0.0001，动量为0.9，采用了[13]中的权值初始化以及BN [16]来训练网络，但是不使用Dropout，mini-batch的大小为128，模型在2 块GPU 上进行训练。学习率初始为0.1，在第32000和48000次迭代时将其除以10，总的迭代次数为64000，这是由45000/5000的训练集/验证集分配所决定的。我们在训练阶段遵循[24] 中的数据增强原则：在图像的每条边填充4个像素，然后在填充后的图像或者它的水平镜像上随机裁剪一个32×32 的图像块。在测试阶段，我们只使用原始32×32的图像进行评估。

我们比较了$n = \{3, 5, 7, 9\}$的情况，也就是20、32、44以及56层的网络。Fig. 6 (左) 展示了平铺网络的结果。深度平铺网络随着层数的加深，训练错误率也变大。这个现象与在ImageNet(Fig. 4, 左)和MNIST (见[41])上的结果很相似，表明了优化上的难度确实是一个很重要的问题。

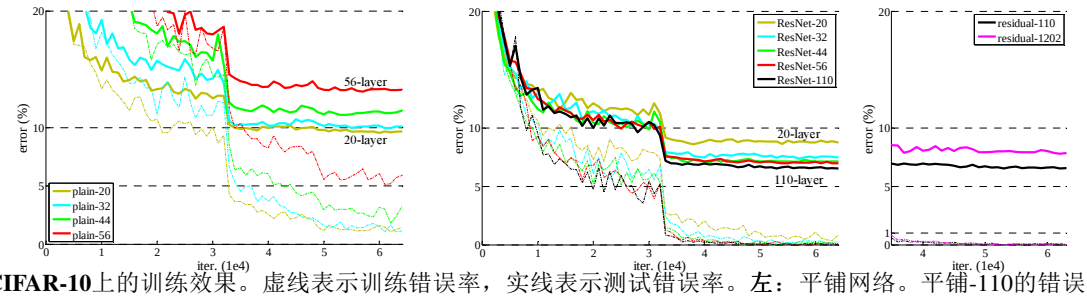

Figure 6. **CIFAR-10**上的训练效果。虚线表示训练错误率，实线表示测试错误率。**左**：平铺网络。平铺-110的错误率高达60%以上，因此并没有展示出来。**中**：ResNets. **右**：110层和1202层的ResNets。

method			error (%)
Maxout [10]			9.38
NIN [25]			8.81
DSN [24]			8.22
	# layers	# params	
FitNet [33]	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

Table 6. **CIFAR-10**测试集上的分类错误率。我们对数据进行了增强。如[42]所述，对于ResNet-110，我们运行了5遍，然后展示了"(均值±方差)最优"的结果。

Fig. 6 (中)展示了ResNets的效果。与ImageNet(Fig. 4, 右)中类似，我们的ResNets能够很好的克服优化难题，并且随着深度加深，准确率也得到了提升。

我们进一步探索了$n = 18$，也就是110层的ResNet。在这里，我们发现0.1的初始学习率有点太大而不能很好的收敛[5]。所以我们刚开始使用0.01的学习率，当训练错误率在80%以下(大约400次迭代)之后，再将学习率调回0.1继续训练。剩余的学习和之前的一致。110层的ResNets很好的收敛了(Fig. 6, 中)。它与其他的深层窄模型，如FitNet [33] 和Highway [41] (Table 6)相比，具有更少的参数，然而却达到了最好的结果(6.43%, Table 6)。

Analysis of Layer Responses. Fig. 7展示了层响应的标准方差(std)。响应是每一个3×3卷积层在BN之后、非线性层(ReLU/addition)之前的输出。对于ResNets，这个分析结果也揭示了残差函数响应强度的变化情况。Fig. 7表明了ResNets的响应比它对应的平铺网络的响应要小。这些结果也验证了我们的基本动机(Sec.3.1)，即残差函数比非残差函数更接近于0。

[5]当初始学习率设置为0.1时，模型在几个epochs之后开始收敛(错误率<90%)，但仍然能够达到很好的准确率。

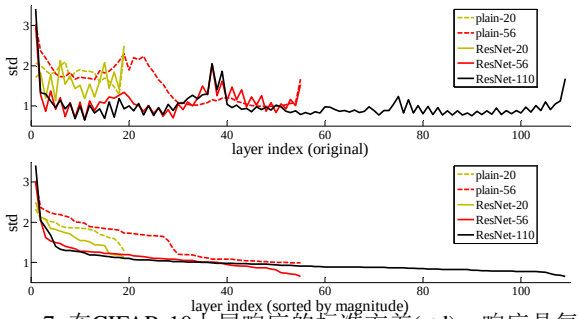

Figure 7. 在CIFAR-10上层响应的标准方差(std)。响应是每一个3×3卷积层的BN之后、非线性层之前的输出。**顶部**：层是按照它们原始的顺序。**底部**：响应按降序排列。

