Rethinking the Inception Architecture for Computer Vision

Christian Szegedy
Google Inc.
szegedy@google.com

Vincent Vanhoucke
vanhoucke@google.com

Sergey Ioffe
sioffe@google.com

Jonathon Shlens
shlens@google.com

Zbigniew Wojna
University College London
zbigniewwojna@gmail.com

Abstract

提示：该*PDF*由*SpeedPaper*生成，版权归原文作者所有。翻译内容仅供参考，请仔细鉴别并以原文为准。

查看更多论文翻译与复现代码：*https://github.com/hanknewbird/SpeedPaper*

卷积网络是当今计算机视觉众多任务的尖端解决方案的核心。自*2014*年起，非常深层的卷积网络开始成为主流，极大地提升了各种基准测试的性能。尽管增加模型大小和计算成本往往会为大多数任务带来即时的质量提升（只要提供足够的标记数据进行训练），但计算效率和低参数数量仍然是移动视觉和大数据场景等各种使用案例的关键因素。在这里，我们探索了通过适当分解卷积和积极的正则化等方式扩展网络的方法，旨在尽可能高效地利用额外的计算。我们在*ILSVRC 2012*分类挑战的验证集上对我们的方法进行基准测试，展示了明显超越当下尖端技术的进展：使用具有*50*亿次乘加操作的网络进行“单帧”评估，仅使用不到*2500*万参数，使得*top-1*误差为*21.2%*，*top-5*误差为*5.6%*。通过使用*4*个模型的集成和多裁剪评估，我们报告了*top-5*错误率为*3.5%*，*top-1*错误率为*17.3%*。

1. Introduction

自2012年 ImageNet 比赛[16]中由Krizhevsky等人[9]获胜以来，他们的网络“AlexNet”已成功应用于更多类型的计算机视觉任务，例如目标检测[5]、分割[12]、人体姿势估计[22]、视频分类[8]、目标跟踪[23]和超分辨率[3]。

这些成功激发了一个新的研究方向，专注于寻找性能更高的卷积神经网络。从2014年开始，通过使用更深更宽的网络，网络架构的质量显著提高。VGGNet[18]和 GoogLeNet[20] 在2014年ILSVRC分类挑战中表现出类似的高性能。一个有趣的观察是，分类性能的提升往往会转化为广泛应用领域的质量提升。这意味着深度卷积架构的结构改进可以用于提高越来越依赖于高质量学习视觉特征的大多数其他计算机视觉任务的性能。此外，网络质量的改进导致了卷积网络在一些情况下出现在新的应用领域，其中 AlexNet 的特征无法与手工设计的解决方案竞争，例如在检测中的提案生成[4]。

尽管 VGGNet [18] 具有架构简单的吸引力，但这也带来了很高的成本：评估网络需要大量计算。另一方面，GoogLeNet 的 Inception 架构[20] 也被设计为在对内存和计算预算有严格限制的情况下表现出色。例如，GoogLeNet 只使用了500万个参数，相对于其前身 AlexNet 的6000万参数，减少了12倍。此外，VGGNet 使用的参数约为 AlexNet 的3倍。

Inception 的计算成本也远低于 VGGNet 或其性能更高的后继版本[6]。这使得在需要以合理的成本处理大量数据或内存或计算能力本质上受限的情况下（例如在移动视觉设置中）可以利用 Inception 网络。当然，可以通过应用专门的解决方案来针对内存使用压

缩[2][15]，或通过计算技巧优化执行某些操作来减轻部分问题。然而，这些方法会增加额外的复杂性。此外，这些方法也可以用于优化 Inception 架构，再次扩大效率差距。

然而，Inception 架构的复杂性使得对网络进行更改更加困难。如果天真地扩展架构，可能会立即丢失大部分计算优势。此外，文章 [20] 并未清楚描述导致 GoogLeNet 架构各种设计决策的因素。这使得在保持效率的同时将其适应新用例更加困难。例如，如果有必要增加某个 Inception 风格模型的容量，简单地将所有滤波器组大小加倍会导致计算成本和参数数量增加4倍。在许多实际情况下，尤其是如果关联收益不大，这可能是禁止或不合理的。在本文中，我们首先描述了一些证明对于有效扩展卷积网络是有用的一些一般原则和优化思想。虽然我们的原则不限于 Inception 类型的网络，但在那种背景下更容易观察到，因为 Inception 风格构建块的通用结构足够灵活，可以自然地融入这些约束。这是通过慷慨使用维度减少和 Inception 模块的并行结构实现的，这允许在附近组件上减轻结构变化的影响。但是，任何这样做都需要谨慎，因为必须遵守一些指导原则以保持模型的高质量。

2. General Design Principles

在这里，我们将根据对使用卷积网络的各种架构选择进行大规模实验得出的一些设计原则进行描述。目前，下面的原则的实用性是推测性的，需要额外的未来实验证据来评估其准确性和适用领域。然而，严重偏离这些原则往往导致网络质量的恶化，并且修复检测到这些偏离的情况通常会导致总体上改进的架构。

