超大卷积核架构设计与高效实践

MEGVII 旷视

——解读Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs (CVPR 2022)

主讲人：丁霄汉

目录

Contents

1 WHY：关于CNN和Transformer的基本问题

2 HOW：大kernel设计探索与五大准则

3 RepLKNet

4 实验结果

5 讨论与分析

6 总结

MEGVII 旷视

01

关于CNN和Transformer的基本问题

MEGVII 旷视

问题一：与CNN相比，Transformer性能强的原因是什么？

Transformer的基本组件是self-attention，而self-attention的实质是在**全局尺度或较大的窗口内**进行**Query-Key-Value运算**。其中的关键是？

A. 全局尺度或较大的窗口

B. Query-Key-Value运算

证据一：将Swin中的attention换成**7x7卷积**，性能也很强。

证据二：将ViT中的attention换成**MLP**，性能也很强。

证据三：将attention换成**pooling**，性能还是很强。

[1] Qi Han, Zejia Fan, Qi Dai, Lei Sun, Ming-Ming Cheng, Jiaying Liu, and Jingdong Wang. Demystifying local vision transformer: Sparse connectivity, weight sharing, and dynamic weight.
[2] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision.
[3] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision.

MEGVII 旷视

问题二：如果感受野是关键，那么传统CNN差在哪里了？

深层CNN的感受野不是很大吗？

自VGGNet以来，堆叠小kernel成为了主流设计范式

- AlexNet：11x11
- VGGNet：3x3
- ResNet、ResNeXt：一层7x7，大量3x3
- EfficientNet：3x3、5x5
- MobileNet、ShuffleNet：3x3
- RegNet：3x3

MEGVII 旷视

问题二：如果感受野是关键，那么传统CNN差在哪里了？

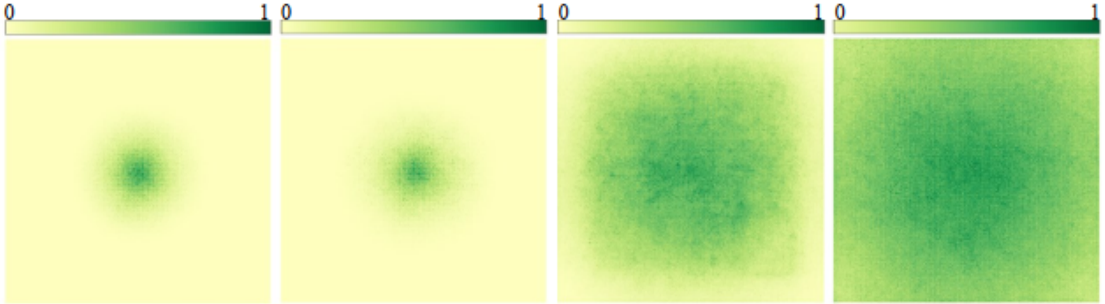
因为我们**相信**，三层3x3比一层7x7要好：

- 3x3x3 < 1x7x7
- 三层3x3可以加入更多非线性，层数越多，拟合能力越强
- **两者的感受野一样大（？）**
- **在ResNet等深度现代模型中，大量3x3堆出来的感受野是足够覆盖全图的（？）**

这是真的吗？

MEGVII旷视

问题二：如果感受野是关键，那么传统CNN差在哪里了？



(A) ResNet-101 (B) ResNet-152 (C) RepLKNet-13 (D) RepLKNet-31

有效感受野可视化

小kernel模型增加深度无法显著增大有效感受野，大kernel模型的有效感受野非常大

有效感受野正比于K$\sqrt{L}$，K为 $\boxed{\text{kernel size}}$，L为深度

Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard S. Zemel. Understanding the effective receptive field in deep convolutional neural networks. NeurIPS 2016.