从Fig. 7中ResNet-20、56和110的结果，我们也注意到，越深的ResNet的响应幅度越小。当使用更多层时，ResNets中单个层对信号的改变越少。

Exploring Over 1000 layers. 我们探索了一个超过1000层的极其深的模型。我们设置$n = 200$，也就是1202层的网络模型，按照上述进行训练。我们的方法对10^3层的模型并不难优化，并且达到了<0.1%的训练错误率(Fig. 6, 右)，它的测试错误率也相当低(7.93%, Table 6)。

但是在这样一个极其深的模型上，仍然存在很多问题。1202层模型的测试结果比110层的结果要差，尽管它们的训练错误率差不多。我们认为这是过拟合导致的。这样一个1202层的模型对于小的数据集来说太大了(19.4M)。在这个数据集上曾应用了强力的正则化方法，如maxout [10] 或者dropout [14]，才获得了最好的结果([10, 38, 24, 33])。本文中，我们并没有使用maxout/dropout，设计上而言是为了仅通过增大或减少网络结构的深度来引入正则化，而且不用担心优化的难度。但是通过强力的正则化或许能够提高实验结果，我们会在今后进行研究。

4.3. Object Detection on PASCAL and MS COCO

我们的方法在其它识别任务上展现出了很好的泛化能力。Table 7 和 8 展示了在PASCAL VOC 2007和2012 [5] 以及COCO [25]上的目标检测结果。我们使用Faster R-CNN [31]作为检测方法。在这里，我们比较关注由ResNet-101 替换VGG-16 [40]所带来的性能

training data	07+12	07++12
test data	VOC 07 test	VOC 12 test
VGG-16	73.2	70.4
ResNet-101	**76.4**	**73.8**

Table 7. 在PASCAL VOC 2007/2012测试集上使用Faster R-CNN的目标检测mAP (%)。更多结果见Table 10 和11。

metric	mAP@.5	mAP@[.5, .95]
VGG-16	41.5	21.2
ResNet-101	**48.4**	**27.2**

Table 8. 在COCO 验证集上使用Faster R-CNN的目标检测mAP (%)。更多结果见Table 9。

提升。使用不同网络进行检测的实现(见附录)是一样的，所以检测结果只能得益于更好的网络。最值得注意的是，在COCO 数据集上，我们在COCO的标准指标(mAP@[.5, .95])上比先前的结果增加了6.0%，这相当于28%的相对提升。而这完全得益于所学到的表达。

基于深度残差网络，我们在ILSVRC & COCO 2015竞赛的ImageNet检测、ImageNet定位、COCO检测以及COCO分割上获得了第一名。具体的细节见附录。

References

[1] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[2] C. M. Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[3] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A multigrid tutorial*. SIAM, 2000.

[4] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: An evaluation of recent feature encoding methods. In *British Machine Vision Conference*, pages 76.1–76.12, 2011.

[5] M. Everingham. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[6] S. Gidaris and N. Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1134–1142, 2015.

[7] R. Girshick. Fast r-cnn. *Computer Science*, 2015.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *Computer Science*, pages 580–587, 2014.

[9] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[10] I. J. Goodfellow, D. Wardefarley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. *Computer Science*, pages 1319–1327, 2013.

[11] K. He and J. Sun. Convolutional neural networks at constrained time cost. In *Computer Vision and Pattern Recognition*, pages 5353–5360, 2014.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 346–361. Springer, 2014.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. pages 1026–1034, 2015.

[14] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *Computer Science*, 3(4):p谩gs. 212–223, 2012.

[15] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735, 1997.

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Computer Science*, 2015.

[17] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2011.

[18] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE transactions on pattern analysis and machine intelligence*, 34(9):1704–1716, 2012.

[19] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM International Conference on Multimedia*, pages 675–678, 2014.

[20] A. Krizhevsky. Learning multiple layers of features from tiny images. 2012.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105, 2012.

