Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs

将你的内核扩大到31x31：重新审视CNN中的大内核设计

arXiv:2022.03 Accepted by CVPR 2022

摘要

我们重新审视了现代卷积神经网络（CNN）中的大核设计。受视觉转换器（ViTs）最新进展的启发，在本文中，我们证明了使用几个大的卷积核而不是一堆小核可能是一个更强大的范式。我们提出了五条准则，例如，应用重新参数化的大型深度卷积，来设计高效的高性能大核CNN。根据这些准则，我们提出了RepLKNet，一个纯CNN架构，其核大小为31×31，与常用的3×3形成对比。RepLKNet1.大大缩小了CNN和ViT之间的性能差距，例如，在ImageNet和一些典型的下游任务上取得了与Swin Transformer相当或更高的结果，而且延迟更低。RepLKNet还2.显示了对大数据和大模型的良好可扩展性，在ImageNet上获得了87.8%的顶级准确率，在ADE20K上获得了56.0%的mIoU，这在具有类似模型规模的先进技术中非常有竞争力。我们的研究进一步显示，3.与小核CNN相比，大核CNN具有更大的有效感受野和更高的形状偏向而不是纹理偏向。

代码和模型见https://github.com/megvii-research/ RepLKNet。

1介绍

卷积神经网络（CNN）[42, 55]曾经是现代计算机视觉系统中视觉编码器的一个常见选择。然而，最近，CNN[42, 55]受到了视觉变换器（ViTs）[35, 61, 89, 98]的极大挑战，它们在许多视觉任务上表现出领先的性能--不仅是图像分类[35, 109]和表示学习[4, 10, 17, 105]，而且还有许多下游任务，如物体检测[25, 61]、语义分割[98, 103]和图像修复[11, 56]。为什么ViTs超级强大？一些研究认为，ViTs中的多头自我注意（MHSA）机制起到了关键作用。他们提供的经验结果证明，MHSA更灵活[52]，更有能力（更少的归纳偏见）[21]，对扭曲更稳健[68，103]，或者能够模拟长距离的依赖关系[71，93]。但是有些工作挑战MHSA的必要性[121]，将ViTs的高性能归因于适当的构建模块[34]，和/或动态稀疏权重[40,116]。更多的工作[21,40,44,100,121]从不同的角度解释了ViTs的优越性。在这项工作中，我们专注于一个观点：建立大感受野的方式。在ViTs中，MHSA通常被设计成全局的[35, 80, 98]或局部的但具有大内核的[61, 72, 92]，因此单个MHSA层的每个输出都能收集大区域的信息。然而，大核在CNN中没有被普遍采用（除了第一层[42]）。相反，一种典型的方式是使用许多小空间卷积的堆栈1 [42, 46, 49, 70, 79, 84, 114]（例如，3×3）来扩大最先进的CNN中的感受野。只有一些老式的网络，如AlexNet[55]、Inceptions[81-83]和一些从神经架构搜索中得到的架构[39, 45, 58, 122]采用大型空间卷积（其大小大于5）作为主要部分。上述观点自然而然地引出了一个问题：如果我们用几个大核而不是许多小核来做传统的CNN呢？大内核或构建大感受野的方式是缩小CNN和ViTs之间性能差距的关键吗？为了回答这个问题，我们系统地探索了CNN的大核设计。我们遵循一个非常简单的 "哲学"：只是在传统网络中引入大深度卷积，其大小从3×3到31×31不等，尽管存在其他通过单层或几层引入大感受野的方法，例如特征金字塔[96]、扩张卷积[14, 106, 107]和可变形卷积[24]。通过一系列的实验，我们总结了五条经验性的准则来有效地运用大型卷积。

