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

Appendix A: Training Configurations

ImageNet-1K

For training MobileNet V2 models (Sec. 3), we use 8 GPUs, an SGD optimizer with momentum of 0.9, a batch size of 32 per GPU, input resolution of 224×224, weight decay of 4×10^{-5}, learning rate schedule with 5-epoch warmup, initial value of 0.1 and cosine annealing for 100 epochs. For the data augmentation, we only use random cropping and left-right flipping, as a common practice.

For training RepLKNet models (Sec. 4.2),we use 32 GPUs and a batch size of 64 per GPU to train for 120 epochs. The optimizer is AdamW [10] with momentum of 0.9 and weight decay of 0.05. The learning rate setting includes an initial value of 4×10^{-3}, cosine annealing and 10-epoch warm-up. For the data augmentation and regularization, we use RandAugment [4] ("rand-m9-mstd0.5-inc1" as implemented by timm [15]), label smoothing coefficient of 0.1, mixup [18] with $\alpha = 0.8$, CutMix with $\alpha = 1.0$, Rand Erasing [19] with probability of 25% and Stochastic Depth with a drop-path rate of 30%, following the recent works [1, 8, 9, 12]. The RepLKNet-31B reported in Sec. 4.3 is trained with the same configurations except the epoch number of 300 and drop-path rate of 50%.

For finetuning the 224×224-trained RepLKNet-31B with 384×384, we use 32 GPUs, a batch size of 32 per GPU, initial learning rate of 4×10^{-4}, cosine annealing, 1-epoch warm-up, 30 epochs, model EMA (Exponential Moving Average) with momentum of 10^{-4}, the same RandAugment as above but *no CutMix nor mixup*.

ImageNet-22K Pretraining and 1K Finetuning

For pretraining RepLKNet-31B/L on ImageNet-22K, we use 128 GPUs and a batch size of 32 per GPU to train for 90 epochs with a drop-path rate of 10%. The other configurations are the same as the aforementioned ImageNet-1K pretraining.

Then for finetuning RepLKNet-31B with 224×224, we use 16 GPUs, a batch size of 32 per GPU, drop-path rate of 20%, initial learning rate of 4×10^{-4}, cosine annealing, model EMA with momentum of 10^{-4} to finetune for 30 epochs. Note again that we use the same RandAugment as above but *no CutMix nor mixup*.

For finetuning RepLKNet-31B/L with 384×384, we use 32 GPUs and a batch size of 16 per GPU, and the drop-path rate is raised to 30%.

RepLKNet-XL and Semi-supervised Pretraining

We continue to scale up our architecture and train a ViT-L [6] level model named RepLKNet-XL. We use $B = [2, 2, 18, 2]$, $C = [256, 512, 1024, 2048]$, $K = [27, 27, 27, 13]$, and introduce inverted bottleneck with expansion ratio of 1.5 to each RepLK Block. During pretraining, we use a private semi-supervised dataset named *MegData73M*, which contains 38 million labeled images and 35 million unlabeled ones. Labeled images come from public and private classification datasets such as ImageNet-1K, ImageNet-22K and Places365 [20]. Unlabeled images are selected from YFCC100M [11]. We design a multi-task label system according to [7], and utilize soft *pseudo* labels which are offline generated by multiple task-specific ViT-Ls wherever human annotations are unavailable. We pretrain our model for up to 15 epochs with similar configurations as ImageNet-1K pretraining. We do *not use CutMix or mixup*, decrease drop-path rate to 20%, and use a lower initial learning rate of 1.5×10^{-3} and a total batch size of 2048. Structural Re-parameterization is omitted because it only brings less than 0.1% performance gain on such a large-scale dataset. In other words, we observe that the inductive bias (re-parameterization with small kernels) becomes less important as the data become bigger, which is similar to the discoveries reported by ViT [6].

