DSAN: Exploring the Relationship between Deformable Convolution and Spatial Attention

Zewen Yu, Xiaoqin Zhang, *Senior Member, IEEE*, Li Zhao and Guobao Xiao, *Senior Member, IEEE*

***Abstract*—Recently, deformable convolutional neural network is commonly used in computer vision tasks, achieving remarkable results. The existing method DCNv3, focuses more on heavyweight models rather than lightweight ones. These heavyweight models are not suitable for small computing devices, which are limited by their hardware to deploy lightweight convolutional neural networks (CNNs). In this article, we focus on applying the DCNv3 operation to lightweight CNNs. To explore the performance of lightweight CNNs based on DCNv3, we conduct experiments and find that DCNv3 does not fully utilize its advantages with lightweight CNNs due to sparse sampling. Yet the traditional solution of increasing kernel size boosts computational load, making it unsuitable. Based on this situation, we solve this dilemma from two levels, the core operation and the visual feature extraction module. At the core operation level, we propose Deformable Strip Convolution (DSCN). As a simplified version of DCNv3 with large kernel, DSCN has only 63.2% computational load of the original with respect to the deformable sampling method. DSCN further avoids a quadratic increase in computational load with kernel size by limiting the deformation sampling kernels to single axis. At the visual feature extraction module level, we propose Deformable Spatial Attention (DSA) constructed from DSCN as a replacement for DCNv3. Specifically, we observe the similarity between the modulation mask branch in DCNv3 and spatial attention, and use spatial attention instead of modulation mask branch based on this similarity to reduce parameters and memory consumption. Finally, in order to verify the effectiveness of our improved design, we further propose a lightweight CNN backbone named DSAN. After conducting numerous extensive experiments, we find that DSA has an inference speed that is 2.1 times faster than that of DCNv3 with large kernel. In dense prediction tasks such as semantic segmentation, DSAN-S with a lightweight decoder achieves 48.8% mIoU on ADE20K, which is higher than the result of InternImage-T based on DCNv3 with a heavyweight decoder, while the number of parameters and computation is only 35.0% and 9.1% of its. Our code is available at https://github.com/MarcYugo/DSAN-Deformable-Spatial-Attention.**

***Index Terms*—Deep neural network, vision fundation models, deformable convolution, spatial attention mechanism**

I. INTRODUCTION

In computer vision tasks, convolutional neural networks (CNNs) has been proven to be very important. From the perspective of the characteristics of visual data, such as images, target objects have three basic attributes: position, size, and shape. These characteristics provide criteria for judging whether a CNN is a good neural network for vision tasks. That is, a good convolutional neural network should have adaptability to object position, size, and shape [1]. However, the design of vanilla convolution operation only considers position adaptability. Therefore, the improvement works on convolution operations mainly focus on enhancing size and shape adaptability.

The adaptability of models to object size changes implies that models should extract the appropriate features from objects of varying size. For vanilla convolution operations, their small sampling range limits their capability to handle objects of different sizes. The mainstream improvement is enlarging the size of sampling range on spatial domain, like using larger kernel size. The effectiveness of increasing the convolution kernel is confirmed by RepLKNet [2], dilation convolution [3], LKA [4] and MSCA [5].

Unlike the method of improving the object size adaptability, the strategy of enlarging the sampling range has a very limited impact on the object shape adaptability of CNNs. The fact is proved by the work Deformable Attention Transformer [6]. ViT-based deep neural networks still are constrained by the regularly sampling, even their sampling is the whole spatial domain. The methods of enhancing the shape adaptability of convolution operations are mainly achieved by designing irregular convolution kernel shapes or rather changing the sampling strategies. These works includes the Deformable Convolution (DCN) series [1], [7], [8], DSC [9], DeBut [10], DIKS [11] and KPN [12], etc. The recent work InternImage [8], a backbone based on DCNv3 has good size and shape adaptability at the same time, achieving SOTA on several computer vision task datasets. Still, the smallest InternImage-T in the existing InternImage series has near 30M parameters, lacks suitability for lightweight CNN applications.

In this article, we find and address the challenges of applying DCNv3 to lightweight CNNs. Firstly, through experiments, we find that DCNv3 does not fully utilize its advantages when applied to lightweight CNNs, and the reason is sparse sampling. To overcome this sparse sampling issue, we attempt to use DCNv3 with large kernel to compensate. Yet this measure leads to some problems, such as high memory consumption, many parameters and slow training speed. Therefore, we propose a deformable sampling core operation DSCN and a feature extraction module DSA that simultaneously address the poor performance of DCNv3 with large kernel on lightweight CNNs, achieve better computational speed and fewer memory consumption. Specifically, DSCN is a simplified version of DCNv3 core operation, which retains the deformation sampling characteristics of DCNv3 while minimizing the compu-

Zewen Yu, Xiaoqin Zhang and Li Zhao are with the Key Laboratory of Intelligent Informatics for Safety and Emergency of Zhejiang Province, Wenzhou University, Wenzhou 325035, China (e-mail: yzwsd1999@gmail.com; zhangxiaoqinnan@gmail.com; lizhao@wzu.edu.cn).

Guobao Xiao is with the College of Computer and Control Engineering, Minjiang University, Fuzhou 350108, China (e-mail: x-gb@163.com).

Xiaoqin Zhang is the corresponding author.

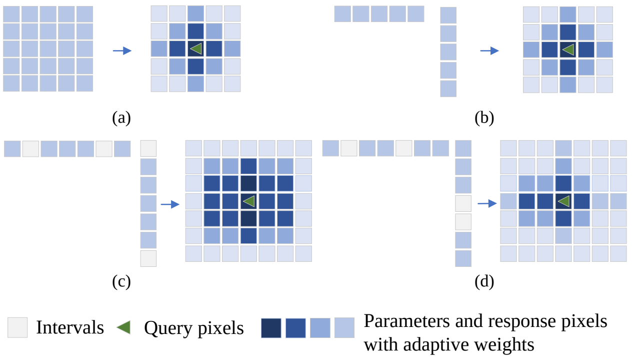


Fig. 1. Illustration of regularly shaped receptive fields of the vanilla convolution and irregularly shaped receptive fields of our proposed DSA. The enhancement of shape adaptability can be reflected by the shape of the receptive field. (a) and (b) show the receptive fields of a vanilla convolutional kernel and the spatial attention implemented by two strip convolutional kernels in MSCA [5], respectively. (c) and (d) show the receptive fields caused by different sampling distributions of DSA, respectively.

