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

论文地址

代码地址

对一些传统认识的挑战

1. 超大卷积不但不涨点，还会掉点?

在现代CNNC设计加持下，kernel size越大越涨点

2. 超大卷积效率很差?

超大depth-wise卷积并不会增加多少FLOPs。如果再加点底层优化，速度会更快，31x31的计算密度最高可达3x3的70倍

3. imagenet点数很重要?

下游任务的性能可能和imagenet关系不大

4. 大卷积只能用在大feature map上?

在7x7的feature map上用13x13的卷积都能涨点

5. 超深CNN堆叠大量3x3，所以感受野很大?

深层小kernel的有效感受野其实很小，反而少量超大的卷积核的有效感受野非常大

6. self-attention在下游任务中性能很好是因为self-attention本质更强?

kernel size可能才是下游任务涨点的关键

提出在线代CNN中应用超大卷积核的五条准则

1. 用depth-wise超大卷积，最好再加底层优化
2. 加shortcut
3. 用小卷积核做重参数化
4. 要看下游任务的性能，不能只看ImageNet点数高低
5. 小feature map上也可以用大卷积，常规分辨率就能训大kernel模型

基于五条准则，提出一种架构RepLKNet

简单借鉴Swin Transformer的宏观架构，其中大量使用超大卷积，如27x27、31x31等。这一架构的其他部分非常简单，都是1x1卷积、Batch Norm等喜闻乐见的简单结构，不用任何attention。

ImageNet上，与Swin-Base相当。在额外数据训练下，超大量级模型最高达到87.8%的正确率。超大卷积核本来不是为刷ImageNet设计的，这个点数也算是可以让人满意。

