# DCB: An Efficient Approach for Building Long-Range Dependencies in CNNs

Tianxiang Lan, Mingyi He* and Yuchao Dai
Northwestern Polytechnical University, Xi'an, China
tianxiang2023@mail.nwpu.edu.cn; myhe@nwpu.edu.cn; daiyuchao@nwpu.edu.cn
*: Corresponding author

***Abstract*—Large-kernel convolution has garnered renewed attention in convolutional neural networks (CNNs) in recent years because of its ability to enhance model performance. However, the computational cost and the number of parameters caused by large-kernel convolution grow quadratically with kernel size, hindering its practical deployment. To address this challenge, we propose a novel cross-linear layer. Compared to standard large-kernel convolution, the cross-linear layer utilizes fewer parameters while efficiently creating feature connections in the spatial domain. In addition, we introduce a secondary propagation strategy to further enhance the cross-linear layer. This strategy can further extend connections among local features over a wide range, leading to the development of a Double Cross Block (DCB). The DCB combines the cross-linear layer and the secondary propagation strategy to efficiently construct local feature connections and significantly reduce computational cost. Furthermore, DCB improves network performance. We validate the effectiveness of the DCB module through extensive experiments on multiple datasets and demonstrate its applicability to state-of-the-art (SOTA) network models. The results indicate that the DCB can serve as an efficient alternative to large-kernel convolution layers with fewer parameters and lower computational complexity.**

## I. Introduction

The convolutional neural network (CNN) is an important architecture in deep learning and has attracted significant attention due to its outstanding performance in computer vision [1]–[4]. However, since 2021, CNNs have been challenged by Transformers [5]–[9]. Transformer-based networks have achieved excellent performance not only in the field of image classification but also in many tasks such as object detection, semantic segmentation, and super-resolution. The researchers analyzed the reason why the Transformer performs better. Liu *et al.* [10] attribute this primarily to the more advanced approach of Transformers. Drawing inspiration from [8], they propose ConvNeXt. Ding *et al.* [11] attribute Transformers' effectiveness to their ability to establish larger effective receptive fields (ERF). Their study showed that large-kernel convolution can effectively expand the ERF to be comparable to that of the Transformer, improving model accuracy across various benchmarks. Some subsequent work [12]–[15] based on large-kernel convolution has confirmed this finding. Although large kernels can improve CNNs performance, their computational cost and parameter count grow quadratically with kernel size, a fundamental limitation inherent to this approach. The resulting increase in computational demands and parameters remains an unavoidable trade-off.

This work focuses on reducing computational cost and the growth of parameters induced by large kernels. Based on recent studies [14], [16], [17], we found that the primary advantage of large kernels lies in their ability to build connections between local and distant features, creating long-distance dependencies that improve network performance. Thus, the key challenge becomes how to build local-to-distant feature interactions effectively while minimizing the computational and parametric costs. To address this challenge, we propose a simple but efficient approach to simplify large-kernel convolution: the Double Cross Block (DCB). This method builds connections among local features, achieves better performance than large-kernel convolution, and significantly reduces computational cost and parameter count, leading to a lighter network.

Our main contributions are summarized as follows.

- We propose a cross-linear layer, a simple yet effective structure composed of linear layers, which builds long-distance dependencies within features with fewer parameters.
- To address the limitation of the cross-linear layer, we introduce a secondary propagation strategy to enhance the modeling capability of the cross-linear layer.
- We propose a new plug-and-play block, called the Double Cross Block (DCB), which can replace the large-kernel convolution in models.
- We construct DemoNet, an experimental network to evaluate the DCB on multiple datasets and validate the effectiveness of our proposed module.

## II. Related Work

### A. Large-Kernel Convolution

Early CNNs like [1], [2], [4], [18]–[20] employ large-kernel convolution in shallow layers for low-level feature extraction. VGGNet [21] empirically established that stacking multiple small-kernel convolutional layers achieves superior performance compared to using larger kernels, setting a new architectural standard for subsequent CNNs.

Contrary to the prevailing trend, [11] re-established the significance of large-kernel convolutions, demonstrating their advantages in capturing global spatial relationships and improving model performance in vision tasks. Similarly to a

979-8-3315-7206-8/25/$31.00 

transformer, large-kernel convolution can effectively improve the ERF of CNNs. Therefore, Ding *et al.* [22] proposed RepLKNet, which uses a maximum convolution kernel size of $31 \times 31$ and incorporates a reparameterization method.