tational load and parameters. DSCN replaces bilinear interpolation with linear interpolation and drops the modulation mask. Furthermore, the deformable sampling is restricted to a single axis to avoid the quadratic growth in parameters and computational load that results from increasing the kernel size. After that, we observe the similarity between modulation mask branch in DCNv3 and spatial attention. Based on this similarity, we design Deformable Spatial Attention (DSA) by replacing the modulation mask branch with spatial attention multiplication to reduce the weight of DCNv3. As a replacement of DCNv3 with large kernel, DSA use a pair of DSCN operation along the x and y axes to implement deformable sampling in the entire spatial domain. DSA learns the irregular sampling distribution during training, which reflects on various receptive field shapes, as shown in Fig. 1. After deconstructing and redesigning DCNv3, we obtain a feature extraction module DSA that maintains the deformable sampling while avoiding sparse sampling. To verify the effectiveness of our design, we further propose a lightweight CNN backbone named DSAN based on DSA according to the principles of large kernel convolutional network design [2]. To examine the performance of DSAN, we evaluate it on four kind benchmark datasets. Our experiments and analyses demonstrate the superiority of DSAN in semantic segmentation, with DSAN-S achieving higher mIoU than InternImage-T based on DCNv3 while having fewer parameters and memory consumption.

Our main contributions can be summarized as follows:

1) We propose Deformable Strip Convollution (DSCN) by simplifying DCNv3, which is more suitable for lightweight CNN than the core operation of DCNv3. Compared to core operation of DCNv3, DSCN has has fewer parameters and computational load, which reflects from two perspective. First, DSCN does not require a modulation mask, reducing the amount of computation and parameters in this part. Second, DSCN is stripped and has linear interpolation, avoiding the computational complexity increases quadratically and reducing the computational load of single pixel deformation sampling.
2) We propose a new attention module named Deformable Spatial Attention (DSA) to replace DCNv3. Since the core part of the deformable sampling in DCNv3 is the offset branch rather than the modulation mask branch, we want to find an alternative to the mask branch. First, we analyze and find the similarity between the mask branch and spatial attention. Based on this, we use a pair of DSCN operations along x and y axes as the deformable sampling unit and spatial attention to form a visual feature extraction module DSA.
3) To verify DSCN and DSA, we propose a new lightweight CNN backbone called the Deformable Spatial Attention Network (DSAN). DSAN demonstrates performance at the intermediate to high levels across various vision tasks, including image classification, semantic segmentation and object detection. Especially in dense prediction task semantic segmentation, DSAN-S with a lightweight decoder outperforms InternImage-T with a heavyweight decoder, achieving 48.8%(+0.7%) mIoU on the ADE20K validation set, with only 35.0% and 9.4% of its parameters and computation. On the other vision tasks, DSAN-S achieves a top-1 accuracy of 82.3% on ImageNet1K [13], 81.4% mIoU on Cityscapes [14], and 46.1% mAP on COCO [15]. Compared to existing deformable CNNs, our model strikes a balance between performance and efficiency, reducing the hardware requirements for its deployment.

The remainder of this article is structured as follows. In Section II, we review related work including vision fundation models, attention mechanism and deformable convolution. We mainly illustrate the details about the proposed DSCN, DSA and our lightweight CNN backbone, DSAN, in Section III. In Section IV, we demonstrate ablation studies and extensive experiments about our proposed method. We summarize our paper in Section V.

II. RELATED WORK

A. *Vision Fundation Models*

For most computer vision tasks, deep neural networks used for these tasks always adopt an encoder-decoder architecture. The encoder part is a vision fundation model, also named backbone, undertakening the main workload of extracting features from input images. The encoders of this architecture have the capability to be categorized into CNN-based models [**?**], [5], [8], [16] and ViT-based models [6], [17], [18], based on the mechanism they employ. In the early works, CNN-based models are the mainstream for vision tasks on large scale datasets. Then, ViT-based models appear and achieve great reasults on many vision tasks, benefiting from their global receptive field. Recently, the improved CNN operations have also been combined with the ViT structure to construct a backbone, such as InternImage [8]. In this article, our work focuses on modifying the attention module and builds lightweight CNN backbones.

B. Attention Mechanism

Attention mechanism is inspired by the perceptual behavior of humans and has been applied to computer vision, such as channel attention, spatial attention and self-attention mechanism. Early works that introduce attention mechanism to improve CNNs include SENet [19], CBAM [20] and Non-local [21]. SENet implements channel attention by utilzing global pooling operations and linear mapping to enable model to focus on the channel domain. Based on channels attention, CBAM adds spatial attention to enable CNNs to extract features on both channel and spatial domains. Non-local introduces self-attention into CNNs, enabling them to extract global features while potentially slowing down the efficiency of inference. In the context of image encoders, Large Kernel Attention (LKA) [4] combines dilation convolution and separable convolution to implement channel and spatial attention at the same time. Based on LKA, Multi-scale Convolutional Attention (MSCA) [5] utilizes the multi-scale strip convolutional kernels to compensate for the shortcomings of LKA, which lack feature extraction capabilities for multi-scale objects. In essence, LKA and MSCA also contain spatial attention. Spatial attention has also been widely validated in numerous visual tasks, such as SCTFA [22], FSAD-Net [23], GCA [24], and so on. Our work is the closest to STDAN [25]. Different from it, which directly combines DCN and spatial attention, we find the similarity between the modulation mask branch of the DCNv3 and spatial attention, and use this as the basis for replacing the modulation mask with spatial attention when constructing our deformable sampling module DSA to reduce parameter and memory consumption.

C. Deformable Convolution

To overcome the limitations of vanilla convolution, there are some methods that work to improve its shape adaptability. Among the existing methods, The most widely applicable is the DCN series. DCNv1 [1] changes the regular sampling in vanilla CNN operation based on the learned offset, or modifies the regular receptive field. The work uses additional branch to attain a adaptive sampling position offset covering the entire spatial domain for each weight in sliding window, resulting in DCN kernels have a global and irregular receptive field. Based on DCNv1, DCNv2 [7] enhance adaptive sampling of the DCN kernels by using modulation masks. The improvement strategy allows DCNv2 kernels to control sampling over a broader range of feature levels and achieve better results than DCNv1. To apply DCNv2 to large-scale CNN-based foundation models, DCNv3 [8] separates DCNv2 weights into depth-wise and point-wise parts, introduces the multi-group mechanism, and applies new modulation mask normalization. These advancements result in stronger sparse sampling ability and more stable training process. Though shared weights in DCN help to reduce the number of parameters, DCNv3 still keeps the offset and mask produced by sibling branches as the source of the additional parameters. The DCN series are extensively tested in a wide range of applications. For example, Xu et al. [26] use DCNv1 to construct a network for multiview face image synthesis, making their model more suitable for situations with large pose variations. Zhu et al. [27] apply DCNv2 to construct the core module PDA of their model DVSRNet and use it to eliminate the motion error targeting high super-resolution quality, achieving better performance than state-of-the-art methods.