1）非常大的内核在实践中仍然是有效的；

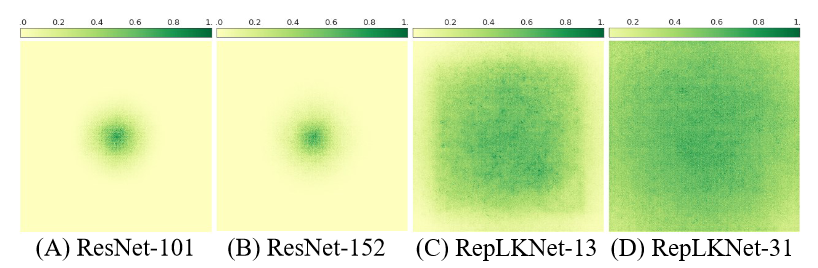
2）身份捷径是至关重要的，特别是对于具有非常大的内核的网络；

3）用小的内核重新参数化[31]有助于弥补优化问题；

4）大的卷积对下游任务的促进作用远远超过ImageNet；

5）大内核即使在小的特征图上也是有用的。

基于上述准则，我们提出了一个名为RepLKNet的新架构，这是一个纯的CNN，其中采用了重新参数化的大卷积来建立大的感受野。我们的网络总体上遵循Swin Transformer[61]的宏观架构，并做了一些修改，同时用大深度卷积代替了多头自留地。我们主要以中型和大型模型为基准，因为ViTs曾经被认为在大型数据和模型上超过了CNNs。在ImageNet分类上，我们的基线（与Swin-B的模型大小相似），其核大小高达31×31，仅在ImageNet1K数据集上训练的准确率就达到了84.8%，比Swin-B好0.3%，但在延迟上更有效率。更重要的是，我们发现大内核设计在下游任务上特别强大。例如，在相似的复杂度和参数预算下，我们的网络在COCO检测[57]上的表现比ResNeXt-101[104]或ResNet101[42]骨干高出4.4%，在ADE20K分割[120]上高出6.1%，这也与对应的Swin Transformers相当甚至更好，但推理速度更高。考虑到更多的预训练数据（如7300万张图像）和更多的计算预算，我们的最佳模型在类似的模型规模下获得了非常有竞争力的结果，如在ImageNet上获得了87.8%的最高准确率，在ADE20K上获得了56.0%，这显示了对大规模应用的良好扩展性。我们认为RepLKNet的高性能主要是由于通过大内核建立的大有效感受野（ERFs）[65]，如图1中的比较。



此外，RepLKNet被证明比传统的CNN利用了更多的形状信息，这与人类的认知部分一致。我们希望我们的发现可以帮助理解CNN和ViTs的内在机制。  
  
2. 相关工作

2.1. 大核模型

正如介绍中提到的，除了一些老式的模型，如Inceptions[81-83]，大核模型在VGG-Net[79]之后变得不受欢迎。一个代表性的工作是全局卷积网络（GCNs）[69]，它使用1×K之后的K×1的超大卷积来改善语义分割任务。然而，据报道，大内核会损害ImageNet的性能。局部关系网络（LRNet）[47]提出了一个空间聚合算子（LRLayer）来代替标准卷积，它可以被视为一个动态卷积。LR-Net可以从7×7的核大小中获益，但性能会随着9×9而下降。在核大小与特征图一样大的情况下，top-1的准确率从75.7%显著降低到68.4%。最近，Swin Transformers[61]提出用移位的窗口注意力来捕捉空间模式，其窗口大小从7到12，这也可以看作是大核的一个变体。后续的研究[33, 60]采用了更大的窗口尺寸。受这些局部变换器的成功启发，最近的一项工作[40]在[61]中用静态或动态的7×7深度卷积代替了MHSA层，同时仍然保持了可比的结果。虽然[40]提出的网络与我们的设计模式相似，但动机不同：[40]没有研究ERF、大内核和性能之间的关系；相反，它将视觉变换器的卓越性能归功于稀疏连接、共享参数和动态机制。另外三个有代表性的工作是全局过滤网络（Global Filter Networks，GFNets）[74]、CKConv[76]和FlexConv[75]。GFNet优化了傅里叶域中的空间连接权重，这相当于空间域中的循环全局卷积。CKConv将核制定为连续函数来处理顺序数据，可以构建任意大的核。FlexConv为不同的爱因斯坦学习不同的核大小，其大小可以和特征图一样大。虽然他们使用了非常大的核，但他们并不打算回答我们所希望的关键问题：为什么传统的CNN表现不如ViTs，以及如何在普通的CNN中应用大核。此外，[40]和[74]都没有在强大的基线上评估他们的模型，例如，大于SwinL的模型。因此，目前还不清楚大核CNN是否能很好地作为转化器进行扩展。同时进行的工作。ConvMixer[90]使用高达9×9的卷积来替代ViTs[35]或MLPs[87, 88]的 "混合器 "组件。MetaFormer[108]建议池化层是自我关注的一个替代方案。ConvNeXt[62]采用了7×7深度卷积来设计强大的架构，推动了CNN性能的极限。尽管这些作品显示了出色的性能，但它们并没有显示出更大的卷积（例如31×31）的好处。  
2.2. 模型缩放技术