Following [11], researchers have increasingly recognized the efficacy of large-kernel convolution. [16] has replaced the kernel $K \times K$ with two factorized rectangular kernels ($K \times N$ and $N \times K$, where $N < K$), expanding the kernel size to $51 \times 51$ and improving model performance. [23] has introduced the Large Kernel Attention (LKA) mechanism to capture long-distance dependencies within features and proposed the Visual Attention Network (VAN). [24] utilized large-kernel convolution to capture contextual information from medical images without using Transformers, improving the semantic segmentation network. [25] utilized a $7 \times 7$ depthwise convolution (DWConv) to extract features between star operation blocks, thereby achieving better performance.

However, large-kernel convolution remains computationally prohibitive due to its quadratic complexity scaling with kernel size $K$ ($O(K^2)$), creating significant barriers to network miniaturization.

### B. Lightweight Large Convolutions

Some studies [14], [16], [17], [26] have focused on overcoming the computational cost of large-kernel convolution. Through empirical studies, Liu *et al.* [16] observed that overly large convolutional kernels can impair network performance by compromising locality, leading them to propose decomposing large square kernels into two rectangular ones to maintain locality while capturing long-range dependencies. Inspired by the peripheral vision mechanism in humans, which is clear in the center and blurred in the periphery, [14] proposed the Parameter-efficient Large Kernel Network (PeLK). This architecture reduced the number of parameters while mitigating overfitting induced by excessive kernel expansion. However, this method is complicated and does not reduce computational complexity. [17] utilized two convolutions (a kernel $1 \times K$ and a kernel $K \times 1$) connected in series to build long-distance dependencies. Although it slightly reduced computational complexity and improved the performance of the semantic segmentation network, it did not clarify the underlying mechanism and lacked nonlinearity. None of these methods achieves a lightweight substitution for large-kernel convolutions. Implementing large-kernel convolutions in lightweight models remains a challenge.

## III. Method

### A. Cross-Linear Layer

Recent studies [11], [14], [16] have shown that in large-kernel convolutions, the weights located along the x-axis and y-axis of the convolution kernels have higher values, indicating that these values are more important than the surrounding weights and can greatly affect network performance. We hypothesize that these weights govern the establishment of correlations among local features. Therefore, based on this assumption, we propose a novel cross-linear layer to establish correlations among local features.

The cross-linear layer has two linear branches, as shown in Figure 1. Assume that the input feature is $F_1$. We use a sliding window with a kernel length of $K$ to extract a vector $v_i$ along the x-axis of $F_1$ as input to one linear layer. Let $x_i$ denote the point obtained after $v_i$ passes through the linear layer. $x_i$ is the feature point that forms the new feature map $F_2$, and it corresponds to a linear-shaped area along the x-axis of $F_1$. Following the same steps along the y-axis, we obtain another new feature map $F_3$. We merge the two feature maps ($F_2$ and $F_3$) to obtain the final result $F_{out}$. The process is shown in Equations 1-3. Similar to a Depthwise Convolution (DWConv), the cross-linear layer can also be extended to the depthwise domain. This extension is called the DW mode.

$$v_1 = Slide_x(F_1) \qquad v_2 = Slide_y(F_1) \tag{1}$$

$$F_2 = Linear_x(v_1) \qquad F_3 = Linear_y(v_2) \tag{2}$$

$$F_{out} = F_2 + F_3 \tag{3}$$

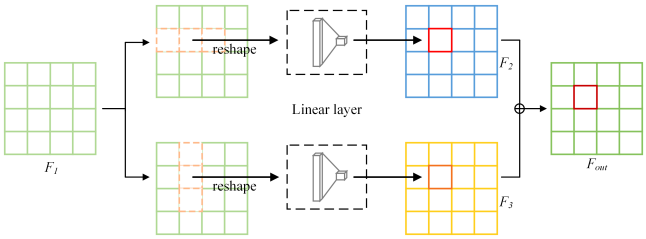



Fig. 1: Cross-Linear Layer

The computational cost and the parameter count of standard convolution scale quadratically with the kernel size, whereas those of the cross-linear layer scale linearly with it. The calculation process is shown in Equations 4-7. Due to the low computational cost of merging two vectors, this operation is omitted.

$$Standard_{params} = K^2 \times C_{in} \times C_{out} \tag{4}$$

$$Standard_{flops} = K^2 \times H \times W \times C_{in} \times C_{out} \tag{5}$$

$$Cross_{params} = 2 \times K \times C_{in} \times C_{out} \tag{6}$$

$$Cross_{flops} = 2 \times K \times C_{in} \times C_{out} \times H \times W \tag{7}$$