1. 避免表征瓶颈，特别是在网络的早期阶段。前馈网络可以通过从输入层到分类器或回归器的无环图表示。这定义了信息流的明确方向。对于任何将输入与输出分隔开的切割，可以访问通过该切割传递的信息量。应避免出现极端压缩的瓶颈。一般来说，表示的大小应该在从输入到输出逐渐减小，最后达到用于当前任务的最终表示。理论上，信息内容不能仅通过表示的维度来评估，因为这种方法会忽略诸如相关结构之类的重要因素；维度仅提供信息内容的粗略估计。

2. 更高维度的表示更容易在网络中进行局部处理。在卷积网络中增加每个切片的激活能够产生更加解耦的特征。由此产生的网络将训练更快。

3. 空间聚合可以在较低维嵌入中进行，而几乎不会损失任何表征能力。例如，在执行更分散的（例如3×3）卷积之前，可以在空间聚合之前减少输入表征的维度，而不会期望出现严重的不利影响。我们假设这样做的原因是相邻单元之间的强相关性导致在维度降低时在空间聚合上使用时信息损失更少。鉴于这些信号应该很容易压缩，维度降低甚至促进更快的学习。

4. 平衡网络的宽度和深度。通过平衡每个阶段的滤波器数量和网络的深度，可以实现网络的最佳性能。增加网络的宽度和深度都可以促进生成更高质量的网络。然而，当两者同时增加时，对于相同计算量的最佳改进才能实现。因此，计算预算应在网络的深度和宽度之间以平衡的方式分配。

尽管这些原则可能是合理的，但要直接将它们用于改善网络质量并不简单。关键是只在模糊的情况下谨慎使用这些原则。

3. Factorizing Convolutions with Large Filter Size

GoogLeNet网络 [20] 的许多初始收益大部分源自对维度降维的极其慷慨的使用。这可以被视为以一种计算高效的方式对卷积进行因式分解的特例。例如，考虑一个1×1卷积层后跟一个3×3卷积层的情况。在视觉网络中，预期附近激活的输出高度相关。因此，我们可以期待在聚合之前减少它们的激活，并且这应该会导致类似表现力的局部表示。

在这里，我们探讨了在各种设置中对卷积进行因式分解的其他方式，特别是为了提高解决方案的计算效率。由于Inception网络是完全卷积的，每个权重对应于每个激活的一次乘法。因此，任何计算成本的降低都会导致参数数量的减少。这意味着通过适当的因式分解，我们最终可以得到更多解耦参数，从而实现更快的训练。此外，我们可以利用计算和内存的节省来增加网络的滤波器组大小，同时保持我们在单台计算机上训练每个模型副本的能力。

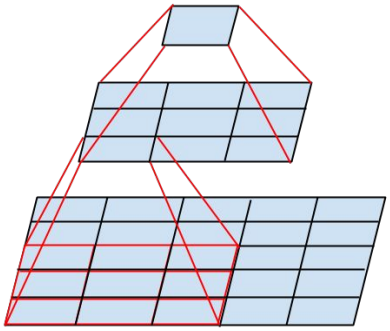

图 1. 替换 5×5 卷积的迷你网络。

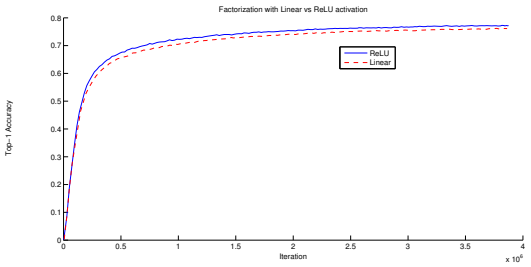

图 2. 在两个Inception模型之间进行了几个控制实验，其中一个使用线性+ReLU层的因式分解，另一个使用了两个ReLU层。经过386万次操作后，前者在验证集上的top-1准确率为76.2%，而后者达到了77.2%。

3.1. Factorization into smaller convolutions

使用较大的空间滤波器（例如5×5或7×7）进行卷积在计算方面往往是不成比例的昂贵。例如，一个包含n个滤波器的5×5卷积在含有m个滤波器的网格上的计算成本是3×3卷积的2.78倍。当然，5×5滤波器可以捕捉较早层中距离较远的单元激活之间信号的依赖关系，因此减小滤波器的几何尺寸会带来表达能力的巨大代价。然而，我们可以探讨是否可以用参数更少的多层网络来替代5×5卷积，同时保持相同的输入尺寸和输出深度。如果我们将视线聚焦在5×5卷积的计算图中，我们会看到每个输出看起来像一个小的全连接网络在其输入上滑动5×5的瓦片（参见图 1）。由于我们正在构建一个视觉网络，因此再次利用平移不变性并将完全连接的组件替换为两层卷积架构似乎是自然的选择：第一层是一个 3×3 卷积，第二层是在第一层的 3×3 输出网格上方的一个全连接层（参见图 1）。将这个小网络滑动到输入激活网格上相当于用两层 3×3 卷积替换了 5×5 卷积（可参考图 4 和 5 的对比）。