In this article, we pay attention to the optimization of the DCN core operation. Our proposed DSCN greatly reduces the computational load while retaining the deformable sampling ability of DCNv3, which improve the usability of DCNv3 on lightweight CNNs.

III. PROPOSED METHOD

In this section, we first discuss the characteristics of DCNv3 and the reasons that it fails to exhibit its inherent advantages when applied to lightweight CNNs. We then introduce the simplified DCNv3 operation Deformable Strip Convolution (DSCN), the visual extraction module Deforamble Spatial Attention (DSA), and the lightweight CNN backbone Deformable Spatial Attention Network (DSAN) in detail.

A. Preliminaries

Parameters and memory consumption of DCNv3. On the one hand, in design of DCNv3 , despite the efforts made to detach weights into depth-wise and point-wise parts and introduce multi-group machanism to reduce the parameters of DCN, the offset and modulation mask still account for the additional parameters and memory consumption. On the other hand, bilinear interpolation used in DCNv3 operation has a high computational load. The core operation of DCNv3 restricted to two axes results in a quadratic increase in the number of deforamble sampling operations with the kernel size, which greatly increases the computational load.

Core operation and deformable sampling. There are two side branches in DCNv3: the offset branch and the mask branch. The mask branch is mainly responsible for weighting in the spatial domain. The offset branch is responsible for learning sampling offsets and shares the ability to improve shape adaptability with the core operation. The core operation of DCNv3 is formulated as Eqn. 1 [8].

$$\mathbf{y}(p_0) = \sum_{g=1}^{G} \sum_{k=1}^{K} \mathbf{w}_g m_{gk} \mathbf{x}_g (p_0 + p_k + \Delta p_{gk}) \tag{1}$$

Sparse sampling. We directly use the DCNv3 (3×3) as a feature extraction module to construct a lightweight CNN based on ViT structure and train it on ImageNet1K. On samples with poor recognition performance, we conduct visual analysis and find that on its shallow layers, the offset

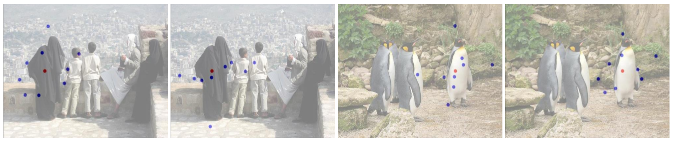


Fig. 2. Visualization of sparse sampling. DCNv3 determines whether to activate the central red sampling point based on the offset blue sampling point.

sampling of DCNv3 occurs very far from the center target point, with relatively few sampling points within the sampling range, i.e. sparse sampling, as shown in the Fig. 2. This sparse sampling occurring in shallow layers can be compensated for by the multiple groups in DCNv3 in heavyweight CNNs, but in lightweight CNNs with small parameter sizes and shallow layers, it increases noise and interfere with the later layers' judgment of the target.

B. Deformable Strip Convolution

To make DCNv3 with large kernel more suitable for lightweight CNNs, we design Deformable Strip Convolution (DSCN). We mainly use two measures to simplify the core operation of DCNv3. One measure is to directly constrain irregular sampling along the x or y axis and replace bilinear interpolation with linear interpolation. The other measure is to remove the mask weights in the core operation. We reveal the similarity between the modulation mask branch and use spatial attention to replace the modulation mask branch in the next subsection.

Firstly, a sampling process based on learned offset has the capability to be decomposed into separate sampling along the x and y axes. The analysis presented below provides the justification for this idea. In Eqn. 1, Δp_k represents the coordinate offset of the k-th sample point in the convolution kernel on the spatial domain. Here, k is equivalent to (i, j) when expressed in spatial coordinates. In Eqn. 1, $\mathbf{x}_g$ and $\mathbf{w}_g$ represent logical divisions of the tensor instead of actual splitting actions. And $\sum_{g=1}^{G}$ indicates the individual sampling based on learned offsets of per deformable group, where same group shares one offset tensor. Considering two strip DCNv3 operations along the x and y axes, and they are formulated as Eqn. 2a and Eqn. 2b, respectively.

$$\mathbf{y}^1(i_0, j_0) = \sum_{g=1}^{G} \sum_{j=0}^{K_w-1} \mathbf{w}^1 m_{0,j}^1 \mathbf{x}(i_0, j_0 + j + \Delta p_j^g) \tag{2a}$$

$$\mathbf{y}^2(i_0, j_0) = \sum_{g=1}^{G} \sum_{i=0}^{K_h-1} \mathbf{w}^2 m_{i,0}^2 \mathbf{y}^1(i_0 + i + \Delta p_i^g, j_0) \tag{2b}$$

The two operations successively perform irregular sampling along the x and y axes. The K_w and K_h represent the sampling size of a DCNv3 operation on the x and y axis, respectively. Substituting Eqn. 2a for Eqn. 2b, we get the final output superpixel after separate sampling along the x and y axes (Eqn. 3).

$$\mathbf{y}^2(i_0, j_0) = \sum_{g=1}^{G} \sum_{i=0}^{K_h-1} \sum_{j=0}^{K_w-1} \mathbf{w}' m_{i,j}' \mathbf{x}(i_0{+}i{+}\Delta p_i^g, j_0{+}j{+}\Delta p_j^g) \tag{3}$$

Due to $\mathbf{w}^1, \mathbf{w}^2 \in \mathbb{R}^{C \times C}$ and $m_{0,j}^1, m_{i,0}^2 \in \mathbb{R}$, the product of them has the capability to be rewritten into $\mathbf{w}' m_{i,j}' = \mathbf{w}^2 \mathbf{w}^1 m_{i,0}^2 m_{0,j}^1$, which is similar to Eqn. 1. In essence, the offset sampling that provides deformation capability in DCNv3 is to integrate the four pixel points around the offset position using bilinear interpolation according to their distance. Two successive bilinear interpolations along x and y axes also achieve this. After the analysis and reasoning, we demonstrate that DCNv3 sampling in the spatial domain has the capability to be replaced by separate DCNv3 sampling along the x and y axes.