问题三：为什么我们不用大kernel？

为什么历史淘汰了大kernel？

- **太大**：卷积的参数量和FLOPs与kernel size的平方成正比，如(256, 256, 31, 31)的一个卷积核就有63M参数
- **反而掉点**：
  - [1] GCN：Kx1和1xK大卷积，ImageNet上掉点
  - [2] LR-Net：大kernel，非卷积，kernel size从7x7到9x9即开始掉点

[1] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. CVPR 2017
[2] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. CVPR 2019

MEGVII旷视

问题三：为什么我们不用大kernel？

曾被历史淘汰 ≠ 不应该得到复兴

时代变了：

- **太大**：
  - Depth-wise卷积
  - 恰当的底层优化
- **反而掉点**：
  - 配套的结构设计
  - 重参数化等新方法

MEGVII旷视

问题四：如何在现代模型中复兴大kernel设计？

大kernel不是不能用，而是需要用到好处

How？

MEGVII 旷视

02

“大kernel设计探索与五大准则”

MEGVII旷视

准则一：depth-wise卷积+恰当的底层优化

- 从3x3改到31x31，模型会增大100倍？
- Depth-wise (DW) 卷积占的参数量和FLOPs本来就不多！

Table 5. RepLKNet with different kernel sizes. The models are pretrained on ImageNet-1K in 120 epochs with 224×224 input and finetuned on ADE20K with UperNet in 80K iterations. On ADE20K, we test the *single-scale* mIoU, and compute the FLOPs with input of 2048×512, following Swin.

	ImageNet			ADE20K		
Kernel size	Top-1	Params	FLOPs	mIoU	Params	FLOPs
3-3-3-3	82.11	71.8M	12.9G	46.05	104.1M	1119G
7-7-7-7	82.73	72.2M	13.1G	48.05	104.6M	1123G
13-13-13-13	83.02	73.7M	13.4G	48.35	106.0M	1130G
25-25-25-13	83.00	78.2M	14.8G	48.68	110.6M	1159G
31-29-27-13	83.07	79.3M	15.3G	49.17	111.7M	1170G

准则一：depth-wise卷积+恰当的底层优化

- 加底层优化，速度更快
  - Depth-wise卷积效率低？**DW 3x3效率低，不代表DW 31x31效率低。**
  - 优化得当，可加速20倍

Table 1. Inference speed of a stack of 24-layer depth-wise convolutions with various kernel sizes and resolutions on a single GTX 2080Ti GPU. The input shape is (64, 384, R, R). Baselines are evaluated with Pytorch 1.9.0 + cuDNN 7.6.5, in FP32 precision.

Resolution R	Impl	Latency (ms) @ Kernel size									
		3	5	7	9	13	17	21	27	29	31
16×16	Pytorch	5.6	11.0	14.4	17.6	36.0	57.2	83.4	133.5	150.7	171.4
	Ours	5.6	6.5	6.4	6.9	7.5	8.4	8.4	8.4	8.3	8.4
32×32	Pytorch	21.9	34.1	54.8	76.1	141.2	230.5	342.3	557.8	638.6	734.8
	Ours	21.9	28.7	34.6	40.6	52.5	64.5	73.9	87.9	92.7	96.7
64×64	Pytorch	69.6	141.2	228.6	319.8	600.0	977.7	1454.4	2371.1	2698.4	3090.4
	Ours	69.6	112.6	130.7	152.6	199.7	251.5	301.0	378.2	406.0	431.7

- 已集成进开源框架MegEngine
- 开源的PyTorch实现，参见示例：https://github.com/DingXiaoH/RepLKNet-pytorch

MEGVII 旷视

准则二：加shortcut！

- 实验：MobileNet V2，有/无shortcut，加大到13x13

Table 2. Results of different kernel sizes in normal/shortcut-free MobileNet V2.