We finetune on ImageNet-1K with input resolution of 320×320 for 30 epochs following BeiT [1], except for a higher learning rate of 10^{-4} and stage-wise learning rate decay of 0.4. Finetuning with a higher resolution of 384×384 brings no further improvements. For downstream tasks, we use the default training setting except for a drop-path rate of 50% and stage-wise learning rate decay.

Appendix B: Visualizing the ERF

Formally, let $\mathrm{I}(n \times 3 \times h \times w)$ be the input image, $\mathrm{M}(n \times c \times h' \times w')$ be the final output feature map, we desire to measure the contributions of every pixel on I to

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	0.9%	1.5%	3.2%	22.4%
ResNet-152	1.1%	1.8%	3.9%	34.4%
RepLKNet-13	11.2%	17.1%	30.2%	96.3%
RepLKNet-31	16.3%	24.7%	43.2%	98.6%

the central points of every channel on M, *i.e.*, $\mathrm{M}_{:,:,h'/2,w'/2}$, which can be simply implemented via taking the derivatives of $\mathrm{M}_{:,:,h'/2,w'/2}$ to I with the auto-grad mechanism. Concretely, we sum up the central points, take the derivatives to the input as the pixel-wise contribution scores and remove the negative parts (denoted by P). Then we aggregate the entries across all the examples and the three input channels, and take the logarithm for better visualization. Formally, the aggregated contribution score matrix $\mathrm{A}(h \times w)$ is given by

$$\mathrm{P} = \max(\frac{\partial(\sum_i^n \sum_j^c \mathrm{M}_{i,j,h'/2,w'/2})}{\partial \mathrm{I}}, 0)\,, \tag{1}$$

$$\mathrm{A} = \log_{10}(\sum_i^n \sum_j^3 \mathrm{P}_{i,j,:,:} + 1)\,. \tag{2}$$

Then we respectively rescale A of each model to $[0, 1]$ via dividing the maximum entry for the comparability across models.

Table 10 presents a quantitative analysis, where we report the high-contribution area ratio r of a minimum rectangle that covers the contribution scores over a given threshold t. For examples, 20% of the pixel contributions (A values) of ResNet-101 reside within a 103×103 area at the center, so that the area ratio is $(103/1024)^2 = 1.0\%$ with $t = 20\%$. We make several intriguing observations. **1)** While being significantly deeper, ResNets have much smaller ERFs than RepLKNets. For example, over 99% of the contribution scores of ResNet-101 reside within a small area which takes up only 23.4% of the total area, while such area ratio of RepLKNet-31 is 98.6%, which means most of pixels considerably contribute to the final predictions. **2)** Adding more layers to ResNet-101 does not effectively enlarge the ERF, while scaling up the kernels improves the ERF with marginal computational costs.

Appendix C: Large-Kernel Models have High Shape Bias

A recent work [13] reported that vision transformers are more similar to the human vision systems in that they make predictions more based on the overall shapes of objects, while CNNs focus more on the local textures. We follow its methodology and use its toolbox [2] to obtain the shape bias (*e.g.*, the fraction of predictions made based on the shapes, rather than the textures) of RepLKNet-31B and Swin-B pretrained on ImageNet-1K or 22K, together two small-kernel baselines RepLKNet-3 and ResNet-152. Fig. 5 shows that RepLKNet has higher shape bias than Swin. Considering RepLKNet and Swin have similar overall architectures, we reckon shape bias is closely related to the *Effective Receptive Field* rather than the concrete formulation of self-attention (*i.e.*, the *query-key-value* design). This also explains **1)** the high shape bias of *ViTs* [6] reported by [13] (since ViTs employ global attention), **2)** the low shape bias of 1K-pretrained Swin (attention within local windows), and **3)** the shape bias of the small-kernel baseline RepLKNet-3, which is very close to ResNet-152 (both models are composed of 3×3 convolutions).