通过共享相邻瓦片之间的权重，这样的设置明显减少了参数数量。为了分析预期的计算成本节约，我们将做出几个适用于典型情况的简化假设：我们可以假设 $n = \alpha m$，即我们希望通过一个常数alpha因子改变单位的激活数量。由于 5×5 卷积是聚合的，通常情况下alpha略大于1（在GoogLeNet中大约为1.5）。为了简化我们的估计，我们选择 $\alpha = 1$（没有扩张）。如果我们简单地滑动一个网络而不重用邻近网格瓦片之间的计算，我们将增加计算成本。将这个网络表示为两个可以重复使用邻近瓦片之间激活的 3×3 卷积层。通过这种方式，我们最终实现了计算量的 $\frac{9+9}{25}\times$ 减少，这导致了 28% 的相对收益。这个因式分解也适用于参数数量，因为每个参数在每个单位激活的计算中只被使用一次。然而，这个设置引发了两个一般性问题：这种替代是否导致了表达能力的损失？如果我们的主要目标是因式分解计算的线性部分，那么不会建议在第一层中保留线性激活吗？我们进行了几个对照实验（例如，参见图 2），并且在因式分解的所有阶段中使用线性激活总是逊色于使用修正线性单元。我们认为这种收益归因于网络可以学习的增强变化空间，特别是当我们批量标准化输出激活时。当使用线性激活进行维度缩减时，也可以看到类似的效果。

3.2. Spatial Factorization into Asymmetric Convolutions

以上结果表明，使用大于3×3的卷积滤波器可能并不会普遍有用，因为它们总是可以被简化为一系列3×3的卷积层。但我们可以问一个问题，是否应该将它们分解为更小的卷积，例如2×2。然而，事实证明，通过使用不对称卷积，如$n \times 1$，甚至可以做得更好。例如，使用一个3×1的卷积接着一个1×3的卷积，相当于在具有与3×3卷积相同感受野的情况下滑动一个具有两层网络的解决方案（见图 3）。然而，如果输入和输出的滤波器数量相等，相同数量的输出滤波器的情况下，两层解决方案要比一个3×3卷积便宜33%。相比之下，将一个 3×3 的卷积因式分解为两个 2×2 的卷积，仅节省了 11% 的计算量。

在理论上，我们甚至可以认为可以用一个$1 \times n$卷

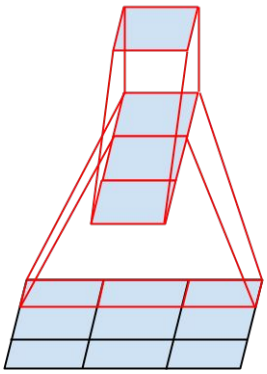

图 3. 使用迷你网络替换3 × 3卷积。该网络的较低层包括一个3 × 1卷积，具有3个输出单元。

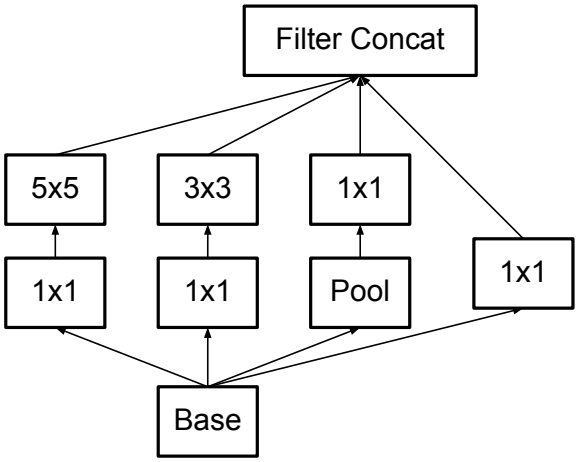

图 4. 在[20]中描述的初始模块。

积和一个$n \times 1$卷积替代任何$n \times n$卷积，并且随着n的增长，计算成本节约会显著增加（见图6）。在实践中，我们发现采用这种分解方法在初期层面效果不佳，但在中等网格大小（在$m \times m$特征图上，其中m介于12和20之间）上取得了非常好的结果。在这个层面上，通过使用1×7卷积接着7×1卷积可以取得非常好的结果。