Here, $K$ is the kernel size of the standard convolution and cross-linear layer. $H$ and $W$ are the height and width of the input tensor. $C_{in}$ and $C_{out}$ denote the number of input and output channels. So, under equivalent parameter budgets compared to large-kernel convolution, the cross-linear layer

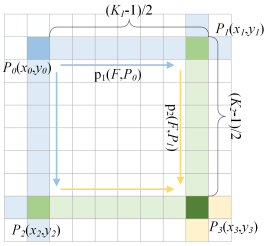



Fig. 2: Schematic Diagram of Secondary Propagation Strategy

enables broader feature integration, while its local feature connectivity improves model performance, manifesting as the expansion of the ERF.

However, this method has a limitation: the connections constructed by the cross-linear layer can only cover a cross-shaped area and fail to reach the remaining features beyond, resulting in a decrease in network performance. To address this limitation, we introduce a secondary propagation strategy [27].

### B. Secondary Propagation Strategy

Denote $F'$ as the feature after $F$ is processed by a cross-linear layer. $F'$ then passes through the second cross-linear layer, producing $F''$. $F$, $F'$, and $F''$ all have the same size. To ensure universality, assume that the kernel size in the first cross-linear layer is $K_1$ and that in the second one is $K_2$, as shown in Figure 2. In Figure 2, the blue area represents the propagation process of the first cross-linear layer, the yellow area is the second propagation process, and the green area represents the overlapping region between the first and second propagation processes. The darker areas indicate the intersection regions of the cross-layer propagation. Denote $P_0$ as a point in $F$. $P_1$ and $P_2$ are the results of $P_0$ after the first propagation $p_1$. $P_3$ is the result of the second propagation $p_2$.

$p_1$ propagates $P_0$ horizontally and vertically to the points $P_1$ and $P_2$ in $F'$, respectively. Then after $p_2$, the results of $P_1$ and $P_2$ converge at $P_3$. This process is expressed in Equations 8. With the secondary propagation strategy, local features can further build connections with more distant features, greatly improving the globality of the module.

$$\begin{aligned} F'(P_1), F'(P_2) &= p_1(F, P_0) \\ F''(P_3) &= p_2(F', P_1, P_2) \end{aligned} \tag{8}$$

### C. Double Cross Block (DCB)

Based on the cross-linear layer and the secondary propagation strategy, we propose a novel approach, the Double Cross Block (DCB). The architecture of DCB is illustrated in Figure 3. First, DCB uses a $3 \times 3$ convolution layer to extract local features from the input tensor, then uses two cross-linear layers named Cross Blocks to build long-term connections. The Cross Block has two branches: Row Linear (RL) and Column Linear (CL), corresponding to the x-axis and the y-axis respectively. The first Cross Block can build connections in the cross-shaped area. The second Cross Block further diffuses connections to a larger area, so that the receptive field spreads to the surroundings. After the first Cross-Block, we employ Batch Normalization (BN) to stabilize module training and incorporate GELU activation to enhance the nonlinearity of DCB. Compared to large-kernel convolution, DCB achieves a connection effect similar to large-kernel convolution while requiring significantly fewer parameters due to its use of cross-linear layers.

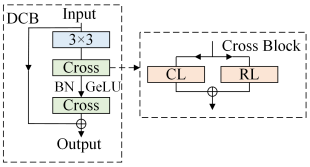



Fig. 3: DCB and Cross Block

### D. Complexity Analysis

For standard convolution, we have obtained its parameter count and computational cost from Equations 4-5. Those of DCB can be calculated as shown in Equations 9-10, respectively.

$$\begin{aligned} DCB_{params} =& 3^2 \times C_{in} \times C_{out} + 4 \times K \times C_{in} \times C_{out} \\ =& (9 + 4K) \times C_{in} \times C_{out} \end{aligned} \tag{9}$$

$$\begin{aligned} DCB_{flops} =& 3^2 \times H \times W \times C_{in} \times C_{out} \\ &+ 4 \times K \times H \times W \times C_{in} \times C_{out} \\ =& (9 + 4K) \times H \times W \times C_{in} \times C_{out} \end{aligned} \tag{10}$$

From Equations 4-5, 9-10, it is clear that the computation cost is directly proportional to the number of parameters. Therefore, reducing the number of parameters will proportionally reduce the computational cost.