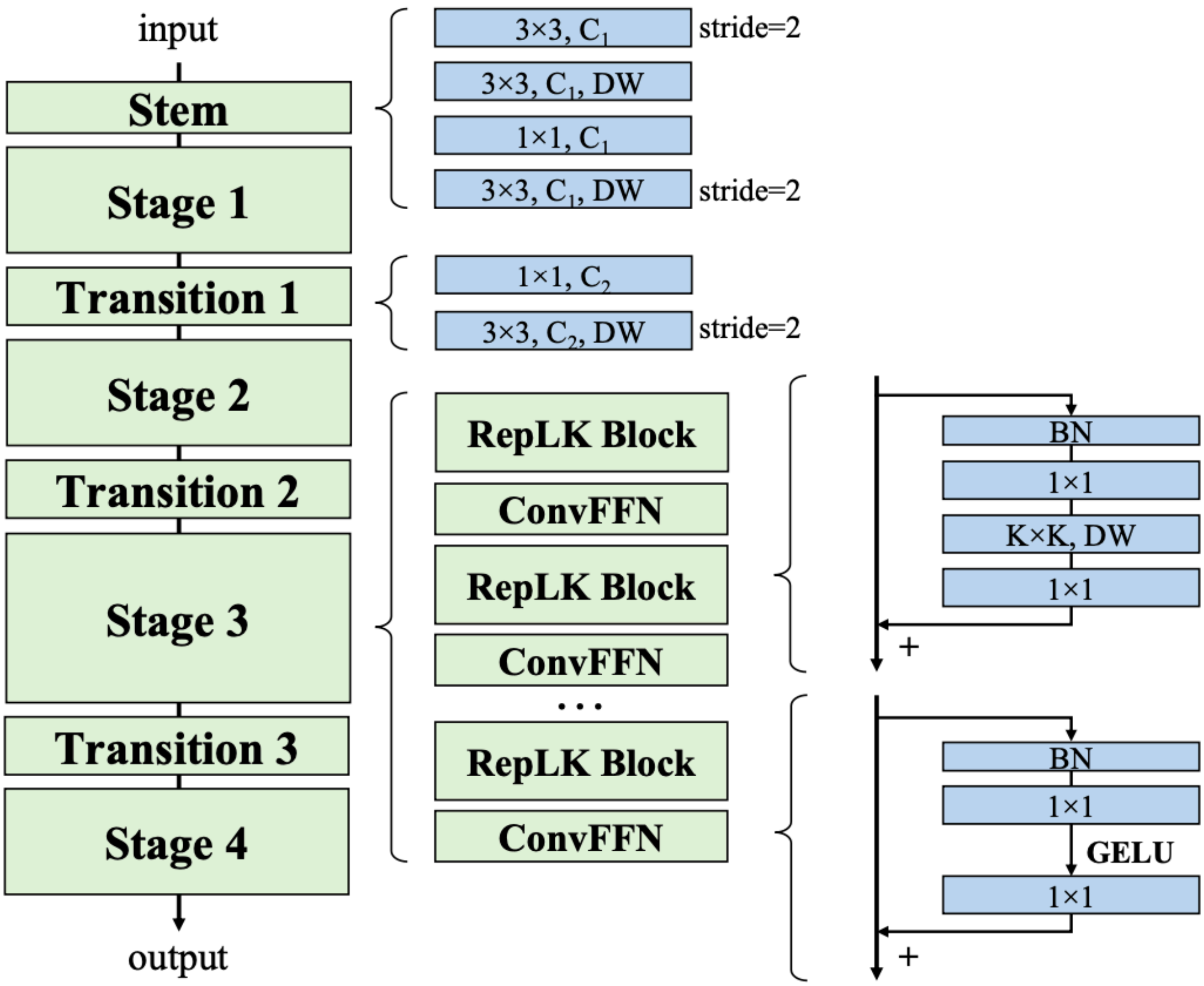

Figure 4. RepLKNet comprises Stem, Stages and Transitions. Except for depth-wise (DW) large kernel, the other components include DW 3×3, dense 1×1 conv, and batch normalization [51] (BN). Note that every conv layer has a following BN, which are not depicted. Such conv-BN sequences use ReLU as the activation function, except those before the shortcut-addition (as a common practice [42, 77]) and those preceding GELU [43].

在各种下游任务上的效果

1. 分类

ImageNet上，与Swin-Base相当。在额外数据训练下，超大量级模型最高达到**87.8%**的正确率。超大卷积核本来不是为刷ImageNet设计的，这个点数也算是可以让人满意。

是为刷ImageNet设计的，这个点数也算是可以让人满意。

Table 6. ImageNet results. The throughput is tested with FP32 and a batch size of 64 on 2080Ti. ‡ indicates ImageNet-22K pretraining. ◇ indicates pretrained with extra data.

Model	Input resolution	Top-1 acc	Params (M)	FLOPs (G)	Throughput examples/s
RepLKNet-31B	224×224	83.5	79	15.3	295.5
Swin-B	224×224	83.5	88	15.4	226.2
RepLKNet-31B	384×384	84.8	79	45.1	97.0
Swin-B	384×384	84.5	88	47.0	67.9
RepLKNet-31B ‡	224×224	85.2	-	-	-
Swin-B ‡	224×224	85.2	-	-	-
RepLKNet-31B ‡	384×384	86.0	-	-	-
Swin-B ‡	384×384	86.4	-	-	-
RepLKNet-31L ‡	384×384	86.6	172	96.0	50.2
Swin-L ‡	384×384	87.3	197	103.9	36.2
RepLKNet-XL ◇	320×320	87.8	335	128.7	39.1

2. 语义分割

Cityscapes语义分割上，仅用**ImageNet-1K pretrain的RepLKNet-Base**，甚至超过了**ImageNet-22K pretrain的Swin-Large**。这是**跨模型量级、跨数据量级的超越**。

Table 7. Cityscapes results. The FLOPs is computed with 1024×2048 inputs. The mIoU is tested with single-scale (ss) and multi-scale (ms). The results with Swin are implemented by [38]. ‡ indicates ImageNet-22K pretraining.