4. Utility of Auxiliary Classifiers

[20] 提出了辅助分类器的概念，以改善非常深层网络的收敛性。最初的动机是将有用的梯度推送到较

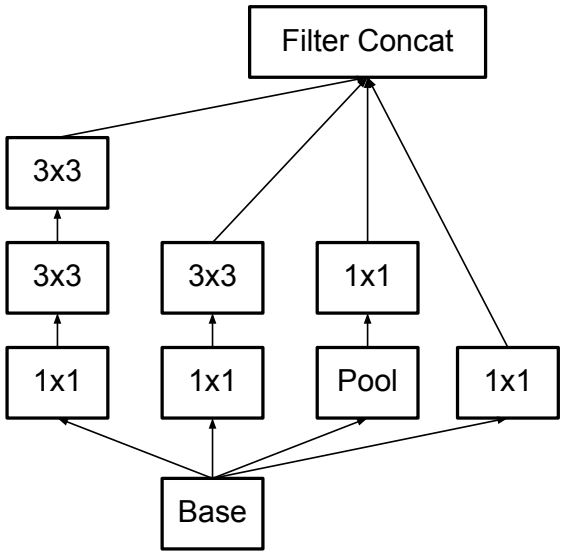

图 5. 在Inception模块中，每个5 × 5卷积被两个3 × 3卷积所取代，正如3原则 2节所建议的。

低层，使它们立即有用，并通过对抗非常深层网络中的梯度消失问题，在训练过程中改善收敛性。此外，Lee等人[11] 认为辅助分类器促进了更稳定的学习和更好的收敛性。 有趣的是，我们发现辅助分类器并没有在训练的早期阶段带来改善的收敛：在两种模型都达到高准确率之前，带有和不带有辅助头部的网络的训练进展几乎是相同的。在训练接近结束时，带有辅助分支的网络开始超越没有任何辅助分支的网络的准确性，并达到稍微更高的水平。

此外，[20]在网络的不同阶段使用了两个辅助头部。去除较低的辅助分支并没有对网络最终质量产生任何不利影响。结合前面一段的观察来看，这意味着[20]最初的假设，即这些分支有助于演变低级特征，很可能是错误的。相反，我们认为辅助分类器起到了正则化的作用。这一观点得到了支持，因为如果网络的主分类器具有批量归一化[7]或具有一个丢弃层，则网络的主分类器表现更好。这也为批量归一化起到正则化作用的猜测提供了一点弱支持证据。

5. Efficient Grid Size Reduction

传统上，卷积网络使用一些池化操作来减小特征映射的网格大小。为了避免表征瓶颈，在应用最大或平均池化之前，扩展了网络滤波器的激活维度。例如，

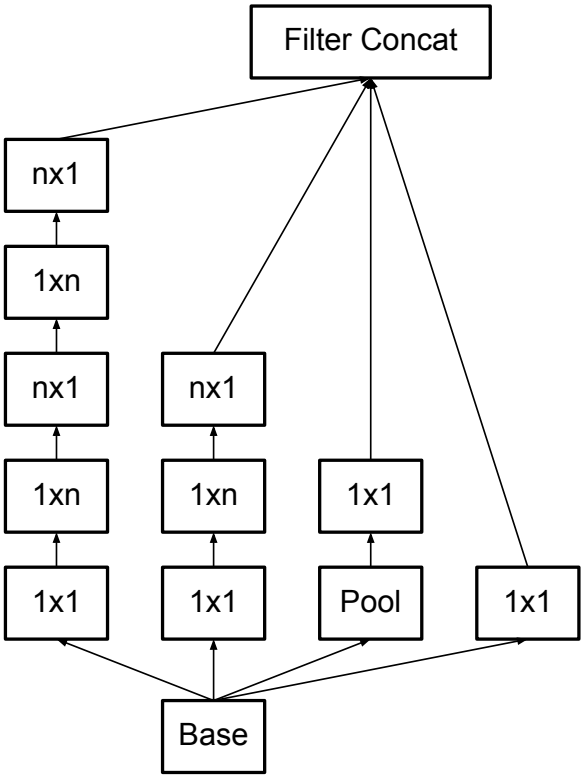

图 6. 在将 $n \times n$ 卷积分解后的初始模块。在我们提出的架构中，我们选择 $n = 7$ 用于 17×17 网格。（滤波器尺寸是根据原则 3 选定的）

.

在一个 $d \times d$ 网格上使用 k 个滤波器，如果我们想要得到一个 $\frac{d}{2} \times \frac{d}{2}$ 网格，有 $2k$ 个滤波器，我们首先需要计算一个步幅为 1 的具有 $2k$ 个滤波器的卷积，然后应用额外的池化步骤。这意味着整体计算成本主要由在较大网格上使用 $2d^2k^2$ 次运算的昂贵卷积支配。一种可能的解决方案是切换到带卷积的池化，从而减少四分之一的计算成本，但是这会导致表征瓶颈，因为表示的整体维度降至 $(\frac{d}{2})^2k$，导致网络表达能力较弱。不过，我们建议采用另一种变体进一步降低计算成本，同时消除表征瓶颈。我们可以使用两个并行的步幅为 2 的块：P 和 C。P 是一个池化层（可以是平均池化或最大池化）激活，它们的滤波器组是串联的，如图10所示。