Shortcut	Kernel size	ImageNet top-1 accuracy (%)
✓	3×3	71.76
✓	13×13	**72.53**
	3×3	**68.67**
	13×13	53.98

- 解释：shortcut产生了“**组合式**”的感受野；没有shortcut，感受野单一且巨大，难以捕捉小的特征

Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. NeurIPS 2016

MEGVII旷视

准则三：用小kernel做重参数化

- 实验：MobileNet V2，有/无3x3重参数化，加大到9x9或13x13
- 重参数化：
  - 训练时有并行的小kernel，如3x3
  - 训练后等效地将小kernel叠加到大kernel的中心
  - 原理：卷积的可加性

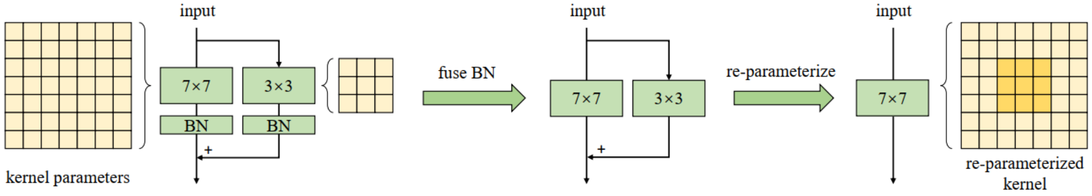


Figure 2. An example of re-parameterizing a small kernel (*e.g.*, 3×3) into a large one (*e.g.*, 7×7). See [27, 30] for details.

Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. **RepVGG**: Making VGG-style ConvNets Great Again.

MEGVII 旷视

准则三：用小kernel做重参数化

- 效果：
  - 无重参数化：9x9涨点，13x13掉点
  - 有重参数化：**13x13涨点**
- 解释：小kernel使组合式的感受野更加丰富，更容易提取小的特征
- 注：在数据量很大（MegData-73M）时，小kernel重参数化（inductive bias）效果不大

Table 3. Results of 3×3 re-parameterization on MobileNet V2 with various kernel sizes.

Kernel	3×3 re-param	ImageNet top-1 acc (%)	Cityscapes val mIoU (%)
3×3	N/A	71.76	72.31
9×9		72.67	76.11
9×9	✓	**73.09**	**76.30**
13×13		72.53	75.67
13×13	✓	**73.24**	**76.60**

Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. **RepVGG**: Making VGG-style ConvNets Great Again.

MEGVII 旷视

准则四：看下游任务！

- 解释：
  - ImageNet分类需要的信息量不大，更大的感受野和更强的表征能力不一定能转化为涨点，但下游需要的语义信息更加丰富。
  - ImageNet模型主要靠纹理来做决定

Table 3. Results of 3×3 re-parameterization on MobileNet V2 with various kernel sizes.

Kernel	3×3 re-param	ImageNet top-1 acc (%)	Cityscapes val mIoU (%)
3×3	N/A	71.76	72.31
9×9		72.67	76.11
9×9	✓	**73.09**	**76.30**
13×13		72.53	75.67
13×13	✓	**73.24**	**76.60**

Table 5. RepLKNet with different kernel sizes. The models are pretrained on ImageNet-1K in 120 epochs with 224×224 input and finetuned on ADE20K with UperNet in 80K iterations. On ADE20K, we test the *single-scale* mIoU, and compute the FLOPs with input of 2048×512, following Swin.

	ImageNet			ADE20K		
Kernel size	Top-1	Params	FLOPs	mIoU	Params	FLOPs
3-3-3-3	82.11	71.8M	12.9G	46.05	104.1M	1119G
7-7-7-7	82.73	72.2M	13.1G	48.05	104.6M	1123G
13-13-13-13	83.02	73.7M	13.4G	48.35	106.0M	1130G
25-25-25-13	83.00	78.2M	14.8G	48.68	110.6M	1159G
31-29-27-13	83.07	79.3M	15.3G	49.17	111.7M	1170G

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness.