DSCN. When offset sampling is restricted to a single axis, there is no difference between bilinear sampling and linear sampling in the addition and integration of pixels, but in practical programming, it reduces the number of zero-value operations. Hence, we use linear interpolation in the design of DSCN. Then, according to the similarity between modulation mask and spatial attention, it is not necessary to contain the mask. After all simplification measures, the design of DSCN units along the x and y axes are formulated as Eqn. 4a and Eqn. 4b.

$$\mathbf{y}^1(i_0, j_0) = \sum_{g=1}^{G} \sum_{j=0}^{K_w-1} \mathbf{w}\mathbf{x}(i_0, j_0 + j + \Delta p_j^g) \tag{4a}$$

$$\mathbf{y}^2(i_0, j_0) = \sum_{g=1}^{G} \sum_{i=0}^{K_h-1} \mathbf{y}^1(i_0 + i + \Delta p_i^g, j_0) \tag{4b}$$

Consistent with the symbol meanings in Eqn. 1, G represents the number of deformable groups. Within the g-th deformable group, $\mathbf{w} \in \mathbb{R}^{C \times C}$ and $\mathbf{x} \in \mathbb{R}^{C \times H \times W}$ denote the weights of linear mappings and their corresponding inputs.

For the computational complexity, the parameters and computational load of DSCN increase linearly with kernel size, that $O(n)$ instead of $O(n^2)$. An offset superpixel is obtained by two linear interpolation operations, which reduces the computational load to 63.2% on theory compared to a bilinear interpolation operation. The proof is as follows.

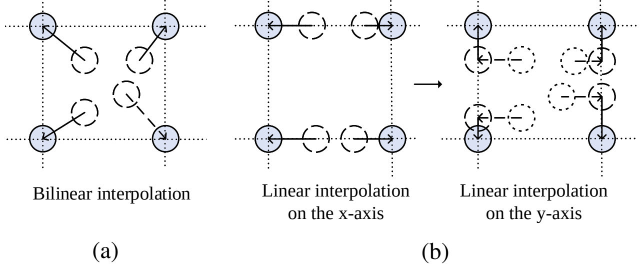

Bilinear interpolation
Linear interpolation on the x-axis
Linear interpolation on the y-axis
(a)
(b)

Fig. 3. The difference between bilinear interpolation and linear interpolation. (a) shows a deformable sampling for a grid with four pixels used by bilinear interpolation. (b) shows that two successive linear interpolation operations on the grid.

Suppose three pixels on spatial domain, and their coordinations are (x_0, y_0),(x_1, y_1) and (x_2, y_2). For bilinear interpolation, the pixel values $f(x_1, y_1)$ and $f(x_2, y_2)$ are known, $f(x_0, y_0)$ is obtained by a bilinear interpolation operation. The bilinear interpolation operation is formulated as Eqn. 5.

$$\begin{aligned} f(x_0, y_0) =& f(x_1, y_1)(x_2 - x_0)(y_2 - y_0) \\ &+ f(x_1, y_2)(x_0 - x_1)(y_2 - y_0) \\ &+ f(x_1, y_2)(x_2 - x_0)(y_0 - y_1) \\ &+ f(x_2, y_2)(x_0 - x_1)(y_0 - y_1) \end{aligned} \tag{5}$$

In linear interpolation, the $f'(x_0, y_1)$ and $f(x_0, y_0)$ are obtained through two successive interpolation operations, respectively. These two successive linear interpolation operations are formulated as Eqns. 6.

$$f'(x_0, y_1) = f(x_1, y_1)(x_0 - x_1) + f(x_2, y_1)(x_2 - x_1) \tag{6a}$$

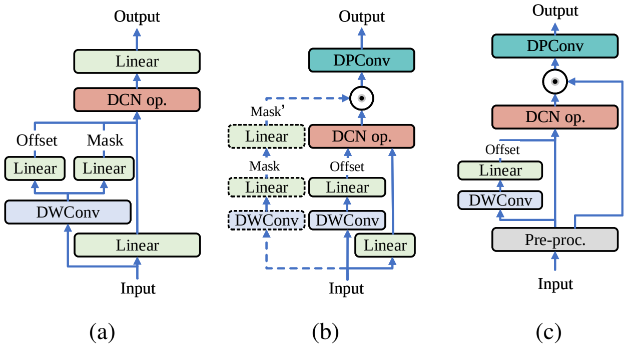
$$f(x_0, y_0) = f'(x_0, y_1)(y_0 - y_1) + f'(x_1, y_2)(y_2 - y_1) \tag{6b}$$

If the pixel at the coordinates (x_1, y_2) is not obtained by offset sampling, then $f'(x_1, y_2) = f(x_1, y_2)$. Resume that there is a grid with four pixels and all pixels are offset sampled, as shown in Fig. 3. After counting the floating-point operations (FLOPs), these are 76 FLOPs for a bilinear interpolation operation while two successive linear interpolation operations require only 48 FLOPs, which is 63.2% of the original amount required for bilinear interpolation. Therefore, we use DSCN to replace a DCNv3 with large kernel, achieving the goal of lightweighting.

C. Deformable Spatial Attention

In this subsection, we first discuss the similarity between modulation mask branch in DCNv3 and spatial attention. Then, we introduce Deformable Spatial Attention (DSA).

Similarity between modulation mask branch in DCNv3 and spatial attention. In the design of DCNv3, there are six parts of it: input linear projection, depth-wise convolution, offset linear projection, mask linear projection, DCNv3 core operation and output linear projection, as shown in Fig. 4a. We believe the three flow charts shown in Fig. 4 have equivalent functions, even though they appear different. The processing flow shown in Fig. 4c has the capability to be derived step by step from the one shown in Fig. 4a. The basis and analysis are as follows.



(a) (b) (c)

Fig. 4. DCNv3 and its evolution. We try to use these flow charts to reveal the similarity between modualtion mask and spatial attention. (a) shows the processing flow of a DCNv3 unit. (b) and (c) show processing units with an equivalent processing flow of a DCNv3 unit. “DCN op.” represents the core operation of DCNv3. Linear operation acts on channel domain. The terms “DWConv” and “DPConv” represent the depth-wise convolution and its combination with point-wise convolution, respectively. The symbol $\odot$ represents the element-wise multiplication operation.