给定一个小的模型，通常的做法是将其放大以获得更好的性能，因此缩放策略在所产生的准确性-效率权衡中起着至关重要的作用。对于CNN来说，现有的扩展方法通常集中在模型深度、宽度、输入分辨率[32, 70, 84]、瓶颈比率和群组宽度[32, 70]。然而，内核的大小往往被忽略了。在第3节中，我们将说明内核大小也是CNN中一个重要的缩放维度，特别是对于下游任务。

2.3. 结构重参数化

结构重参数化[27-31]是一种通过转换参数来等效转换模型结构的方法。

基础拓展：参数A-参数B，结构a-结构b

例如，RepVGG针对深度推理时间的VGG-like（例如，无分支）模型，在训练过程中构建了与3×3层平行的额外ResNet式快捷通道。与难以训练的真正的VGG-like模型相比[42]，这种捷径帮助模型达到了令人满意的性能。训练结束后，捷径通过一系列的线性变换被吸收到平行的3×3内核中，这样，产生的模型就成为一个类似VGG的模型。在本文中，我们用这种方法将一个相对较小的（如3×3或5×5）内核加入到一个很大的内核中。通过这种方式，我们使非常大的核能够捕捉到小规模的模式，从而提高模型的性能。

3. 应用大卷积的准则   
简单地将大卷积应用于CNN通常会导致较差的性能和速度。在这一节中，我们总结了有效使用大内核的5条准则。  
**准则1：大深度卷积在实践中可以很有效**。人们认为大核卷积的计算成本很高，因为核的大小会四次增加参数和FLOPs的数量。这个缺点可以通过应用深度明智（DW）卷积来大大克服[18,46]。例如，在我们提出的RepLKNet中（详见表5），将不同阶段的核大小从[3, 3, 3, 3]增加到[31, 29, 27, 13]，只使FLOPs和参数数分别增加18.6%和10.4%，这是可以接受的。剩下的1×1卷积实际上主导了大部分的复杂性。人们可能会担心DW卷积在现代并行计算设备（如GPU）上的效率会很低。对于传统的DW 3×3内核来说是如此[46, 77, 114]，因为DW操作引入了低比率的计算与内存访问成本[66]，这对现代计算架构并不友好。然而，我们发现当内核大小变大时，计算密度也会增加：例如，在DW 11×11内核中，每次从特征图中加载一个值，最多可以参加121次乘法，而在3×3内核中，这个数字只有9。因此，根据屋顶线模型，当内核尺寸变大时，实际延迟的增加不应该和FLOPs的增加一样多。  
**备注1**. 不幸的是，我们发现现成的深度学习工具（如Pytorch）对大型DW卷积的支持很差，如表1所示。因此，我们尝试了几种方法来优化CUDA内核。基于FFT的方法[67]对于实现大型卷积似乎是合理的。然而，在实践中我们发现块状（反）隐式gemm算法是一个更好的选择。该实现已被集成到开源框架MegEngine[1]中，我们在此略去细节。我们还发布了一个用于PyTorch的高效实现[2]。表1显示，与Pytorch的基线相比，我们的实现要有效得多。通过我们的优化，RepLKNet中DW卷积的延迟贡献率从49.5%降低到12.3%，这与FLOPs占用的比例基本一致。

**准则2：身份捷径是至关重要的，特别是对于具有非常大内核的网络**。为了证明这一点，我们使用MobileNet V2 [77]作为基准，因为它大量使用DW层，并且有两个已发布的变体（有或没有捷径）。对于大核的对应模型，我们只是用13×13替换所有的DW 3×3层。所有的模型都在ImageNet上用相同的训练配置训练了100次（详见附录A）。表2显示，大内核使有捷径的MobileNet V2的准确性提高了0.77%。然而，在没有捷径的情况下，大内核的准确率只降低到53.98%。

**备注2.** 该准则也适用于ViTs。最近的一项工作[34]发现，如果没有身份捷径，注意力会随着深度的增加而加倍地失去等级，从而导致过度平滑的问题。尽管大核CNN可能会以不同于ViT的机制退化，但我们也观察到如果没有捷径，网络就很难捕捉到局部细节。从与[94]类似的角度来看，捷径使模型成为由众多具有不同感受野（RFs）的模型组成的隐性集合体，因此它可以从更大的最大RF中获益，同时不失捕捉小规模模式的能力。