[22] Y. Lecun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[23] Y. LeCun, L. Bottou, and G. Orr. Efficient backprop in neural networks: Tricks of the trade (orr, g. and müller, k., eds.). *Lecture Notes in Computer Science*, 1524.

[24] C.-Y. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AISTATS*, volume 2, page 5, 2015.

[25] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[26] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[27] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932, 2014.

[28] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. *Proc Icml*, pages 807–814, 2015.

[29] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[30] T. Raiko, H. Valpola, and Y. Lecun. Deep learning made easier by linear transformations in perceptrons. 22:924–932, 2012.

[31] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[32] B. D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

[33] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. *Computer Science*, 2014.

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2014.

[35] A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[36] N. N. Schraudolph. *Centering Neural Network Gradient Factors*. Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, 1997.

[37] N. N. Schraudolph. Accelerated gradient descent by factor-centering decomposition. *Idsia*, 1999.

[38] ScienceOpen. Network in network. 2014.

[39] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *Eprint Arxiv*, 2013.

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *Computer Science*, 2014.

[41] R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *Computer Science*, 2015.

[42] R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. *Computer Science*, 2015.

[43] C. Szegedy, W. Liu, Y. Jia, and P. Sermanet. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[44] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2014.

[45] R. Szeliski. Fast surface interpolation using hierarchical basis functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):513–528, 1990.

[46] R. Szeliski. Locally adapted hierarchical basis preconditioning. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 1135–1143. ACM, 2006.

[47] T. Vatanen, T. Raiko, H. Valpola, and Y. LeCun. Pushing stochastic gradient towards second-order methods–backpropagation learning with transformations in nonlinearities. In *International Conference on Neural Information Processing*, pages 442–449. Springer, 2013.

[48] A. Vedaldi and B. Fulkerson. Vlfeat:an open and portable library of computer vision algorithms. In *International Conference on Multimedea 2010, Firenze, Italy, October*, pages 1469–1472, 2010.

[49] W. N. Venables and B. D. Ripley. *Modern applied statistics with S-PLUS*. Springer Science & Business Media, 2013.

[50] M. D. Zeiler and R. Fergus. *Visualizing and Understanding Convolutional Networks*. Springer International Publishing, 2014.

A. Object Detection Baselines

这部分介绍了我们基于Faster R-CNN [31]的检测方法。模型使用ImageNet分类模型初始化，然后在目标检测数据上进行微调。在ILSVRC & COCO 2015检测竞赛期间，我们已经尝试了ResNet-50/101的模型。

与[31]中的VGG-16不同的是，我们的ResNet没有隐含的fc层。我们是采取“Networks on Conv feature maps” (NoC) [31]的思路来解决这个问题。我们使用在图像上的stride不超过16个像素的层来计算整幅图的共享卷积特征图(*i.e.*, conv1、conv2_x、conv3_x以及conv4_x，一共有91个卷积层；Table 1)。我们把这些层看作是VGG-16 中类似的13个卷积层，通过这么做，ResNet和VGG-16 就具有相同总stride(16 pixels)的卷积特征图。这些层由一个RPN(生成300个proposals)[31]和一个Fast R-CNN网络[7]共享。在conv5_1之前执行RoI pooling [7]。在RoI pooling产生的特征上，conv5_x及以上的所有层用来提取每个区域的特征，起到了VGG-16中fc层的作用。最后的分类层替换成两个同级层(分类和边界回归[7])。

对于BN层的使用，在预训练之后，我们在ImageNet训练集上对每一层计算了BN统计量(均值和方差)。然后微调阶段，BN层固定。这样BN层就变成了具有固定偏移量和尺度的线性激活，BN统计量在微调阶段也不会更新。我们固定BN层主要是为了减少Faster R-CNN训练阶段的内存消耗。

PASCAL VOC

根据[7, 31]，针对PASCAL VOC 2007测试集，我们使用VOC 2007中的5千张训练验证图片以及VOC 2012中的1万6 千张训练验证图片来训练模型(“07+12”)。针对PASCAL VOC 2012测试集，我们使用VOC 2007中的1万张训练验证+测试图片以及VOC 2012中的1万6 千张训练验证图片来训练模型(“07++12”)。训练Faster R-CNN 的超参数与[31]中一致。Table 7中展示了实验结果。ResNet-101比VGG-16的mAP提高>3%。这完全得益于由ResNet学到的更好的特征。