大kernel设计探索与五大准则

准则五：小feature map上也能用大kernel ！

- 大kernel一定要用大分辨率？不需要！
- 7x7的feature map上用13x13的大kernel？不严格符合平移不变性
    - 平移不变性一定是好的？
    - 视角一：可以看成一种不同位置卷积核参数不同的特殊卷积
    - 视角二：padding引入了绝对位置信息

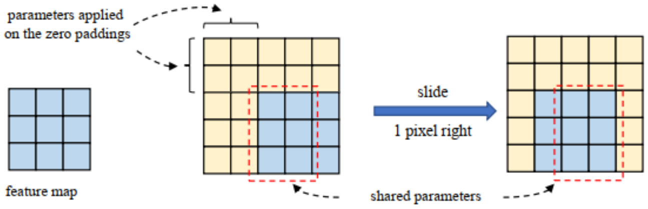


Figure 3. Illustration to convolution with small feature map and large kernel. Two outputs at adjacent locations only share a part of kernel weights. Translational equivariance does not strictly hold.

Table 4. Results of various kernel sizes in the *last stage* of MobileNet V2. Kernel sizes in previous stages remain to be 3 × 3.

Kernel size	ImageNet acc (%)	Cityscapes mIoU (%)
3×3	71.76	72.31
7×7	**72.00**	74.30
13×13	71.97	**74.62**

Osman Semih Kayhan and Jan C van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. CVPR 2020.

大kernel设计探索与五大准则

准则一：depth-wise卷积+恰当的底层优化

准则二：加shortcut！

准则三：用小kernel做重参数化

准则四：看下游任务！

准则五：小feature map上也能用大kernel ！

/ 03

RepLKNet

RepLKNet

- 基于以上准则，设计RepLKNet
- 借鉴Swin的宏观架构
- CNN风格的一点改动

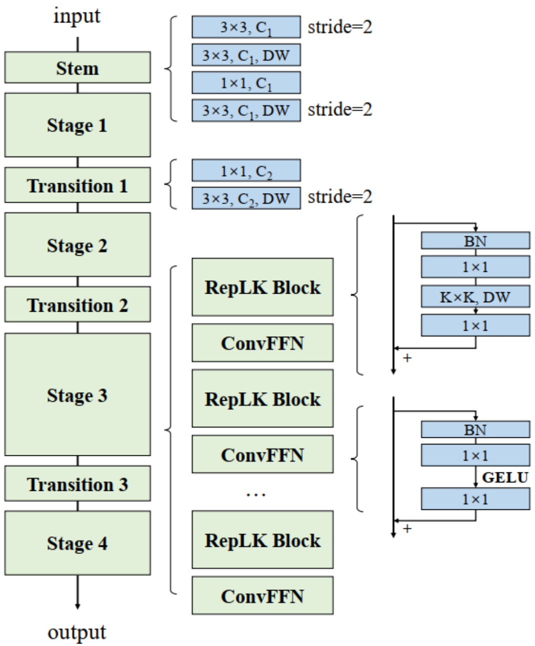


Figure 4. RepLKNet comprises Stem, Stages and Transitions. Except for depth-wise (DW) large kernel, the other components include DW 3×3, dense 1×1 conv, and batch normalization [50] (BN). Note that every conv layer has a following BN, which are not depicted. Such conv-BN sequences use ReLU as the activation function, except those before the shortcut-addition (as a common practice [41, 73]) and those preceding GELU [42].

RepLKNet

MEGVII 旷视

- 加大kernel size!
- RepLKNet-31B：宽度[128, 256, 512, 1024]
- RepLKNet-31L：宽度[192, 384, 768, 1536]
- RepLKNet-XL：宽度[256, 512, 1024, 2048] + 1.5x inverted bottleneck

Table 5. RepLKNet with different kernel sizes. The models are pretrained on ImageNet-1K in 120 epochs with 224×224 input and finetuned on ADE20K with UperNet in 80K iterations. On ADE20K, we test the *single-scale* mIoU, and compute the FLOPs with input of 2048×512, following Swin.