6. Inception-v2

在这里，我们连接了上面的要点并提出了一个在ILSVRC 2012分类基准上性能提升的新架构。我们

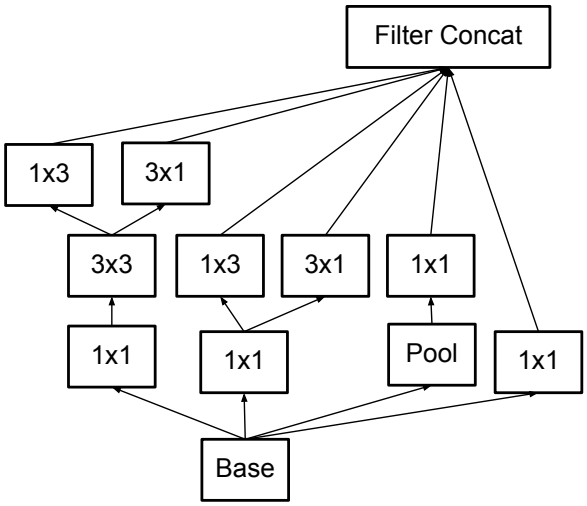

图 7. 在初始模块中扩展了滤波器组输出。这种架构在最粗糙的（8×8）网格上使用，以促进高维表示，如第 2 节中原则 2 所建议的。我们仅在最粗糙的网格上使用此解决方案，因为在那里，通过 1×1 卷积进行本地处理的比例相对于空间聚合增加，生产高维稀疏表达最为关键。

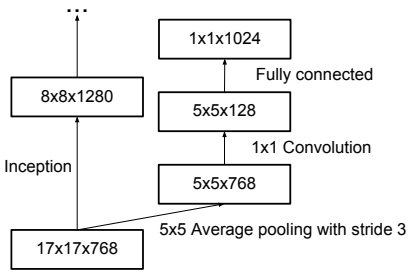

图 8. 在最后的17×17层之上使用辅助分类器。在侧头部的层上应用批归一化[7]会使top-1精度提升0.4%。下方轴显示进行的迭代次数，每次使用批量大小为32。

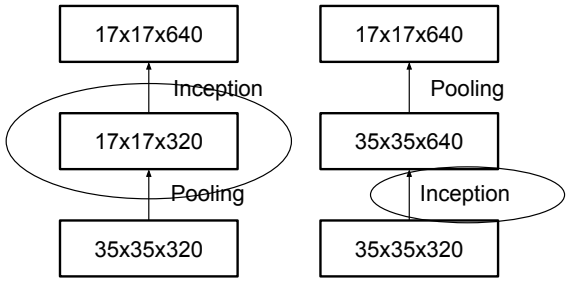

图 9. 两种减小网格尺寸的替代方法。左侧的解决方案违反了第 1原则，即不引入表象瓶颈(见第 2节)。右侧的版本在计算上更昂贵，耗用了3倍的计算资源。

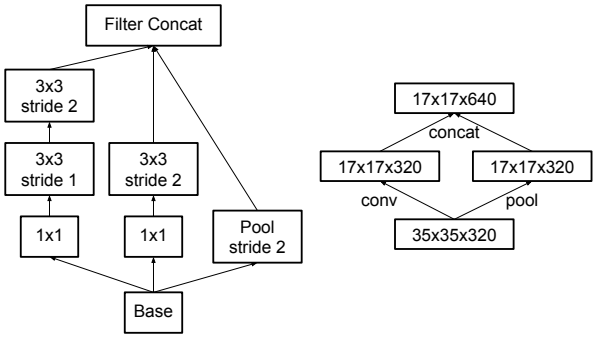

图 10. Inception模块减小了网格尺寸，同时扩大了滤波器组。它既廉价又避免了表示瓶颈，正如原则1所建议的。右侧图表展示了相同的解决方案，但从网格尺寸的角度而非操作的角度。

type	**patch size/stride** or remarks	**input size**
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

表 1. 提出的网络架构概述。每个模块的输出大小是下一个模块的输入大小。我们使用的减少技术的变体如图 10所示，用于在可能的情况下减少Inception块之间的网格大小。我们标记了带有0填充的卷积，用于保持网格大小。在那些不减少网格大小的Inception模块中也使用0填充。所有其他层不使用填充。各种滤波器组大小是根据第 4节的原则 2选择的。

网络的布局如表 1 中所示。请注意，我们已经将传统的7 × 7卷积因式分解为基于与第 3.1节中描述的相同思想的三个3 × 3卷积。对于网络的Inception部分，我们在35 × 35处有3个传统的Inception模块，每个模块有288个滤波器。通过使用第 5节描述的网格降维技术，将其减少到一个带有768个滤波器的17 × 17网格。接着，采用如图 5所示的5个实例的因式分解Inception模块。通过如图 10所示的网格降维技术，将其减少到一个8 × 8 × 1280的网格。在最粗糙的8 × 8级别，我们有两个Inception模块，如图 6所示，每个瓦片的连接输出滤波器大小为2048。网络的详细结构，包括Inception模块内部滤波器组大小，在补充材料中给出，该材料包含在此提交的tar文件的`model.txt`中。然而，我们观察到只要遵守第 2节的原则，网络的质量相对稳定。尽管我们的网络深度为42层，但我们的计算成本仅比GoogLeNet高约2.5，仍然比VGGNet更高效。