**准则3：用小内核重新参数化[31]有助于弥补优化问题。**我们将MobileNet V2的3×3层分别替换为9×9和13×13，并选择性地采用结构重参数化[27,28,31]方法。具体来说，我们构建一个与大层平行的3×3层，然后将它们的输出在批量标准化（BN）[51]层后相加（图2）。训练结束后，我们将小核以及BN参数合并到大核中，所以结果模型等同于训练时的模型，但不再有小核。表3显示直接将核的大小从9增加到13会降低准确性，而重新参数化则可以解决这个问题。然后，我们用DeepLabv3+[16]在Cityscapes[22]上将ImageNet训练的模型转移到语义分割中。我们只替换主干，并保留MMSegmentation[20]提供的所有默认训练设置。观察结果与ImageNet上的情况相似。3×3的重新参数使9×9模型的mIoU提高了0.19，13×13模型提高了0.93。通过这样简单的重新参数化，将核大小从9增加到13不再降低ImageNet和Cityscapes的性能。

**备注3.** 众所周知，ViTs有优化问题，特别是在小数据集上[35, 59]。一个常见的解决方法是引入卷积先验，例如，在每个自注意块上添加一个DW 3×3卷积[19, 101]，这与我们的方法类似。这些策略为网络引入了额外的平移等价性和定位先验，使其更容易在小数据集上进行优化而不损失通用性。与ViT的表现类似[35]，我们也发现当预训练数据集增加到7300万张图片时（参考下一节中的RepLKNet-XL），可以省略重新参数化而不降低性能。

**准则4：大型卷积对下游任务的促进作用远远超过ImageNet分类。**表3（重新参数化后）显示，将MobileNet V2的核大小从3×3增加到9×9，使ImageNet的准确性提高了1.33%，但Cityscapes mIoU提高了3.99%。表5显示了类似的趋势：当内核大小从[3，3，3，3]增加到[31，29，27，13]时，ImageNet的准确性只提高了0.96%，而ADE20K[120]上的mIoU提高了3.12%。这种现象表明，ImageNet得分相似的模型在下游任务中可能有非常不同的能力（就像表5中最下面的3个模型）。

**备注4.** 是什么导致了这种现象？首先，大核设计大大增加了有效感受野（ERFs）[65]。许多工作已经证明了 "上下文 "信息，这意味着大的ERFs，在许多下游任务中是至关重要的，如物体检测和语义分割[63, 69, 96, 106, 107]。我们将在第5节讨论这个话题。第二，我们认为另一个原因可能是大核的设计给网络带来了更多的形状偏差。简而言之，ImageNet图片可以根据纹理或形状进行正确的分类，正如[8, 36]中提出的那样。然而，人类识别物体主要是基于形状线索而不是纹理，因此，具有更强形状偏向的模型可能会更好地转移到下游任务中。最近的一项研究[91]指出，ViTs的形状偏向性很强，这部分解释了为什么ViTs在转移任务中超级强大。相反，在ImageNet上训练的传统CNN倾向于偏向于纹理[8,36]。幸运的是，我们发现简单地扩大CNN的核大小可以有效地改善形状偏差。详情请参考附录C。  
**准则5：大内核（如13×13）即使在小特征图（如7×7）上也是有用的。**为了验证这一点，我们在MobileNet V2的最后阶段将DW卷积放大到7×7或13×13，因此内核大小与特征图大小（默认为7×7）相当甚至更大。我们按照准则3的建议，对大核进行了重新参数化。表4显示，尽管最后阶段的卷积已经涉及到非常大的感受野，但进一步增加核的大小仍然会导致性能的改善，特别是在下游任务中，如语义分割。