In Eqn. 3, $\mathbf{w}' \in \mathbb{R}^{C\times C}$, and $\mathbf{x} \in \mathbb{R}^{C\times H\times W}$ represent the group weight and the input features, respectively. The modulation mask is represented by $M \in \mathbb{R}^{(K_h\times K_w)\times H\times W}$, and $m'_{i,j}$ represents the value at spatial coordinate (i_0, j_0) as $M(i \times K_w + j, i_0, j_0)$. From Eqn. 3, we observe the core operation of DCNv3, which involves sampling based on the learned offset, a spatial coordination $(\Delta p_i^g, \Delta p_j^g)$. In Fig. 4a, the depth-wise convolution (DWConv) in DCNv3 is shared by the producing both the offset and modulation mask producing branches, which is used to reduce the number of parameters. Therefore, a new DWConv is added to detach modulation mask branch out from offset branch. Then, Eqn. 3 implies that the nature of DCNv3 core operation is collecting superpixels in the spatial domain according to learned offset and multiply it with modulation mask value at the same time. Hence, Eqn. 3 can be reformulated as Eqn. 7.

$$\mathbf{y}(i_0, j_0) = \mathbf{w}O_1 * (O_2 M'(i_0, j_0) \odot \mathbf{x}_u) \tag{7}$$

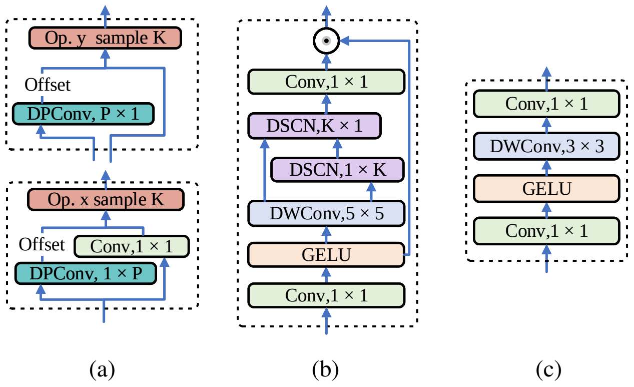
In Eqn. 7, $O_1 \in \mathbb{R}^{C\times K_h\times K_w}$ and $O_2 \in \mathbb{R}^{C\times 1}$ denote tensors filled with ones, respectively. The term $M'(i_0, j_0) \in \mathbb{R}^{1\times(K_h\times K_w)}$ denotes the reshaped version $M(i_0, j_0)$ with an additional dimension while the term $\mathbf{x}_u \in \mathbb{R}^{C\times K_h\times K_w}$ denotes the corresponding element tensor of sampling grid collected by DCNv3 core operation within the spatial domain. Symbols $*$ and $\odot$ represent the depth-wise convolution operation and element-wise multiplication operation, respectively.

Then, a linear operation $W' \in \mathbb{R}^{(K_h\times K_w)\times H\times W}$ is added to map sliding windows to the spatial domain and Eqn. 7 is applied to full input $\mathbf{x}$, resulting in Eqn. 8. The reason for such replacement is that the weights of modulation mask are not spatially repeated, different from the repeated weights of vanilla convolution.

$$\mathbf{y} = \mathbf{w}O_1 * (O_2 M' W' \odot \mathbf{x}) \tag{8}$$

In this equation, two terms are identified as representing specific operations. The terms $\mathbf{w}O_1 * ()$, O_2 and W' are equivalent to a combination of depth-wise convolution and point-wise convolution operation, and two linear operations, respectively. After these equivalent transformation, the processing unit shown in Fig. 4b is formed. Ultimately, upon fully integrating the modulation mask branch into into the prefixed processing module, a modualtion mask branch in DCNv3 unit is transformed into a branch similar to spatial attention.

DSA. Inspired by the similarity between DCNv3 and spatial attention, we propose to substitute spatial attention instead of the modulation mask branch, and combining it with DSCN to design DSA. DSA comprises a pair of DSCN operations along x and y axes, two 1×1 convolution kernels, a 5×5 depth separable convolution kernel, a GELU activation function, and spatial attention element-wise multiplication, as shown in Fig. 5b. Among these components, the pair of DSCN operations along x and y axes is responsible for for conducting deformable sampling within the spatial domain. The other CNN kernels assist DSCN in feature extraction. After information extraction by the first single-axis DSCN, the spatial information along the other axis temporarily changes, which results in an inaccurate offset tensor being learned by the second DSCN. Therefore, the feature tensor should not simply pass through two consecutive DSCN operations. To avoid this, we add a pathway between the second DSCN and the 5×5 depth separable convolution, enabling the offset branch of this DSCN to receive feature tensors that have not been processed by the first DSCN, as depicted in Fig.5a and Fig.5b. The pair of DSCN operations along the x and y axes work together to achieve irregular sampling in the spatial domain, inheriting



(a) (b) (c)

Fig. 5. The components and processing flow of DSCN, DSA and FFN. (a) illustrates that two DSCN core operations (Op.) along the x and y axes, respectively. (b) presents DSA, which is composed of a pair DSCN operations, two linear operations acting on channel domain and a GELU activation function. (c) shows FFN, consisting of two linear operations, a combination of depth-wise convolution and point-wise convolution, and a GELU activation function.

the characteristic of globally irregular sampling from DCNv3 while being lightweight.

Additionally, aking into account both the FFN module in spatial-attention-based CNNs and the principles of designing large kernel networks proposed in [2], the FFN module is designed with a sequence of operations, shown in Fig. 5c. This sequence consists of two linear operations acting on channel domain, a combination of depth-wise and point-wise convolution, and a GELU activation function.

D. DSAN

In this section, we introduce how the lightweight CNN backbone DSAN is constructed. There are two types of blocks used to build DSAN. The first one is named the embedding block, which is utilized to adjust the spatial size and channels of input tensors. The second type is the basic block, responsible for extracting visual features from input tensors. We construct it based on the basic block structure of ViT [17], incorporating DSA, FFN, and batch normalization. Finally, these embedding blocks and basic blocks are stacked together to form the lightweight CNN backbone DSAN.