7. Model Regularization via Label Smoothing

我们在这里提出了一种机制，通过在训练过程中估算标签丢失的边际效应来规范分类器层。

对于每个训练示例 x，我们的模型计算每个标签 $k \in \{1 \dots K\}$ 的概率为：$p(k|x) = \frac{\exp(z_k)}{\sum_{i=1}^{K} \exp(z_i)}$。这里，$z_i$ 是未归一化的对数概率，或称为 *logits*。考虑关于该训练示例的标签的真实分布 $q(k|x)$，使其归一化，使得 $\sum_k q(k|x) = 1$。为简洁起见，我们省略了 p 和 q 对示例 x 的依赖。我们定义示例的损失为交叉熵：$\ell = -\sum_{k=1}^{K} \log(p(k))q(k)$。最小化这个损失等价于最大化标签的期望对数似然，其中标签是根据其真实分布 $q(k)$ 进行选择的。交叉熵损失对 logits z_k 可微分，因此可用于深度模型的梯度训练。梯度具有相当简单的形式：$\frac{\partial \ell}{\partial z_k} = p(k) - q(k)$，其值介于 -1 和 1 之间。

考虑仅存在一个真实标签 y 的情况，因此对所有 $k \neq y$ 有 $q(k) = 0$。在这种情况下，最小化交叉熵等价于最大化正确标签的对数似然。对于具体的带有标签 y 的示例 x，当 $q(k) = \delta_{k,y}$ 时，对数似然达到最大值，其中 $\delta_{k,y}$ 是狄拉克 δ 函数，对于 $k = y$ 时等于 1，其他情况下为 0。这个最大值不可达对于有限的 z_k，但如果对于所有 $k \neq y$ 有 $z_y \gg z_k$，则会接近此最大值，即，如果与真实标签对应的 logit 远大于所有其他 logit。但是，这可能会导致两个问题。首先，可能会导致过拟合：如果模型学会为每个训练示例分配完整的概率给真实标签，就不能保证泛化能力。其次，它鼓励最大 logit 与所有其他 logit 之间的差异变大，再加上有界梯度 $\frac{\partial \ell}{\partial z_k}$，这降低了模型的适应能力。直觉上，这是因为模型对其预测过于自信。

我们提出了一种机制来鼓励模型变得不那么自信。如果目标是最大化训练标签的对数似然，这可能不是理想的，但它确实规范了模型并使其更易适应。这种

方法非常简单。考虑一个标签分布 $u(k)$，独立于训练示例 x，以及一个平滑参数 ϵ。对于具有真实标签 y 的训练示例，我们将标签分布 $q(k|x) = \delta_{k,y}$ 替换为

$$q'(k|x) = (1-\epsilon)\delta_{k,y} + \epsilon u(k)$$

这是原始地面真实分布 $q(k|x)$ 和固定分布 $u(k)$ 的混合，其权重分别为 $1-\epsilon$ 和 ϵ。可以将其视为通过以下步骤获得的标签 k 的分布：首先，将其设置为地面真实标签 $k = y$；然后，以概率 ϵ，用从分布 $u(k)$ 抽样得到的样本替换 k。我们建议使用标签的先验分布作为 $u(k)$。在我们的实验中，我们使用均匀分布 $u(k) = 1/K$，这样

$$q'(k) = (1-\epsilon)\delta_{k,y} + \frac{\epsilon}{K}.$$

我们将真实标签分布的这种变化称之为标签平滑正则化*(label-smoothing regularization)*，或简称*LSR*。

需要注意的是，*LSR* 达到了防止最大 *logit* 远远大于其他所有的 *logit* 的期望目标。实际上，如果这种情况发生，那么单个 $q(k)$ 会接近于 1，而其他所有的 $q(k)$ 则会接近 0。这将导致由于与 $q'(k)$ 不同的是所有的 $q'(k)$ 都有一个正的下界，因此会产生一个大的 *cross-entropy*。

LSR 的另一种解释可以通过考虑交叉熵来获得：

$$H(q', p) = -\sum_{k=1}^{K} \log p(k) q'(k) = (1-\epsilon)H(q,p) + \epsilon H(u,p)$$

因此，*LSR*相当于将单个交叉熵损失函数$H(q,p)$替换为一对这样的损失函数$H(q,p)$和$H(u,p)$。第二个损失函数惩罚了预测标签分布p与先验分布u之间的偏差，相对权重为$\frac{\epsilon}{1-\epsilon}$。请注意，由于$H(u,p) = D_{KL}(u\|p) + H(u)$且$H(u)$固定，此偏差也可以等价地通过*KL*散度来捕捉。当u为均匀分布时，$H(u,p)$衡量了预测分布p与均匀分布的不相似程度，这也可以通过负熵$-H(p)$来衡量（但不是等价的）。我们尚未尝试过这种方法。

在我们针对$K = 1000$类别的*ImageNet*实验中，我们使用了$u(k) = 1/1000$和$\epsilon = 0.1$。对于*ILSVRC 2012*，我们发现在*top-1*错误率和*top-5*错误率上都实现了大约0.2%的绝对改进（参见表*3*）。