**备注5.** 当内核大小变大时，注意到CNN的平移等价性并不严格成立。如图3所示，

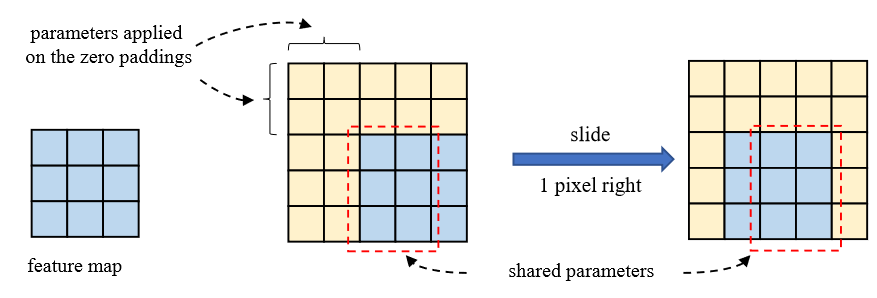


图3. 用小特征图和大内核进行卷积的说明。相邻位置的两个输出只共享一部分核的权重。平移等价性并不严格成立。

相邻空间位置的两个输出只共享内核权重的一部分，也就是说，被不同的映射所转换。这一特性也符合ViTs的 "哲学"--放宽对称先验以获得更多的容量。有趣的是，我们发现在变换器界广泛使用的二维相对位置嵌入（RPE）[5,78]，也可以看作是一个大小为(2H -1)×(2W -1)的大深度核，其中H和W分别为特征图的高度和宽度。大核不仅有助于学习概念之间的相对位置，而且由于填充效应，还可以编码绝对位置信息[53]。

4. RepLKNet：一个大内核架构

遵循上述准则，在本节中我们提出了RepLKNet，一个具有大内核设计的纯CNN架构。据我们所知，到目前为止，CNN仍然主导着小模型[113, 115]，而视觉变换器被认为在更多的复杂性预算下比CNN更好。因此，在本文中，我们主要关注相对较大的模型（其复杂性与ResNet-152[42]或Swin-B[61]相当或更大），以验证大内核设计是否能消除CNN和ViTs之间的性能差距。

4.1. 架构规范

我们在图4中简述了RepLKNet的架构：

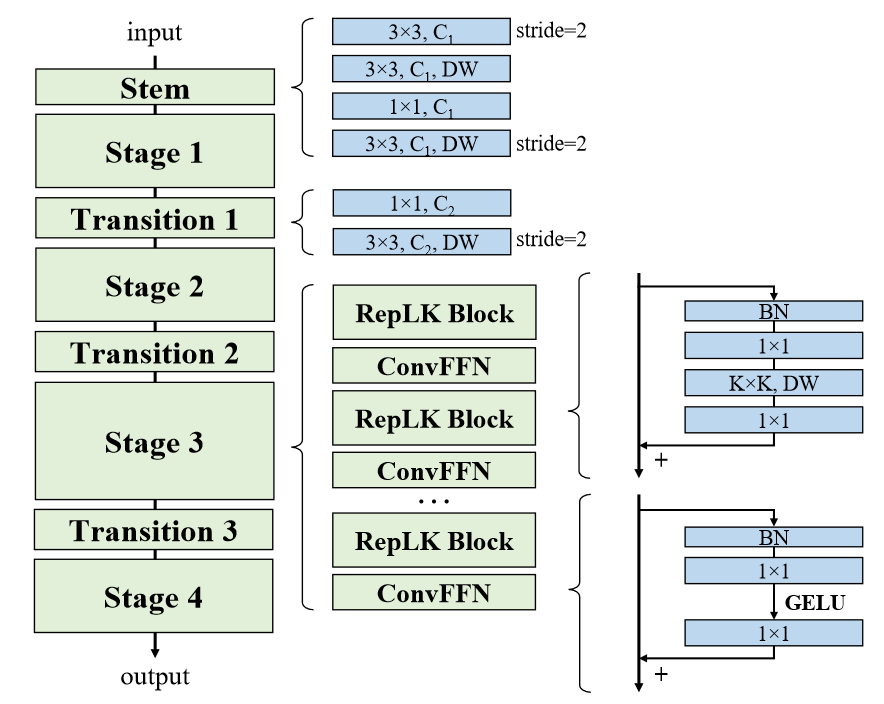


图4. RepLKNet由干、阶段和过渡组成。除了纵深（DW）大核，其他组件包括DW 3×3，密集1×1 conv，和批量归一化[51]（BN）。请注意，每一个conv层都有一个下面的BN，这些都没有描绘出来。这样的conv-BN序列使用ReLU作为激活函数，除了捷径添加之前的序列（作为一种常见的做法[42, 77]）和GELU之前的序列[43]。

Stem指的是起始层。由于我们的目标是在下游的密集预测任务中实现高性能，我们希望在开始时通过几个conv层来捕获更多的细节。在第一个3×3的2×下采样之后，我们安排了一个DW 3×3层来捕捉低层次的模式，一个1×1的conv，以及另一个DW 3×3层来下采样。