DSAN has a classical structure with a sequence of four stages. In this structure, the output spatial resolution decreases, i.e $\frac{H}{4} \times \frac{W}{4}$, $\frac{H}{8} \times \frac{W}{8}$, $\frac{H}{16} \times \frac{W}{16}$ and $\frac{H}{32} \times \frac{W}{32}$, while the number of the channels increases, which implemented by embedding blocks, as shown in Fig. 6.

The input images are first processed by the embedding block, which spatially splits the images into overlapping patches. Within each stage, the layers maintain a consistent composition, with the same input and output sizes as well as the number of channels. To thoroughly explore the performance potential of DSAN, we construct DSAN-T and DSAN-S using two sets of hyperparameters. The details for both DSAN-T and DSAN-S, including the depth of stages, the number of channels, the output spatial size of the stages, the sampling size of DSCN operations, and the total number of parameters (excluding the linear head), are shown in Tab. I.

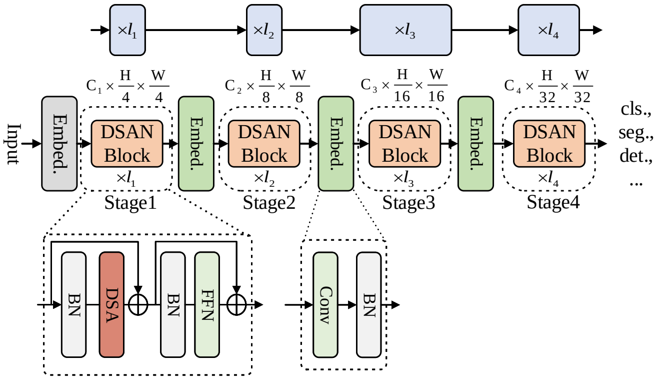


Fig. 6. Overall structure of DSAN. DSAN has a sequence of four hierarchical stages. Each stage consists of a stack of basic blocks, where the core visual extraction module is DSA. Each basic block is stacked by DSA, FFN and batch normalization, utilizing the residual connections throughout. Additionally, a embedding block consists of a vanilla convolution and a batch normalization.

TABLE I
THE HYPERPARAMETERS FOR DSAN OF DIFFERENT SCALES

Model & #Params	Stage1	Stage2	Stage3	Stage4
Spatial size	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{32} \times \frac{W}{32}$
DSAN-T 4.3 M	C = 32 $l_1 = 3$ $K = 11$ $P = 11$ $G = 1$	C = 64 $l_2 = 3$ $K = 11$ $P = 11$ $G = 4$	C = 160 $l_3 = 5$ $K = 7$ $P = 7$ $G = 8$	C = 256 $l_4 = 2$ $K = 5$ $P = 5$ $G = 8$
DSAN-S 19.4 M	C = 64 $l_1 = 2$ $K = 15$ $P = 9$ $G = 4$	C = 128 $l_2 = 2$ $K = 13$ $P = 7$ $G = 8$	C = 320 $l_3 = 5$ $K = 7$ $P = 5$ $G = 16$	C = 512 $l_4 = 3$ $K = 5$ $P = 5$ $G = 16$

From our perspective, the reason why DCNv3 (3 × 3) sampling operation loses its advantage in lightweight CNNs is due to its limited sampling coverage across the entire spatial domain, leading to a dearth of sampling information for effective perception and reasoning. Enhancing the sampling capability would typically involve increasing the sampling size or stacking more layers. However, both methods lead to an increase in the number of parameters, thereby negating the ‘lightweight’ attribute of the model. Due to the compact computational requirements and reduced parameter count in DSCN and DSA, DSAN is still considered lightweight even with the application of large kernel sampling.

IV. EXPERIMENTS

In this section, we carry out a series of experiments to validate our lightweight CNN backbone DSAN. These experiments include an ablation study, image classification, semantic segmentation, and object detection. The ablation study is specifically designed to demonstrate the effectiveness of the key components within DSA. Meanwhile, the other experiments are aimed at verifying the general performance and adaptability of DSAN across various computer vision applications.

A. Datasets and Implementation Details

Datasets. Our method is evaluated on multiple datasets to validate its performance. These datasets include ImageNet1K [13], ADE20K [28], Cityscapes [14] and COCO [15]. ImageNet1K is a vast image classification dataset with 1,000 categories. It consists of 1.28M images for training and 50K images for validation. ADE20K is a popular dataset specifically designed for semantic segmentation task. It includes 150 semantic categories and consists of 20K, 2K, and 3K images for training, validation, and testing, respectively. Cityscapes is a dataset for semantic segmentation in the context of autonomous driving. It consists of 19 categories and utilizes 2,975, 500, and 1,525 images for training, validation, and testing, respectively. Lastly, COCO is a large-scale dataset for many computer vision tasks. For object detection, it has 80 categories, including 118K and 5K images for training and validation, respectively.

Implementation Details. To guarantee the highest possible performance, we implement a two-stage training strategy. Initially, we pre-train DSAN on ImageNet1K. Following this, we proceed to fine-tune the pretrained DSAN on the datasets of specific downstream tasks, including semantic segmentation and object detection. The experimental implementation is carried out using PyTorch [29], timm [30], mmsegmentation [31] and mmdetection [32]. All models are trained on a node equipped with eight nvidia RTX 3080Ti GPUs and another node equiped with four nvidia RTX 3090 GPUs. The specific training configurations for the different tasks are detailed as follows.

Image classification. The dataset is ImageNet1K. The training settings are followed in [4], [5], [33], with a total of 310 epochs. For optimization, we use the AdamW [34] , setting its momentum to 0.9 and weight decay at 5×10^{-2}. The maximum learning rate is configured as 1×10^{-3}, and a warm-up strategy coupled with a cosine scheduler [35] is employed to adjust the learning rate throughout the training process. For the training images, they are resized to a dimension of 224×224. For both DSAN-T and DSAN-S, we set the batch size to 512×512, which optimizes the balance between computational efficiency and effective training performance. At this stage, we use mixup, random clipping, random flipping and cutmix to augment the training data. Furthermore, we incorporate Layerscale [36] and Droppath [37] methodologies in the training process. For performance evaluation, we utilize the top-1 accuracy metric.