MS COCO

MS COCO dataset [25]包含了80 个目标类别。我们在PASCAL VOC指标(mAP @ IoU = 0.5)及标准的COCO指标(mAP @ IoU = .5:.05:.95)上进行评估。我们使用训练集上的8万张图片进行训练，使用验证集上的4万张图片进行验证。我们在COCO上的检测模型也PASCAL VOC上的模型基本一致。我们使用8个GPU来训练COCO模型，因此RPN步骤的mini-batch大小为8(*i.e.*, 每个GPU上1张图片)，Fast R-CNN步骤的mini-batch为16。RPN步骤和Fast R-CNN步骤的训练在前24万次迭代上的学习率为0.001，接下来的8万次迭代上的学习率为0.0001。

Table 8展示了MS COCO验证集上的结果。ResNet-101在mAP@[.5, .95]指标上比over VGG-16增加了6%，这相当于是28% 的提升，完全得益于更好的模型学到的特征。值得注意的是，在mAP@[.5, .95]指标

上的绝对提高(6.0%)和在mAP@.5指标上的几乎一样大(6.9%)。这表明了一个更深的模型对于识别和定位都能提高效果。

B. Object Detection Improvements

为了文章的完整性，我们报告针对竞赛对模型的改进。这些改进都基于深层特征，因此应该得益于残差学习。

MS COCO

边界精细化。我们的边界精细化大致遵循[6]中的迭代定位。在Faster R-CNN中，最终的输出一个回归的边界(regressed box)，它和proposal边界是有区别的。因此推断，我们可以从回归的边界中池化出一种新的特征，从而能够获得一个新的分类得分和一个新的回归边界。我们将这300个新的预测和300个原始预测组合。在这个预测边界的集合上使用非极大值抑制(NMS)，IoU的阈值为0.3 [8]，然后进行边界投票(box voting) [6]。边界精细化能在mAP上提高2个百分点(Table 9)。

全局上下文。我们在Fast R-CNN步骤中结合了全局上下文。给定整幅图片的卷积特征图，我们使用全局空间金字塔池化(SPP) [12]来池化特征。(使用一个"单级"金字塔)，这相当于用整幅图片的边界框作为RoI来进行"RoI" pooling。这个池化得到的特征被传到post-RoI层就可以得到一个全局上下文特征。这个全局上下文特征与原始每个区域的特征连接起来，然后接一个分类层以及一个边界回归层。这个新的框架是端到端的。全局上下文能在mAP@.5上提高1个百分点(Table 9)。

多尺度测试。之前左右的结果都是由单一尺度的训练/测试所获得的[31]，在这个尺度下，图片的短边长为$s = 600$像素。多尺度训练/测试已经有过一些发展，如在[12, 7]中通过从特征金字塔中选择一个尺度以及在[31]中同时maxout层的使用。在我们的实验中，我们使用[**?**]中的方法进行多尺度测试；我们不进行多尺度训练是因为时间的限制。此外我们只在Fast R-CNN步骤中进行多尺度测试(而并不在RPN步骤中进行)。对于训练好的模型，我们在一个图像金字塔上计算卷积特征图，其中图像的短边长度为$s \in \{200, 400, 600, 800, 1000\}$。根据[31]，我们从金字塔中选择两个相邻的尺寸，并在这两个尺度的特征图上计算RoI pooling以及之后的层[31]，然后再通过maxout进行合并[31]。多尺度测试能在mAP上提高2个百分点(Table 9)。

使用验证数据。接下来我们使用8万+4万的训练验证集进行训练，使用2万的test-dev进行验证，test-dev并没有公开真值，验证结果是由验证服务器提供的。在这个设置下，mAP@.5达到了55.7%，mAP@[.5, .95]达到了34.9% (Table 9)。这只是我们单个模型的结果。

模型组合。在Faster R-CNN中，系统用来学习region proposal以及目标分类器，因此可以用模型的组合来对两个任务进行提示。我们使用一个组合来提取region proposal，然后这些proposal的集合再通过每个区域分类器的组合来处理。Table 9展示了我们使用3个网络来组合的结果。在test-dev上的mAP为59.0% 和37.4%。这个结果在COCO 2015检测任务上获得了第一名。

	val2	test
GoogLeNet [43] (ILSVRC'14)	-	43.9
our single model (ILSVRC'15)	60.5	58.8
our ensemble (ILSVRC'15)	**63.6**	**62.1**