第1-4Stage各包含几个RepLK块，它们使用快捷方式（准则2）和DW大内核（准则1）。我们在DW conv之前和之后使用1×1 conv，这是一个常见的做法。请注意，每个DW大卷积使用5×5内核进行重新参数化（准则3），这在图4中没有显示。除了大conv层提供足够的感受野和聚集空间信息的能力外，模型的表征能力也与深度密切相关。为了提供更多的非线性和跨通道的信息交流，我们希望使用1×1层来增加深度。受广泛用于变换器[35, 61]和MLPs[27, 87, 88]的前馈网络（Feed-Forward Network, FFN）的启发，我们使用一个类似CNN风格的块，由快捷键、BN、两个1×1层和GELU[43]组成，所以被称为ConvFFN块。与经典的FFN在全连接层之前使用层归一化[3]相比，BN的优势在于它可以融合到conv中以实现高效推理。按照通常的做法，ConvFFN块的内部通道数为4×，作为输入。简单地按照ViT和Swin的做法，将注意力和FFN块交错放置，我们在每个RepLK块之后放置一个ConvFFN。

过渡块被放置在各阶段之间，首先通过1×1 conv增加通道尺寸，然后用DW 3×3 conv进行2×下采样。 总之，每个阶段有三个架构超参数：RepLK块的数量B、通道尺寸C和核大小K。  
4.2. 让大核更大

我们继续通过固定B=[2, 2, 18, 2], C=[128, 256, 512, 1024], 改变K来评估RepLKNet上的大核，并观察分类和语义分割的性能。在没有仔细调整超参数的情况下，我们随便将核的大小设置为[13，13，13，13]，[25，25，25，13]，[31，29，27，13]，并将这些模型称为RepLKNet-13/25/31。我们还构建了两个小核基线，其中核大小都是3或7（RepLKNet-3/7）。在ImageNet上，我们使用AdamW[64]优化器、RandAugment[23]、mixup[111]、CutMix[110]、Rand Erasing[118]和Stochastic Depth[50]，按照最近的工作[4, 61, 62, 89]，训练了120个epoch。详细的训练配置见附录A。对于语义分割，我们使用ADE20K[120]，这是一个广泛使用的大规模语义分割数据集，包含150个类别的2万张图像用于训练，2千张用于验证。我们使用ImageNet训练的模型作为骨干，并采用由MMSegmentation[20]实现的UperNet[102]的80K迭代训练设置，测试单尺度mIoU。表5显示了我们在不同核大小下的结果。在ImageNet上，虽然将核的大小从3增加到13可以提高准确率，但如果将核的大小变得更大，则不会带来进一步的改善。然而，在ADE20K上，将内核从[13, 13, 13, 13]扩大到[31, 29, 27, 13]，只增加了5.3%的参数和3.5%的FLOPs，就带来了0.82%的高mIoU，这突出了大内核对于下游任务的重要性。在下面的小节中，我们使用RepLKNet31与更强的训练配置来比较ImageNet分类、Cityscapes/ADE20K语义分割和COCO[57]物体检测方面的先进技术。我们将上述模型称为RepLKNet-31B（B代表Base），将C=[192, 384, 768, 1536]的更广泛的模型称为RepLKNet31L（Large）。我们构建另一个RepLKNet-XL，C = [256, 512, 1024, 2048]，在RepLK Blocks中采用1.5×倒置瓶颈设计（即DW大conv层的通道为1.5×作为输入）。  
4.3. ImageNet分类

由于RepLKNet的整体架构与Swin相似，我们希望首先做一个比较。对于ImageNet-1K上的RepLKNet-31B，我们将上述的训练计划扩展到300个epochs，以进行公平的比较。然后我们在输入分辨率为384×384的情况下微调30个epochs，这样总的训练成本就比Swin-B模型低很多，因为Swin-B模型是用384×384从头开始训练的。然后我们在ImageNet-22K上预训练RepLKNet-B/L模型，在ImageNet-1K上进行微调。RepLKNetXL在我们名为MegData73M的私有半监督数据集上进行了预训练，该数据集在附录中有所介绍。我们还介绍了在同一2080Ti GPU上以64个批次的规模测试的吞吐量。训练配置在附录中介绍。表6显示，尽管非常大的内核并不打算用于ImageNet分类，但我们的RepLKNet模型在准确性和效率之间表现出了有利的权衡。值得注意的是，仅在ImageNet-1K的训练下，RepLKNet-31B就达到了84.8%的准确率，比Swin-B高0.3%，而且运行速度快43%。而即使RepLKNet-XL的FLOPs比Swin-L高，它的运行速度也更快，这突出了超大内核的效率。