Semantic segmentation. The datasets are Cityscapes and ADE20K. This experiment mainly refers to [5]. The optimizer employed is AdamW, with a momentum of 0.9 and weight decay set at 5×10^{-2}. The maximum learning rate is set to 6×10^{-5}, the minimum learning rate is set to 1×10^{-6}. The training process utilizes 180K iterations for ADE20K. Similarly, for Cityscapes, it uses 80K iterations. he input sizes for ADE20K and Cityscapes are configured to be 512×512 and $512 \times 1,024$, respectively. For the evaluation of semantic segmentation performance, we adopt the mean Intersection over Union (mIoU) as our primary metric.

Object detection. The dataset is COCO. This experiment mainly refers to [8]. We employ two popular object detection frameworks, Mask R-CNN [38] and RetinaNet [39], to validate our backbones. The latter is utilized solely for the ablation study purposes. In this study, we set the batch size is 32 and configure the input image size as $1,024 \times 800$. For Mask R-CNN, the training settings are followed in [8]. The input sizes of images are $1,333 \times 800$, and the total number of training epochs is set at 12. The batch size is adjusted to 16. AdamW serves as the chosen optimizer, with the learning rate established at 1×10^{-4}. We employ the mean Average Precision (mAP) as our primary metric for evaluation.

B. Ablation Study on Attention Module

This subsection is divided into two parts. The first part presents an ablation study focusing on the key components within DSA, specifically the dual DSCN operations along the x and y axes, as well as the attention multiplication mechanism. We begin by pre-training DSAN-T that lack one of these key components and then proceed to fine-tune them on the semantic segmentation dataset ADE20K and object detection dataset COCO. he outcomes of this experiment are tabulated in Tab. III.

TABLE III
ABLATION STUDY ON THE KEY COMPONENTS OF DSA

Op.(x)	Op.(y)	Atten.	Acc (%)	mIoU (%)	mAP (%)
×	✓	✓	75.2	42.2	39.5
✓	×	✓	75.4	42.8	39.6
✓	✓	×	75.8	43.0	40.1
✓	✓	✓	76.4	43.5	40.7

The pair of DSCN operations along the x and y axes. In the table, "Op.(x)" and "Op.(y)" denote the DSCN operations along the x and y axes or DSCN $1 \times K$ and $K \times 1$, respectively. The pair of DSCN operations enables DSA module to have a globally deformable receptive field, improving its adaptability to the irregular shape of objects and the ability to capture

TABLE II
ABLATION STUDY ON THE FEATURE EXTRACTING MODULE

Module	#Params.	Classification			Segmetation			Detection			Inference	
		Acc	Mem.	Speed	mIoU	Mem.	Speed	mAP	Mem.	Speed	Mem.	Speed
DSA w/ DSCN	4.3M	76.4	78.4	1993	43.5	68.0	166	40.7	68.8	74	2.3	62
DSA w/ strip DCNv3	4.7M	76.4	84.8	1606	42.5	68.8	158	37.8	69.7	68	2.6	50
DSA w/ strip conv.	3.9M	74.9	83.6	2930	41.8	67.2	187	39.3	57.5	89	1.8	95
DCNv3 (3×3)	3.8M	74.3	75.2	2760	42.1	67.2	158	39.9	69.7	68	1.8	81
DCNv3 (large kernel)	5.4M	/	OOM	/	/	/	/	/	/	/	2.6	30

a long range of pixel relationships. After removing DSCN operations along the x and y axes, the accuracy of the model on the validation set of ImageNet1K drops by 1.2% and 1.0%, respectively. And the mIoU on the validation set of ADE20K drops by 1.3% and 0.7%, respectively. The mAP on the validation set of COCO drops by 1.2% and 1.1%, respectively.

Attention. Attention makes the model achieve adaptive property and replace the function of modulation masks. After removing it, the accuracy of the model on the ImageNet1K validation set decreases by 0.6%, the ADE20K validation set mIoU decreases by 0.5% and the COCO validation set mAP decreases by 0.6%.

The results of this abalation sutdy imply that DSCN operations and attention multiplication have a significant impact on DSA, which are proved to be the effective components.

Another part is the ablation study on the parts related to deformation sampling of DSA. We substitute the pair of DSCN operations with a pair of vanilla strip convolution operations and another pair of strip DCNv3 operations to demonstrate the advantage of the large deformable receptive field, as well as the effectiveness of our proposed improvements. All strip convolution operations have an identical kernel size for fair comparison. We also compare DSA against a DCNv3 unit with a smaller kernel size (3×3) to validate our hypothesis regarding the performance limitations caused by sparse sampling of DCNv3. Furthermore, we include performance metrics for DCNv3 with larger kernel, where the kernel size matches that of the strip convolution operations, in order to provide a comprehensive evaluation.

Based on the results from this ablation study, as presented in Tab. II, it is evident that DSCN reduces the parameter count and memory consumption of DCNv3 while maintaining or even surpassing its performance levels. In the table, the metrics for memory consumption (labeled ”Mem.”) and speed are measured in GB and frames per second (FPS), respectively. Specially, apart from DSCN, the other three convolutional kernel configurations do not show superiority across all three tasks. The pair of DSCN operations outperforms a pair of strip vanilla convolutions by achieving an improvement of 1.5% top-1 accuracy, 1.7% mIoU, and 1.4% mAP. Additionally, there is a distinction between DSA with strip DCNv3 and the one with DSCN in that the former does not incorporate attention multiplication. The comparison of the results from these experiments also reveals that spatial attention plays a similar role to modulation mask branch when employing DCNv3 with large kernel.

From the analysis of the training process, DSA accelerates the training process of DSAN-T. We use the FPS to evaluate the training computitional speed of DSAN-T on a node with eight 3080Ti GPUs. Specifically, DSCN operations leading to a 24.1% 5.1% and 8.8% training acceleration effect on image classification, semantic segmentation and object detection, respectively, compared to strip DCNv3. Our design is enable to be pretrained on the same equipment while DCNv3 with large kernel has a Out-Of-Memory (OOM) state. Furthermore, DSCN also exhibits a 7.5% decrease in training memory consumption when compared to strip DCNv3.