Compared to standard convolution, the computational ratio $\alpha$ can be calculated as shown in Equation 11.

$$\alpha = \frac{DCB_{flops}}{Standard_{flops}} = \frac{4}{K} + \frac{9}{K^2} \tag{11}$$

$\alpha \geq 1.16$ when $K \leq 5$, which means that the DCB module introduces additional parameters to the standard convolution. However, the efficiency of DCB will be better when $K = 6$, $\alpha \approx 0.92$. As $K$ increases, DCB demonstrates more significant reductions in both computational cost and parameter count.

## IV. Experiments and Results

### A. Datasets and Settings

We used the common datasets CIFAR10, CIFAR100, and SVHN for our experiments. The experiments are conducted on a server with an Intel(R) Xeon(R) CPU E5-2640 v4, an NVIDIA GeForce GTX 1080 Ti GPU, running Ubuntu 18.04.1. All networks trained on the same datasets use the same hyper-parameters and are trained from scratch.

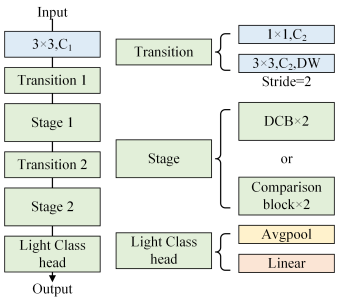



Fig. 4: Structure of DemoNet

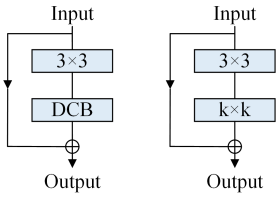



Fig. 5: DCB and Comparison Block

### B. DemoNet

Since modern network backbones are typically designed for larger input sizes, their inherent local feature connections are significantly larger than the image sizes of our target datasets. Using such networks as experimental baselines would obscure the actual performance gains. To better evaluate the effectiveness of DCB, we designed a network architecture named DemoNet for controlled evaluation. Its architecture is illustrated in Figure 4.

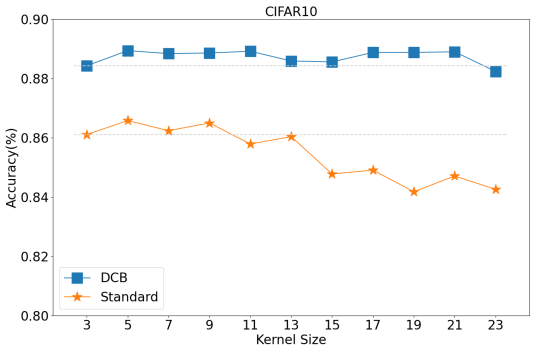
To clearly illustrate the effectiveness of DCB, we built two similar blocks that utilize DCB and large-kernel convolution, respectively, as shown in Figure 5.

### C. Results

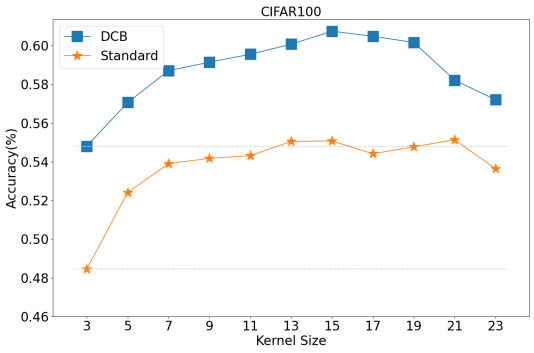
Figure 6 shows the network performance with different $K$ in different datasets. Models employing DCB consistently outperform their large-kernel counterparts in varying kernel dimensions. Figure 7 presents the parameters and the MACs (Multiply-Accumulate Operations) variation with respect to the kernel size. DCB exhibits linear parameter growth in contrast to the exponential scaling characteristic of large-kernel convolution, thereby confirming our theoretical analysis. The results demonstrate that DCB can effectively reduce the computational cost and improve the performance of the models. The joint observations from Figures 6-7 conclusively demonstrate the DCB's dual advantages: superior accuracy with parameter efficiency, establishing its computational superiority over standard large-kernel approaches.