Table 12. 在ImageNet检测数据集上的结果(mAP, %)。我们的检测系统是Table 9中改进的Faster R-CNN [31]，使用了ResNet-101网络。

LOC method	LOC network	testing	LOC error on GT CLS	classification network	top-5 LOC error on predicted CLS
VGG's [40]	VGG-16	1-crop	33.1 [40]		
RPN	ResNet-101	1-crop	13.3		
RPN	ResNet-101	dense	11.7		
RPN	ResNet-101	dense		ResNet-101	14.4
RPN+RCNN	ResNet-101	dense		ResNet-101	**10.6**
RPN+RCNN	ensemble	dense		ensemble	**8.9**

Table 13. 在ImageNet验证集上的定位错误率(%)。在"LOC error on GT class" ([40]) 列中，使用了类别真值。在"testing"列中，"1-crop"代表着224×224像素的一个center crop，"dense"代表着稠密(全卷积)和多尺度测试。

PASCAL VOC

基于以上模型，我们回到PASCAL VOC数据集。我们在PASCAL VOC数据集上微调COCO数据集上的单一模型(55.7% mAP@.5 如Table 9所示)。同样进行边界精细化、上下文以及多尺度测试的改进，然后我们在PASCAL VOC 2007和PASCAL VOC 2012上分别达到了85.6%(Table 10)和83.8%(Table 11)的mAP[6]。PASCAL VOC 2012上的结果比之前最好的结果高出了10个百分点[6]。

ImageNet Detection

ImageNet检测(DET)任务包含了200个目标类别。准确率指标是mAP@.5。对于ImageNet DET，我们的目标检测算法和Table 9中的MS COCO一致。网络在1000类的ImageNet分类数据集上预训练，然后在DET数据集上微调。根据[8]，我们将验证集分为两部分(val1/val2)。我们使用DET训练集合val1来微调模型，val2用来验证。我们并不使用其它的ILSVRC 2015数据。在DET测试集上，我们单一的ResNet-101达到了58.8%的mAP，三个模型的组合达到了62.1%的mAP(Table 12)。这个结果在*ILSVRC 2015*的*ImageNet*检测任务上获得了第一名，超过了第二名**8.5个百分点**。

training data	COCO train		COCO trainval	
test data	COCO val		COCO test-dev	
mAP	@.5	@[.5, .95]	@.5	@[.5, .95]
baseline Faster R-CNN (VGG-16)	41.5	21.2		
baseline Faster R-CNN (ResNet-101)	48.4	27.2		
+box refinement	49.9	29.9		
+context	51.1	30.0	53.3	32.2
+multi-scale testing	53.8	32.5	**55.7**	**34.9**
ensemble			**59.0**	**37.4**

Table 9. 在MS COCO上使用Faster R-CNN 和ResNet-101的改进。

system	net	data	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
baseline	VGG-16	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
baseline	ResNet-101	07+12	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	**89.8**	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72.0
baseline+++	ResNet-101	COCO+07+12	**85.6**	**90.0**	**89.6**	**87.8**	**80.8**	**76.1**	**89.9**	**89.9**	89.6	**75.5**	**90.0**	**80.7**	**89.6**	**90.3**	**89.1**	**88.7**	**65.4**	**88.1**	**85.6**	**89.0**	**86.8**

Table 10. 在PASCAL VOC 2007测试集上的检测结果。其中baseline是Faster R-CNN系统，“baseline+++”包括了Table 9中的边界精细化、上下文和多尺度测试。

system	net	data	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
baseline	VGG-16	07++12	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
baseline	ResNet-101	07++12	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	93.2	48.6	80.4	59.0	92.1	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
baseline+++	ResNet-101	COCO+07++12	**83.8**	**92.1**	**88.4**	**84.8**	**75.9**	**71.4**	**86.3**	**87.8**	**94.2**	**66.8**	**89.4**	**69.2**	**93.9**	**91.9**	**90.9**	**89.6**	**67.9**	**88.2**	**76.8**	**90.3**	**80.0**

Table 11. PASCAL VOC 2012测试集上的检测结果(`http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?challengeid=11&compid=4`)。其中baseline是Faster R-CNN 系统。“baseline+++” 包括了Table 9中的边界精细化、上下文和多尺度测试。