8. Training Methodology

我们使用 *TensorFlow [1]* 分布式机器学习系统，利用随机梯度训练了我们的网络模型，每个模型使用 50 个副本，在 *NVidia Kepler GPU* 上运行，批量大小为 32，训练 100 个周期。之前的实验使用了动量 *[19]*，衰减率为 0.9，而我们最佳的模型是使用了 *RMSProp [21]*，衰减率为 0.9，$\epsilon = 1.0$。我们使用学习率为 0.045，每两个周期衰减一次，指数速率为 0.94。此外，发现使用梯度修剪 *[14]*，阈值为 2.0，对稳定训练很有帮助。模型评估是通过随时间计算的参数的滑动平均值来执行的。

9. Performance on Lower Resolution Input

典型的视觉网络用例是用于检测后的后分类，例如在*Multibox [4]* 的上下文中。这包括分析包含单个对象及其一些背景的相对小的图像区域。任务是决定图像区域中心部分是否对应某个对象，并确定对象的类别（如果是对象）。挑战在于对象往往是相对较小和低分辨率的。这引发了对如何正确处理较低分辨率输入的问题。

通常被认为，采用更高分辨率感受野的模型往往会显着提高识别性能。然而，重要的是要区分第一层感受野分辨率增加的影响和更大模型容量和计算效果的影响。如果仅仅改变输入的分辨率而没有对模型进行进一步调整，那么我们最终会使用计算成本更低的模型来解决更加困难的任务。当然，自然而然的是，这些解决方案已经因为减少的计算工作而失去了优势。为了进行准确评估，模型需要分析模糊的线索以便“幻想”出细节。这在计算上是昂贵的。因此，问题依然是：如果计算工作维持不变，更高的输入分辨率能带来多大帮助。确保持续工作的一个简单方法是在低分辨率输入的情况下减少前两层的步幅，或者干脆删除网络的第一个池化层。

为此，我们进行了以下三个实验：

1. 第一层后的步幅为*2*，并进行最大池化的299×299感受野。
2. 第一层后使用步幅为*1*的151×151感受野和最大池化。
3. 第一层后不经过池化，使用步幅为*1*的79×79感受野。

三个网络的计算成本几乎相同。尽管第三个网络稍微便宜一些，但池化层的成本很小，与网络总成本相比

Receptive Field Size	Top-1 Accuracy (single frame)
79×79	75.2%
151×151	76.4%
299×299	76.6%

表 2. 当感受野的大小变化时，但计算成本保持恒定时的识别性能比较。

Network	Top-1 Error	Top-5 Error	Cost Bn Ops
GoogLeNet [20]	29%	9.2%	1.5
BN-GoogLeNet	26.8%	-	**1.5**
BN-Inception [7]	25.2%	7.8	2.0
Inception-v2	23.4%	-	3.8
Inception-v2 RMSProp	23.1%	6.3	3.8
Inception-v2 Label Smoothing	22.8%	6.1	3.8
Inception-v2 Factorized 7×7	21.6%	5.8	4.8
Inception-v2 BN-auxiliary	**21.2**%	**5.6**%	4.8

表 3. 单作物实验结果比较不同因素的累积效应。我们将我们的数据与Ioffe等人发表的最佳单作物推论结果进行比较[7]。对于“Inception-v2”行，更改是累积的，每个接下来的行都包含前一个以及新更改。最后一行指的是我们称之为下文中的“Inception-v3”的所有更改。不幸的是，He等人[6]仅报告了10作物评估结果，而没有报告单作物结果，这在下文的表4中报告。

不到*1*%。在每种情况下，网络都是训练直到收敛，并且它们的质量是在*ImageNet ILSVRC 2012*分类基准验证集上进行评估的。结果如表 *2*所示。尽管低分辨率网络需要更长时间来训练，但最终结果的质量与高分辨率网络非常接近。

然而，如果仅仅根据输入分辨率天真地减小网络尺寸，那么网络的性能将大大下降。然而，这将是一个不公平的比较，因为我们正在比较一个成本低*16*倍的模型在更困难的任务上。

此外，表 *2*的这些结果表明，在*R-CNN [5]*的背景下，人们可能考虑使用专门的高成本低分辨率网络来处理较小的对象。

Network	Crops Evaluated	Top-5 Error	Top-1 Error
GoogLeNet [20]	10	-	9.15%
GoogLeNet [20]	144	-	7.89%
VGG [18]	-	24.4%	6.8%
BN-Inception [7]	144	22%	5.82%
PReLU [6]	10	24.27%	7.38%
PReLU [6]	-	21.59%	5.71%
Inception-v3	12	19.47%	4.48%
Inception-v3	144	**18.77%**	**4.2%**

表 4. 单模型，多作物实验结果比较不同因素的累积效应。我们将我们的数据与ILSVRC 2012分类基准上最佳发布的单模型推断结果进行比较。

10. Experimental Results and Comparisons

表 *3*展示了我们提出的架构（*Inception-v2*）在识别性能方面的实验结果，如第 *6*节所述。每一行*Inception-v2*都显示了包括突出显示的新修改以及所有先前修改的累积变化结果。标签平滑是指第 *7*节中描述的方法。分解的7×7包含了将第一个7×7卷积层分解为一系列3×3卷积层的变化。辅助*BN*指的是辅助分类器的全连接层也进行批归一化，而不仅仅是卷积操作。