4.4. 语义分割

我们随后将预训练的模型作为Cityscapes（表7）和ADE20K（表8）的骨干。具体来说，我们使用由MMSegmentation[20]实现的UperNet[102]，对Cityscapes使用80K迭代训练计划，对ADE20K使用160K。由于我们只想评估骨干网，所以我们没有使用任何高级技术、技巧或自定义算法。在Cityscapes上，ImageNet-1K训练的RepLKNet31B比Swin-B要好得多（单尺度mIoU为2.7），甚至比ImageNet22K训练的Swin-L还要好。即使配备了DiversePatch[38]，一种为视觉变换器定制的技术，22K预训练的Swin-L的单尺度mIoU仍然低于我们1K预训练的RepLKNet-31B，尽管前者的参数是2倍。在ADE20K上，RepLKNet-31B在1K和22K预训练下都优于Swin-B，而且单尺度mIoU的幅度特别大。用我们的半监督数据集MegData73M进行预训练，RepLKNet-XL实现了56.0的mIoU，这显示了对大规模视觉应用的可行扩展性。  
4.5. 对象检测

对于对象检测，我们使用RepLKNets作为FCOS[86]和Cascade Mask R-CNN[9, 41]的骨干，它们是单阶段和双阶段检测方法的代表，也是MMDetection[13]的默认配置。FCOS模型的训练采用2x（24-epoch）训练计划，以便与同一代码库[20]中的X101（ResNeXt-101的简称[104]）基线进行公平比较，其他使用Cascade Mask R-CNN的结果都采用3x（36-epoch）。同样，我们只是替换了骨干，没有使用任何高级技术。表9显示RepLKNets比ResNeXt-101-64x4d的性能好4.4 mAP，同时参数更少，FLOPs更低。请注意，使用HTC[12]、HTC++[61]、Soft-NMS[7]或6x(72-epoch)时间表等先进技术，结果可能会进一步提高。与Swin相比，RepLKNets以较少的参数和较低的FLOPs实现了更高或相当的mAP。值得注意的是，RepLKNet-XL实现了55.5的mAP，这再次证明了其可扩展性。

5. 讨论

5.1. 大核CNN比深层小核模型有更大的ERF

我们已经证明了大核设计可以大大提升CNN（尤其是在下游任务上）。然而，值得注意的是，大核可以由一系列的小卷积来表达[79]，例如，一个7×7的卷积可以分解为三个3×3的核的堆栈，而没有信息损失（分解后需要更多的通道来保持自由度）。鉴于这一事实，一个问题自然而然地出现了：为什么传统的CNN，可能包含几十或几百个小卷积（例如ResNets[42]），仍然表现得比大核网络差？我们认为，就获得大的感受野而言，一个大核比许多小核要有效得多。首先，1.根据有效感受场（ERF）的理论[65]，ERF与O(K√L)成正比，其中K是核大小，L是深度，即层数。换句话说，ERF随内核大小线性增长，而随深度亚线性增长。其次，2.深度的增加带来了优化困难[42]。尽管ResNets似乎克服了这一困境，设法训练一个有数百层的网络，但一些工作[26, 94]表明ResNets可能并不像它们看起来那样深。例如，[94]认为ResNets的行为就像浅层网络的集合体，这意味着即使深度大大增加，ResNets的ERF仍然可能非常有限。这种现象在以前的工作中也被经验性地观察到[54]。综上所述，大核设计需要较少的层数来获得大的ERFs，并避免了深度增加带来的优化问题。为了支持我们的观点，我们选择ResNet-101/152和前面提到的RepLKNet-13/31作为小核和大核模型的代表，它们都在ImageNet上训练有素，并用ImageNet验证集的50张图像进行测试，尺寸调整为1024×1024。为了使ERF可视化，我们使用了一个简单而有效的方法（代码发布在[2]），如附录B中介绍的，遵循[54]。简而言之，我们产生一个聚合贡献分数矩阵A（1024×1024），其中每个条目a（0≤a≤1）衡量输入图像上相应像素对最后一层产生的特征图中心点的贡献。图1显示ResNet-101的高贡献像素聚集在中心点周围，但外围点的贡献非常低，表明ERF有限。ResNet-152显示了类似的模式，表明更多的3×3层并没有明显增加ERF。另一方面，图1（C）中的高贡献像素分布更均匀，表明RepLKNet-13关注了更多的外围像素。在更大的内核下，RepLKNet-31使高贡献度像素分布更均匀，表明ERF更大。表10给出了一个定量分析，我们报告了一个最小矩形的高贡献面积比r，该矩形覆盖了超过给定阈值t的贡献分数。例如，ResNet-101的20%的像素贡献（A值）位于中心的103×103区域内，因此面积比是（103/1024）2=1.0%，t=20%。我们做了几个有趣的观察。1）虽然明显更深，但ResNets的ERFs比RepLKNets小得多。例如，ResNet-101的99%以上的贡献分数都在一个小区域内，只占总面积的23.4%，而RepLKNet-31的这种面积比为98.6%，这意味着大部分像素对最终的预测有很大的贡献。2）在ResNet-101上添加更多的层并不能有效地扩大ERF，而扩大核的规模却能以边际的计算成本提高ERF。