### D. Performance of DCB in SOTA Models

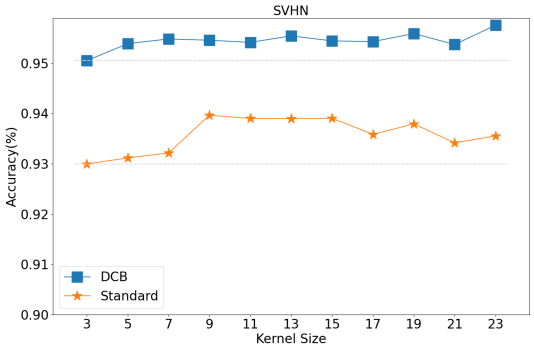
We evaluated DCB in SOTA architectures (RepLKNet [11] and ConvNeXt [10]). To ensure a fair comparison, we disabled structural reparameterization in RepLKNet, as this method is not employed in other architectures. Since the base architectures use DWConv for feature extraction, we also adjusted the DCB to the DW mode. The results are presented in Table I. DCB significantly reduces parameter counts and computational costs while improving Top-1 accuracy across all tested architectures. These results demonstrate the practical utility of DCB. The parameter reduction scales directly with both the kernel size and the frequency of large-kernel convolution usage, aligning closely with our theoretical analysis.




(a) CIFAR10




(b) CIFAR100




(c) SVHN

Fig. 6: DemoNet Results on CIFAR10, CIFAR100 and SVHN

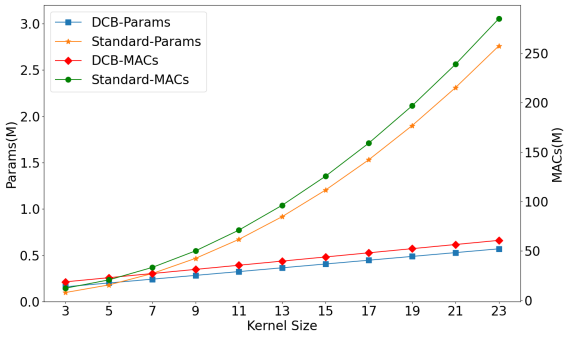



Fig. 7: DemoNet Parameters and MACs Change with $K$

TABLE I: Results of DCB on Different Models

| Dataset | Model | Top1 acc (%) | Parameters (M) |
|---|---|---|---|
| CIFAR10 | RepLK($K$=7) | 91.8 | 0.566 |
| | RepLK($K$=7,DCB) | **92.2** | **0.562** |
| | RepLK($K$=13) | 91.3 | 0.635 |
| | RepLK($K$=13,DCB) | **91.7** | **0.576** |
| | ConvNeXt-tiny | 75.3 | 27.821 |
| | ConvNeXt-tiny(DCB) | **81.4** | **27.775** |
| | ConvNeXt-small | 78.9 | 49.449 |
| | ConvNeXt-small(DCB) | **82.8** | **49.354** |
| CIFAR100 | RepLK($K$=7) | 70.8 | 0.578 |
| | RepLK($K$=7,DCB) | **71.1** | **0.574** |
| | RepLK($K$=13) | 68.2 | 0.647 |
| | RepLK($K$=13,DCB) | **71.6** | **0.588** |
| | ConvNeXt-tiny | 49.2 | 27.891 |
| | ConvNeXt-tiny(DCB) | **57.4** | **27.844** |
| | ConvNeXt-small | 45.9 | 49.518 |
| | ConvNeXt-small(DCB) | **59.1** | **49.424** |

### *E. Ablation Studies*

To validate the effectiveness of the secondary propagation strategy in DCB, we conducted ablation studies using the same experimental setup as the baseline evaluations. Figure 8 compares the DCB module (left) with the SCB (Single Cross Block) module (right). The SCB comprises only a single cross-linear layer.

As evidenced by the results in Table II, the secondary propagation strategy yields significant performance improvements, confirming its critical role in the proposed architecture. Although the secondary propagation strategy incurs higher computational costs, its total cost remains significantly lower than that of the large-kernel convolution. Therefore, after a comprehensive trade-off analysis, we adopted this strategy.

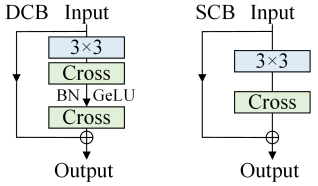



Fig. 8: Comparison of the Ablation Experiment Modules

TABLE II: Ablation Study Results

| Kernel Size | Params (M) | | | Top-1 Accuracy (%) | | |
|---|---|---|---|---|---|---|
| | Standard | SCB | DCB | Standard | SCB | DCB |
| 5 | 0.18 | 0.15 | 0.20 | 86.58 | 86.30 | **88.94** |
| 11 | 0.67 | 0.21 | 0.32 | 85.79 | 88.45 | **88.92** |
| 17 | 1.53 | 0.27 | 0.45 | 84.91 | 88.21 | **88.88** |
| 23 | 2.76 | 0.33 | 0.57 | 84.26 | 88.13 | **88.24** |