	ImageNet			ADE20K		
Kernel size	Top-1	Params	FLOPs	mIoU	Params	FLOPs
3-3-3-3	82.11	71.8M	12.9G	46.05	104.1M	1119G
7-7-7-7	82.73	72.2M	13.1G	48.05	104.6M	1123G
13-13-13-13	83.02	73.7M	13.4G	48.35	106.0M	1130G
25-25-25-13	83.00	78.2M	14.8G	48.68	110.6M	1159G
31-29-27-13	83.07	79.3M	15.3G	49.17	111.7M	1170G

/ 04

实验结果

实验结果

- ImageNet

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

实验结果

- Cityscapes
  - RepLKNet-31**Base** + ImageNet-**1K** > Swin-**Large** + ImageNet-**22K**

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 [37]. ‡ indicates ImageNet-22K pretraining.

Backbone	Method	mIoU (ss)	mIoU (ms)	Param (M)	FLOPs (G)
RepLKNet-31B	UperNet [98]	**83.1**	**83.5**	110	2315
ResNeSt-200	DeepLabv3	-	82.7	-	-
Axial-Res-XL	Axial-DL [91]	80.6	81.1	173	2446
Swin-B	UperNet	80.4	81.5	121	2613
Swin-B	UperNet + [37]	80.8	81.8	121	-
ViT-L ‡	SETR-PUP [112]	79.3	82.1	318	-
ViT-L ‡	SETR-MLA	77.2	-	310	-
Swin-L ‡	UperNet	82.3	83.1	234	3771
Swin-L ‡	UperNet + [37]	82.7	83.6	234	-

实验结果

- ADE20K
  - 仅ImageNet-1K pretraining, **50.6 mIoU**
  - ViT-L级别 + 73M数据, **56.0 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 [98]	43.8	44.9	86	1029
ResNeSt-200	DeepLabv3	-	48.4	113	1752
Swin-B	UperNet	48.1	49.7	121	1188
Swin-B	UperNet + [37]	48.4	50.1	121	-
ViT-Hybrid	DPT-Hybrid	-	49.0	90	-
ViT-L	DPT-Large	-	47.6	307	-
ViT-B	SETR-PUP [112]	46.3	47.3	97	-
ViT-B	SETR-MLA [112]	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

实验结果

- COCO
  - 与Swin相当
  - 与ResNeXt-101-64x4d相比，体量更小，**AP高4.4**

Table 9. Object detection on COCO. The FLOPs is computed with 1280×800 inputs. The FCOS model is trained with the 2x (24-epoch) training schedule for a fair comparison with the X101 (short for ResNeXt-101) baseline from the same code base [19], and the other results with Cascade Mask R-CNN all use 3x (36-epoch). The results of X101-64x4d + Cas Mask are reported by [60]. The results of 22K-pretrained Swin (without HTC++ [60]) are reported by [61]. ‡ indicates ImageNet-22K pretraining. ◇ 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

/ 05

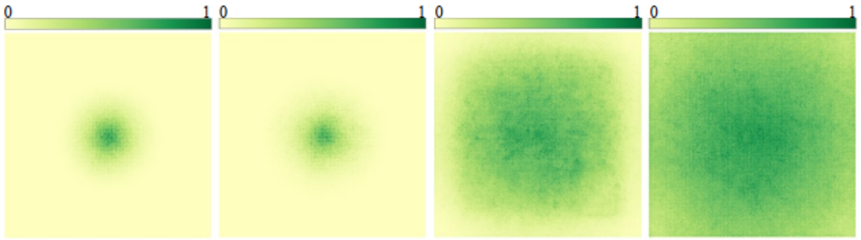
“

讨论与分析

”

讨论与分析

- 有效感受野
  - 定量分析：已经求出了每个像素的贡献值，那么，中间多大的一个区域包含了99%的贡献值？
  - ResNet-101 -> ResNet-152，增加约一半体量
  - RepLKNet-13 -> RepLKNet-31，增加约**7.6%**参数



(A) ResNet-101 (B) ResNet-152 (C) RepLKNet-13 (D) RepLKNet-31

Table 10. Quantitative analysis on the ERF with the high-contribution area ratio r. A larger r suggests a smoother distribution of high-contribution pixels, hence larger ERF.