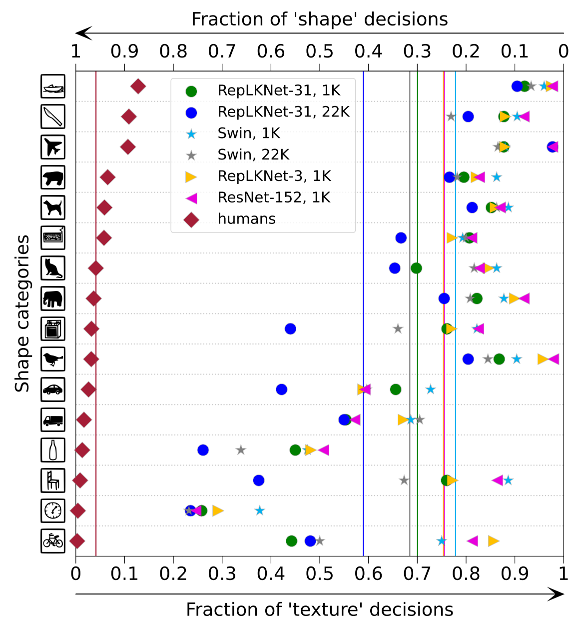


Figure 5. Shape bias of RepLKNet, Swin, and ResNet-152 pretrained on ImageNet-1K or 22K. The scatters represent the shape bias of 16 categories, and the vertical lines are the averages across categories (note RepLKNet-3 and ResNet-152 are very close).

Appendix D: ConvNeXt + Very Large Kernels

We use the recently proposed ConvNeXt [9] as the benchmark architecture to evaluate large kernel as a *generic design element*. We simply replace the 7×7 convolutions in ConvNeXt [9] by kernels as large as 31×31. The training configurations on ImageNet (120 epochs) and ADE20K (80K iterations) are identical to the results shown in Sec. 4.2. Table. 11 shows that though the original kernels are already 7×7, further increasing the kernel sizes still brings significant improvements, especially on the downstream task: with kernels as large as 31×31, ConvNeXt-Tiny outperforms the original ConvNeXt-Small, and the large-kernel ConvNeXt-Small outperforms the original ConvNeXt-Base. Again, such phenomena demonstrate

Table 11. ConvNeXt 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	Architecture	Top-1	Params	FLOPs	mIoU	Params	FLOPs
7-7-7-7	ConvNeXt-Tiny	81.0	29M	4.5G	44.6	60M	939G
7-7-7-7	ConvNeXt-Small	82.1	50M	8.7G	45.9	82M	1027G
7-7-7-7	ConvNeXt-Base	82.8	89M	15.4G	47.2	122M	1170G
31-29-27-13	ConvNeXt-Tiny	81.6	32M	6.1G	**46.2**	64M	973G
31-29-27-13	ConvNeXt-Small	82.5	58M	11.3G	**48.2**	90M	1081G

Table 12. MobileNet V2 with all regular DW 3×3 layers replaced by 3×3 dilated layers.

Max RF	Kernel size	Dilation	ImageNet acc	Params	FLOPs
9	9×9	-	72.67	4.0M	319M
9	3×3	4	57.23	3.5M	300M
13	13×13	-	72.53	4.6M	361M
13	3×3	6	51.21	3.5M	300M

that kernel size is an important scaling dimension.

Appendix E: Dense Convolutions *vs*. Dilated Convolutions

As another alternative to implement large convolutions, *dilated convolution* [3, 17] is a common component to increase the *receptive field (RF)*. However, Table 12 shows though a depth-wise dilated convolution may have the same maximum RF as a depth-wise dense convolution, its representational capacity is much lower, which is expected because it is mathematically equivalent to a *sparse* large convolution. Literature (*e.g*., [14, 16]) further suggests that dilated convolutions may suffer from *gridding problem*. We reckon the drawbacks of dilated convolutions could be overcome by mixture of convolutions with different dilations, which will be investigated in the future.