5.2. 大核模型在形状偏差方面与人类更相似

我们发现RepLKNet-31B的形状偏差比Swin Transformer和小核CNN高很多。最近的一项工作[91]报告说，视觉变换器与人类视觉系统更相似，它们更多的是根据物体的整体形状进行预测，而CNN则更多的是关注局部纹理。我们遵循其方法并使用其工具箱[6]来获得RepLKNet-31B和Swin-B在ImageNet-1K或22K上预训练的形状偏差（例如，基于形状而不是纹理进行预测的部分），以及两个小核基线，RepLKNet-3和ResNet-152的情况。图5显示，RepLKNet比Swin有更高的形状偏差。考虑到RepLKNet和Swin有相似的整体架构，我们估计形状偏差与有效感知场密切相关，而不是自我注意的具体表述（即查询-键值设计）。这也解释了1）[91]报告的ViTs[35]的高形状偏差（因为ViTs采用全局注意），2）1K-retrained Swin的低形状偏差（局部窗口内的注意），以及3）小核基线RepLKNet-3的形状偏差，它与ResNet-152非常接近（两个模型都是由3×3卷积组成）。

5.3. 大核设计是与ConvNeXt协同工作的通用设计元素

用大到31×31的核代替ConvNeXt[62]中的7×7的卷积，会带来明显的改进，例如，ConNeXt-Tiny +大核>ConNeXt-Small ，ConNeXt-Small +大核>ConNeXt-Base。我们使用最近提出的ConvNeXt[62]作为基准架构来评估大内核作为一个通用设计元素。我们简单地将ConvNeXt[62]中的7×7卷积替换为大到31×31的内核。在ImageNet（120 epochs）和ADE20K（80K iterations）上的训练配置与第4.2节中的结果相同。表11显示，虽然原始核已经是7×7，但进一步增加核的大小仍然带来了明显的改进，特别是在下游任务上：在核大到31×31的情况下，ConvNeXt-Tiny优于原始ConvNeXt-Small，而大核的ConvNeXt-Small优于原始ConvNeXt-Base。这类现象再次表明，内核大小是一个重要的扩展维度。

5.4. 大核优于高扩张率的小核

详细内容请参见附录C。

6. 局限性  
尽管大核设计极大地提高了CNN在ImageNet和下游任务上的表现，然而，根据表6，随着数据和模型规模的增加，RepLKNets开始落后于Swin Transformers，例如，RepLKNet-31L的ImageNet top-1准确率比ImageNet-22K预训练的Swin-L低0.7%（而下游的分数仍然是相当的）。目前还不清楚这种差距是由不理想的超参数调整造成的，还是当数据/模型扩大时出现的CNN的一些其他基本缺点。我们正在努力解决这个问题。  
  
7. 结论  
本文重新审视了大卷积核，它在设计CNN架构时长期被忽视。我们证明了使用几个大核而不是许多小核可以更有效地获得更大的有效感受野，使CNN的性能，特别是在下游任务上的性能得到很大的提升，并且在数据和模型扩大时，大大缩小了CNN和ViT之间的性能差距。我们希望我们的工作能够推动CNN和ViTs的研究。一方面，对于CNN社区来说，我们的发现表明我们应该特别关注ERFs，这可能是高性能的关键。另一方面，对于ViT社区来说，由于大的卷积可以替代具有类似行为的多头自留地，它可能有助于理解自留地的内在机制。