## V. Conclusion

This paper proposes a novel cross-linear layer, a parameter-efficient method designed for the spatial integration of features. Based on the cross-linear layer, we propose the DCB module, which employs a secondary feature propagation strategy to enhance the performance of the cross-linear layer. We compare the DCB module with standard large-kernel convolutions to validate the efficacy of DCB. Experiments on multiple datasets demonstrate that DCB can reduce the number of parameters and computational costs while maintaining a comparable level of accuracy, which validates both our theoretical analysis and its practical efficacy. Therefore, DCB can serve as an efficient alternative to large-kernel convolution layers. DCB can be used in classification networks and may also be integrated into other tasks such as semantic segmentation, object detection, super-resolution, and image denoising to improve model performance.

## References

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, IEEE Computer Society, 2016, pp. 770–778.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

[4] C. Szegedy, W. Liu, Y. Jia, *et al.*, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, IEEE Computer Society, 2015, pp. 1–9.

[5] V. Ashish, S. Noam, P. Niki, *et al.*, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17, Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010, ISBN: 9781510860964.

[6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "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.

[7] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I*, A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, Eds., ser. Lecture Notes in Computer Science, vol. 12346, Springer, 2020, pp. 213–229.

[8] Z. Liu, Y. Lin, Y. Cao, *et al.*, "Swin transformer: Hierarchical vision transformer using shifted windows," in *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, IEEE, 2021, pp. 9992–10 002.

[9] A. Srinivas, T. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, Computer Vision Foundation / IEEE, 2021, pp. 16 519–16 529.

[10] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, IEEE, 2022, pp. 11 966–11 976.

[11] X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31×31: Revisiting large kernel design in cnns," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 11 953–11 965.

[12] Y. Chen, X. Hu, T. Lu, L. Zou, and X. Liao, "A multi-scale large kernel attention with u-net for medical image registration," *J. Supercomput.*, vol. 81, no. 1, p. 70, 2025.

[13] H. Bai, "Inception-like large kernel network for lightweight image super-resolution," *Multim. Syst.*, vol. 31, no. 1, p. 78, 2025.

[14] H. Chen, X. Chu, Y. Ren, X. Zhao, and K. Huang, "Pelk: Parameter-efficient large kernel convnets with peripheral convolution," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, IEEE, 2024, pp. 5557–5567.

[15] A. Wang, H. Chen, Z. Lin, J. Han, and G. Ding, "Lsnet: See large, focus small," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, Computer Vision Foundation / IEEE, 2025, pp. 9718–9729.

[16] S. Liu, T. Chen, X. Chen, *et al.*, "More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity," in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, OpenReview.net, 2023.

[17] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, "Large kernel matters - improve semantic segmentation by global convolutional network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, IEEE Computer Society, 2017, pp. 1743–1751.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, IEEE Computer Society, 2014, pp. 580–587.

[19] R. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015. arXiv: 1504.08083.

[20] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, IEEE Computer Society, 2015, pp. 3431–3440.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

[22] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, "Repvgg: Making vgg-style convnets great again," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, Computer Vision Foundation / IEEE, 2021, pp. 13 733–13 742.

[23] M. Guo, C. Lu, Z. Liu, M. Cheng, and S. Hu, "Visual attention network," *Comput. Vis. Media*, vol. 9, no. 4, pp. 733–752, 2023.

[24] F. Tang, J. Ding, Q. Quan, L. Wang, C. Ning, and K. Zhou, "CMUNEXT: an efficient medical image segmentation network based on large kernel and skip fusion," in *IEEE International Symposium on Biomedical Imaging, ISBI 2024, Athens, Greece, May 27-30, 2024*, IEEE, 2024, pp. 1–5.

[25] X. Ma, X. Dai, Y. Bai, Y. Wang, and Y. Fu, "Rewrite the stars," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, IEEE, 2024, pp. 5694–5703.

[26] D. Li, L. Li, Z. Chen, and J. Li, "Shiftwiseconv: Small convolutional kernel with large kernel effect," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, Computer Vision Foundation / IEEE, 2025, pp. 25 281–25 291.

[27] Z. Huang, X. Wang, Y. Wei, *et al.*, "Ccnet: Criss-cross attention for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 6896–6908, 2023.