我们将表 *3*中最后一行的模型称为*Inception-v3*，并在多裁剪和集成设置中评估其性能。

我们的所有评估都是在*ILSVRC-2012*验证集的*48238*个未列入黑名单的示例上进行的，如 *[16]*所建议。我们还评估了所有*50000*个示例，结果在前*5*个错误率上大约差*0.1*%，在前*1*个错误率上大约差*0.2*%。

在本文即将发布的版本中，我们将在测试集上验证我们的集成结果，但截至我们最后评估*BN-Inception*的时间（*[7]*），测试集和验证集的错误率表现出很好地相关性。

11. Conclusions

我们提供了几个设计原则来扩展卷积网络，并在*Inception*架构的背景下研究了它们。这些指导原则可以导致高性能的视觉网络，与更简单、更整体化的架构相比，其计算成本相对较低。我们的*Inception-v3*的最高质量版本在*ILSVR 2012*分类的**单裁剪**评估中达到了21.2%的*top-1*和5.6%的*top-5*错误率，创造了一个新的技术水平。与*Ioffe et al [7]*描述的网络相比，我们

Network	Models Evaluated	Crops Evaluated	Top-1 Error	Top-5 Error
VGGNet [18]	2	-	23.7%	6.8%
GoogLeNet [20]	7	144	-	6.67%
PReLU [6]	-	-	-	4.94%
BN-Inception [7]	6	144	20.1%	4.9%
Inception-v3	4	144	**17.2%**	**3.58%**[*]

表 5. 我们的多模型、多作物集成评估结果与已发表的最佳ILSVRC 2012分类基准集成推理结果进行比较。除了最佳集成结果外，所有结果均基于验证集。该集成在验证集上产生了3.46%的top-5错误率。

的解决方案在计算成本上增加了相对较少（2.5×）。尽管如此，我们的解决方案使用的计算量要远远少于基于更密集网络的最佳已发表成果：我们的模型优于*He*等人*[6] – cutting the top-*5 *(top-*1*) error by* 25% *(*14%*) relative, respectively – while being six times cheaper computationally and using at least five times less parameters (estimated). Our ensemble of four Inception-v3 models reaches* 3.5% *with multi-crop evaluation reaches* 3.5% *top-*5 *error which represents an over* 25% *reduction to the best published results and is almost half of the error of ILSVRC 2014 winining GoogLeNet ensemble.*

We have also demonstrated that high quality results can be reached with receptive field resolution as low as 79×79*. This might prove to be helpful in systems for detecting relatively small objects. We have studied how factorizing convolutions and aggressive dimension reductions inside neural network can result in networks with relatively low computational cost while maintaining high quality. The combination of lower parameter count and additional regularization with batch-normalized auxiliary classifiers and label-smoothing allows for training high quality networks on relatively modest sized training sets.*

参考文献

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] W. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, and Y. Chen. Compressing neural networks with the hashing trick. In Proceedings of The 32nd International Conference on Machine Learning*, 2015.*

[3] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In Computer Vision–ECCV 2014*, pages 184–199. Springer, 2014.*

[4] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 2155–2162. IEEE, 2014.*

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.*

[6] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. arXiv preprint arXiv:1502.01852*, 2015.*

[7] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of The 32nd International Conference on Machine Learning*, pages 448–456, 2015.*

[8] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1725–1732. IEEE, 2014.*

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems*, pages 1097–1105, 2012.*

[10] A. Lavin. Fast algorithms for convolutional neural networks. arXiv preprint arXiv:1509.09308*, 2015.*

[11] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. arXiv preprint arXiv:1409.5185*, 2014.*

[12] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.*

[13] Y. Movshovitz-Attias, Q. Yu, M. C. Stumpe, V. Shet, S. Arnoud, and L. Yatziv. Ontological supervision for fine

grained classification of street view storefronts. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1693–1702, 2015.*

[14] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. arXiv preprint arXiv:1211.5063*, 2012.*

[15] D. C. Psichogios and L. H. Ungar. Svd-net: an algorithm that automatically selects network structure. IEEE transactions on neural networks/a publication of the IEEE Neural Networks Council*, 5(3):513–515, 1993.*

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. 2014.

[17] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. arXiv preprint arXiv:1503.03832*, 2015.*

[18] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556*, 2014.*

[19] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 1139–1147. JMLR Workshop and Conference Proceedings, May 2013.*

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.*

[21] T. Tieleman and G. Hinton. Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 4, 2012. Accessed: 2015-11-05.

[22] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1653–1660. IEEE, 2014.*

[23] N. Wang and D.-Y. Yeung. Learning a deep compact image representation for visual tracking. In Advances in Neural Information Processing Systems*, pages 809–817, 2013.*