Appendix F: Visualizing the Kernel Weights with Small-Kernel Re-parameterization

We visualize the weights of the re-parameterized 13×13 kernels. Specifically, we investigate into the MobileNet V2 models both with and without 3×3 re-parameterization. As Shown in Sec. 3 (Guideline 3) , the ImageNet scores are 73.24% and 72.53%, respectively. We use the first stride-1 13×13 conv in the last stage (*i.e*., the stage with input resolution of 7×7) as the representative, and aggregate (take the absolute value and sum up across channels) the resultant kernel into a 13×13 matrix, and respectively rescale to $[0, 1]$ for the comparability. For the model with 3×3 re-param, we show both the original 13×13 kernel (only after BN fusion) and the result after re-param (*i.e*., adding the 3×3 kernel onto the central part of 13×13). For the model without re-param, we also fuse the BN for the fair comparison.

We observe that every aggregated kernel shows a similar pattern: the central point has the largest magnitude; generally, points closer to the center have larger values; and the "skeleton" parameters (the 13×1 and 1×13 criss-cross parts) are relatively larger, which is consistent with the discovery reported by *ACNet* [5]. But the kernel with 3×3 re-param differs in that the central 3×3 part of the resultant kernel is further enhanced, which is found to improve the performance.

References

[1] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 1

[2] bethgelab. Toolbox of model-vs-human. https://github.com/bethgelab/model-vs-human, 2022. 2

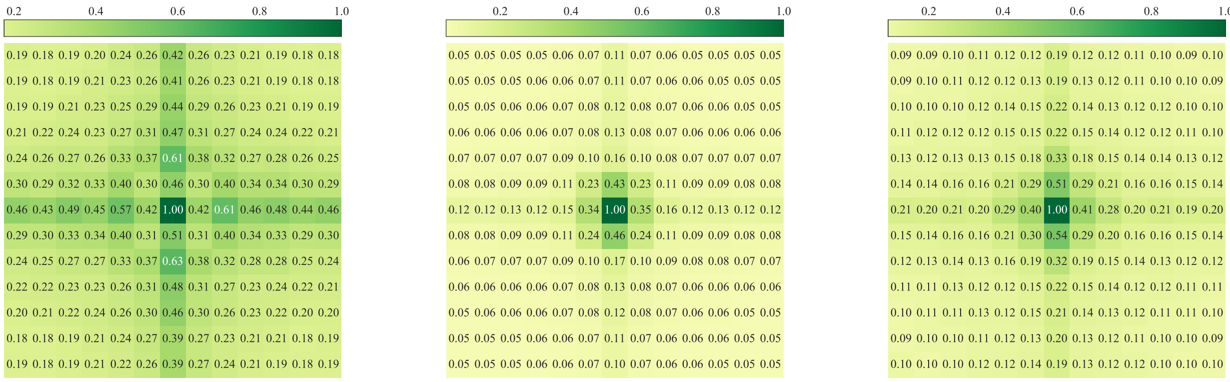
[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 3

[4] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 1

[5] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1911–1920, 2019. 3

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 1, 2

[7] Golnaz Ghiasi, Barret Zoph, Ekin D Cubuk, Quoc V Le, and Tsung-Yi Lin. Multi-task self-training for learning general representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8856–8865, 2021. 1



(a) Before re-param. (b) After re-param. (c) Without re-param.

Figure 6. Parameters of 13×13 kernels in MobileNet V2 aggregated into 13×13 matrices.

[8] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 1

[9] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *arXiv preprint arXiv:2201.03545*, 2022. 1, 2

[10] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[11] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. 1

[12] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 1

[13] Shikhar Tuli, Ishita Dasgupta, Erin Grant, and Thomas L Griffiths. Are convolutional neural networks or transformers more like human vision? *arXiv preprint arXiv:2105.07197*, 2021. 2

[14] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 1451–1460. Ieee, 2018. 3

[15] Ross Wightman. Timm implementation of randaugment. https://github.com/rwightman/pytorch-image-models/blob/master/timm/data/auto_augment.py, 2022. 1

[16] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *arXiv preprint arXiv:2105.15203*, 2021. 3

[17] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 3

[18] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 1

[19] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020. 1

[20] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 1