C. ImageNet Localization

ImageNet定位(LOC)任务[34]需要对目标进行分类和定位。根据[39, 40]，我们假设先使用图像级分类器来预测图像的类别，然后根据所预测的类别使用定位算法来预测边界框。我们采取“per-class regression” (PCR) [39, 40]的策略来对每一类学习一个边界框。我们先针对ImageNet分类对模型进行预训练，然后针对定位对模型进行微调。我们在1000类的ImageNet训练集上训练我们的模型。

我们的定位算法是在[31]的RPN框架的基础上进行了一些修改。与[31]中的类别无关(category-agnostic)不同的是，我们用来定位的RPN是以per-class的形式设计的。[31]中的RPN末端连接了两个同级的1×1卷积层，一个用来做二分类(*cls*)，另一个用来做边界回归。与[31]中不同的是，我们模型中的*cls*层和*reg*层都是per-class的形式。具体来说，*cls*层具有1000维的输出，每一维都是用来预测是否属于该类的二元逻辑回归；*reg*层具有1000×4维的输出，它由1000 类的边界回归组成。正如[31]，我们是根据图像中每一个位置的多个平移不变的“anchor”来进行边界框回归的。

与我们针对ImageNet分类的训练(Sec. 3.4)一样，我们随机的从增强的图像中采集224×224的crops。在微调过程中mini-batch 的大小为256。为了避免负样本在样本集中比例过大，从每幅图像中随机采集8个anchors，并保持正负anchors的比例为1:1 [31]。网络在图像上使用全卷积来进行测试。

[6] `http://host.robots.ox.ac.uk:8080/anonymous/3OJ4OJ.html`, submitted on 2015-11-26.

method	top-5 localization err	
	val	test
OverFeat [39] (ILSVRC'13)	30.0	29.9
GoogLeNet [44] (ILSVRC'14)	-	26.7
VGG [40] (ILSVRC'14)	26.9	25.3
ours (ILSVRC'15)	**8.9**	**9.0**

Table 14. 在ImageNet数据集上与最好方法的定位错误率(%)的比较。

Table 13比较了一些方法的定位结果。根据[40]，我们先用类真值作为类预测结果来进行“oracle”测试。VGG的文章中[40]报告了当使用类真值时他们的center-crop错误率为33.1% (Table 13)。在相同的设置下，使用ResNet-101的RPN方法将center-crop错误率降到了13.3%。这展示了我们框架的优良性能。在稠密(全卷积)和多尺度测试上，当使用类真值时，ResNet-101的错误率为11.7%。而使用ResNet-101的类预测结果(top-5分类错误率为4.6%，Table 4)时，top-5的定位错误率为14.4%。

以上的结果仅仅是基于对Faster R-CNN [31]中RPN的改进，我们还可以利用Faster R-CNN中的检测网络*detection network* (Fast R-CNN [7])来改进结果。但是我们注意到，在这个数据集上，每张图像通常只包含一个占主导地位的目标，而proposal regions之

间可能具有很高的重叠率，因此导致RoI-pooled的特征很相似。因此Fast R-CNN [7]的image-centric训练生成了差别不大的样本，因此并不是理想的随机训练。故在我们的实验中，使用了RoI-centric的R-CNN [8]来替代Fast R-CNN。

R-CNN的实现如下。我们在训练图像上训练per-class的RPN来预测类真值的边界框。这些预测框对于class-dependent的proposals起到了很重要的作用。对于每一张训练图像，得分最高的200个proposals被提取出来作为R-CNN分类器的训练样本。从一个proposal中裁剪出图像区域，并warp成224×224像素，然后传入R-CNN [8]的分类网络中。网络的输出包含两个同级fc层，一个为*cls*，另一个为*reg*，同样是per-class的形式。R-CNN网络以RoI-centric在训练集上的微调，mini-batch的大小为256。测试阶段，RPN对每一个预测的类生成了得分最高的proposals，然后使用R-CNN对这些proposals的得分和边界位置进行更新。

这个方法将top-5的定位错误率降到了10.6% (Table 13)。这是我们单个模型在验证集上的结果。在分类和定位上使用组合模型时，我们在测试集上的的top-5定位错误率仅为9.0%。这个结果明显比ILSVRC 14的结果好很多(Table 14)，这相当于减少了64%的相对错误率。这个结果在*ILSVRC 2015* 的*ImageNet*定位任务中获得了第一名的成绩。

译者后记

本译文仅供学习交流。如果你英语过了6级、且是研究生或博士、熟练DL或CV、NLP等、爱好翻译，欢迎加入我们，微博私信：@研究者July

二零一七年二月二十五日