Backbone	Method	mIoU (ss)	mIoU (ms)	Param (M)	FLOPs (G)
RepLKNet-31B	UperNet [102]	**83.1**	**83.5**	110	2315
ResNeSt-200 [112]	DeepLabv3 [15]	-	82.7	-	-
Axial-Res-XL	Axial-DL [95]	80.6	81.1	173	2446
Swin-B	UperNet	80.4	81.5	121	2613
Swin-B	UperNet + [38]	80.8	81.8	121	-
ViT-L ‡	SETR-PUP [117]	79.3	82.1	318	-
ViT-L ‡	SETR-MLA	77.2	-	310	-
Swin-L ‡	UperNet	82.3	83.1	234	3771
Swin-L ‡	UperNet + [38]	82.7	83.6	234	-

ADE20K语义分割上，ImageNet-1K pretrain的模型大幅超过ResNet、ResNeSt等小kernel传统CNN。**Base级别模型显著超过Swin**，Large模型与Swin相当。超大量级模型达到**56%的mIoU**。

Table 8. ADE20K results. The mIoU is tested with single-scale (ss) and multi-scale (ms). The results with 1K-pretrained Swin are cited from the official GitHub repository. ‡ indicates ImageNet-22K pretraining and 640×640 finetuning on ADE20K. ⋄ indicates pretrained with extra data. The FLOPs is computed with 2048×512 for the ImageNet-1K pretrained models and 2560×640 for the ImageNet-22K and larger, following Swin.

Backbone	Method	mIoU (ss)	mIoU (ms)	Param (M)	FLOPs (G)
RepLKNet-31B	UperNet	**49.9**	**50.6**	112	1170
ResNet-101	UperNet [102]	43.8	44.9	86	1029
ResNeSt-200 [112]	DeepLabv3 [15]	-	48.4	113	1752
Swin-B	UperNet	48.1	49.7	121	1188
Swin-B	UperNet + [38]	48.4	50.1	121	-
ViT-Hybrid	DPT-Hybrid [73]	-	49.0	90	-
ViT-L	DPT-Large	-	47.6	307	-
ViT-B	SETR-PUP [117]	46.3	47.3	97	-
ViT-B	SETR-MLA [117]	46.2	47.7	92	-
RepLKNet-31B ‡	UperNet	**51.5**	**52.3**	112	1829
Swin-B ‡	UperNet	50.0	51.6	121	1841
RepLKNet-31L ‡	UperNet	**52.4**	52.7	207	2404
Swin-L ‡	UperNet	52.1	**53.5**	234	2468
ViT-L ‡	SETR-PUP	48.6	50.1	318	-
ViT-L ‡	SETR-MLA	48.6	50.3	310	-
RepLKNet-XL ⋄	UperNet	**55.2**	**56.0**	374	3431

3. 目标检测

COCO目标检测上，大幅超过同量级的传统模型ResNeXt-101（**超了4.4的mAP**），与Swin相当，在超大量级上达到**55.5%的mAP**。

Table 9. Object detection on COCO. The FLOPs is computed with 1280×800 inputs. The results of ResNeXt-101-64x4d + Cas Mask are reported by [61]. The results of 22K-pretrained Swin (without HTC++ [61]) are reported by [62]. ‡ indicates ImageNet-22K pre-training. ◇ indicates pretrained with extra data.

Backbone	Method	AP^{box}	AP^{mask}	Param (M)	FLOPs (G)
RepLKNet-31B	FCOS	**47.0**	-	87	437
X101-64x4d	FCOS	42.6	-	90	439
RepLKNet-31B	Cas Mask	**52.2**	**45.2**	137	965
X101-64x4d	Cas Mask	48.3	41.7	140	972
ResNeSt-200	Cas R-CNN [9]	49.0	-	-	-
Swin-B	Cas Mask	51.9	45.0	145	982
RepLKNet-31B ‡	Cas Mask	**53.0**	**46.0**	137	965
Swin-B ‡	Cas Mask	**53.0**	45.8	145	982
RepLKNet-31L ‡	Cas Mask	**53.9**	46.5	229	1321
Swin-L ‡	Cas Mask	**53.9**	**46.7**	254	1382
RepLKNet-XL ◇	Cas Mask	**55.5**	**48.0**	392	1958