	$t = 20\%$	$t = 30\%$	$t = 50\%$	$t = 99\%$
ResNet-101	1.0%	1.7%	3.5%	23.4%
ResNet-152	1.3%	2.1%	4.5%	34.9%
RepLKNet-13	10.2%	15.8%	28.5%	96.3%
RepLKNet-31	15.0%	23.4%	41.4%	98.6%

讨论与分析

MEGVII 旷视

- Shape bias：模型有多少决定是根据形状（而非纹理）做出的？
- Shape bias越高，跟人类越像
- 发现：
    - Shape bias跟训练数据关系很大
    - RepLKNet-3和ResNet-152的shape bias几乎一样
    - **Swin的shape bias不高**
    - **RepLKNet-31的shape bias高**
- 已知ViT的shape bias高（参见其论文）
- Shape bias跟感受野关系很大？跟attention关系不大？

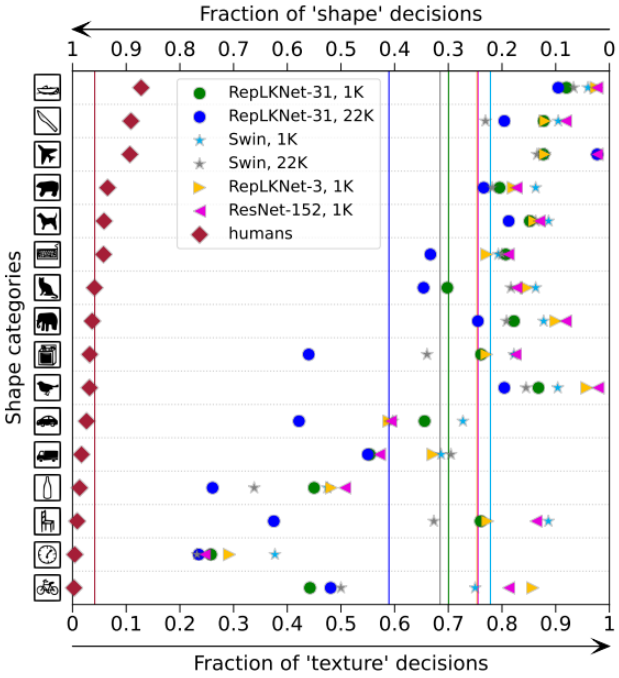


Figure 5. Shape bias of RepLKNet, Swin, and ResNet-152 pre-trained on ImageNet-1K or 22K. The scatters represent the shape bias of 16 categories, and the vertical lines are the averages across categories (note RepLKNet-3 and ResNet-152 are very close).

论文：Shikhar Tuli, Ishita Dasgupta, Erin Grant, and Thomas L Griffiths. Are convolutional neural networks or transformers more like human vision?
开源工具：https ://github.com/bethgelab/model- vs- human

/ 06

总结

总结

- 从小kernel到大kernel
  - Dense conv？太大了！**↓用depthwise** **→还是用3x3吧**
  - Depth-wise大kernel？Depth-wise 3x3都那么慢！ **↓depth-wise大kernel不一定慢** **→还是用3x3吧**
  - 增大kernel size，掉点了？ **↓加shortcut** **→还是用3x3吧**
  - 加了shortcut，从大kernel到超大kernel，又掉点了？ **↓加重参数化/大数据** **→还是用3x3吧**
  - 大kernel肯定要大分辨率啊，训不动！ **↓正常分辨率直接开整** **→还是用3x3吧**
  - ImageNet还是不涨点？**↓再看看下游的** **→还是用3x3吧**

知乎解读

MEGVII 旷视

https://www.zhihu.com/question/517340666/answer/2390516334

论文：https://arxiv.org/abs/2203.06717

代码（MegEngine）：https://github.com/megvii-research/RepLKNet

代码（PyTorch）：https://github.com/DingXiaoH/RepLKNet-pytorch

用人工智能造福大众

MEGVII 旷视