TABLE IV
THE PARAMETERS, FLOPS AND TOP-1 ACCURACY OF DIFFERENT METHODS ON IMAGENET1K VALIDATION SET

Method	#Params (M)	FLOPs (G)	Acc (%)
PVTv2-B0 [18]	3.4	0.6	70.5
MiT-B0 [40]	3.7	0.6	70.5
VAN-T [4]	4.1	0.9	75.4
MSCAN-T [5]	4.2	0.9	75.9
DSAN-T (ours)	**4.5**	**1.0**	**76.4**
ResNet18 [41]	11.7	1.8	69.8
PoolFormer-S12 [42]	11.9	2.0	77.2
PVT-Tiny [43]	13.2	1.9	75.1
VAN-S [4]	13.9	2.5	81.1
MiT-B1 [40]	14.0	2.1	78.7
MSCAN-S [5]	14.0	2.6	81.2
gMLP-S [44]	20.0	4.5	79.6
RegNetY-4G [45]	21.0	4.0	80.0
ResNeXt50-32x4d [16]	25.0	4.3	77.6
MiT-B2 [40]	25.4	4.0	81.6
VAN-B [4]	26.6	5.0	82.8
MSCAN-B [5]	26.8	5.1	83.0
Swin-T [46]	28.3	4.5	81.3
InternImage-T [8]	29.9	4.8	83.5
DSAN-S (ours)	**19.9**	**3.2**	**82.3**

TABLE V
THE PARAMETERS AND RESULTS OF DIFFERENT METHODS ON ADE20K VALIDATION SET

Method	#Params (M)	FLOPs (G)	mIoU (MS)
Segformer-B0 [40]	3.8	8.4	38.0
SegNeXt-T [5]	4.3	6.6	42.2
PVTv2-B0-FPN [18]	7.6	25.0	37.2
VAN-T-FPN [4]	8.0	25.8	38.5
DSAN-T-Ham (ours)	**4.6**	**6.8**	**43.5**
Segformer-B1 [40]	13.7	15.9	43.1
SegNeXt-S [5]	13.9	15.9	45.8
ResNet18-FPN [41]	15.4	31.1	32.9
PoolFormer-S12-FPN [42]	16.0	31.0	37.2
PVT-T-FPN [43]	17.0	33.2	35.7
VAN-S-FPN [4]	17.6	34.6	42.9
PoolFormer-S24-FPN [42]	23.0	39.0	40.3
SegNeXt-B [5]	27.6	34.9	49.9
PVT-S-FPN [43]	28.2	44.5	39.8
VAN-B-FPN [4]	30.3	47.7	46.7
InternImage-T-Uper [8]	59.1	236.1	48.1
DSAN-S-Ham (ours)	**20.7**	**23.0**	**48.8**

From the analysis of the inference, DSA with DSCN achieves an inference speed that is 2.1 times faster than DCNv3 with large kernel and 1.2 times faster than strip DCNv3. This performance comparison is based on processing a single RGB image of size $1,024 \times 1,024$ pixels on a single NVIDIA 3080Ti GPU. Additionally, the inference memory consumption for DSA with DSCN is also notably more efficient in this setup.

By examining the aforementioned experimental outcomes, it becomes evident that the utilization of DSA obtains advantages of both DCNv3 with large kernel and strip vanilla convolution. It reduces the parameters and memory resumption of DCNv3 with large kernel and accelerates the computation. These findings highlight the effectiveness of incorporating DSA as a visual extraction module to enhance the shape adaptability of lightweight CNNs.

C. Image Classification

The model exhibits impressive performance in computer vision task, which heavily relies on pretraining. According to the mainstream approach, we also choose image classification as a pre-training task for other downstream tasks.

The comparative performance of models on the ImageNet1K validation set is thoroughly outlined in Tab. IV, which includes an exhaustive comparison with a variety of model types. These include CNN-based models such as those proposed in [4], [5], [8], [41], ViT-based models such as [40], [42], [43], and MLP-based models represented by [44]. Specifically, when compared to VAN-T [4] and MSCAN-T [5], which also use spatial attention machinism and have fewer than 10M parameters, DSAN-T achieves an top-1 accuracy increases of 1.0% and 0.5%, respectively. Compared to InternImage-T [8], which employs DCNv3 for feature extraction, DSAN-S achieves a top-1 accuracy of 82.3% with a 1.2% performance gap, while reducing parameters and computation by 33.4% and 33.3%, respectively. According to the above experimental data, DSAN-S is able to outperform or closely approach models with parameter counts around 25M even when its own parameter count is below 20M, which attests to the effectiveness of our optimized design.

D. Semantic Segmentation

After classification pre-training, models of different scales are fine-tuned on semantic segmentation datasets of varying sizes and their performance is evaluated on these datasets. Compared to image classification, the performance in image segmentation tasks, which requires better pixel-level classification, more effectively demonstrates the shape adaptability of CNNs.

ADE20K. The performance of DSAN on the ADE20K validation set is shown in Tab. V, alongside the performances of CNNs [4], [5], [41] and ViT-based models [40], [43]. In the table, the floating-point operations (FLOPs) is calculated with an RGB image of size 512×512. n the category of models with parameters under 10M, DSAN-T achieves superior results, demonstrating a 1.2% improvement in mIoU compared to the semantic segmentation CNN SegNeXt-T, which boasts a substantial regular receptive field. This suggests that the large deformable receptive field empowers DSAN-T with enhanced shape adaptability, thus contributing to its improved performance on segmentation tasks. At a larger parameter scale, DSAN does not fully match the performance of SegNeXt-B, which is specifically designed for semantic segmentation. Nonetheless, when compared to VAN-B-FPN, DSAN-S-Ham still delivers superior results. Semantic segmentation is a dense prediction task, and we believe that it reflects the decrease caused by the sparsity of using DCNv3 with small kernel. Evidence for this idea is provided by DSAN-S with Ham achieves 0.7% mIoU higher performance than InternImage-T with UperNet, while its parameters and computation are only 35.0% and 9.4% of InternImage-T with UperNet. Compared to Hamburger, UpperNet should help the backbones to perform better, which is proved by the next experiment.

TABLE VII

THE PARAMETERS, FLOPS AND RESULTS OF DIFFERENT ARCHITECTURES ON ADE20K VALIDATION SET

Architecture	#Params (M)	FLOPs (G)	mIoU (MS)
DSAN-T w/ PSP [50]	15.5	7.7	41.4
DSAN-T w/ Uper [51]	32.1	214.8	45.1
DSAN-T w/ Ham [52]	4.6	6.8	43.5

Different decoders. Three decoder designs for semantic segmentation are based on a four-stage backbone, all employing multi-layer mappings to produce semantic masks. The first design concatenates features from all stages, exemplified by [40], [51], [53]. The second approach selectively concatenates and processes features of the last three stages using specialized decoder heads, prioritizing higher-level features over lower ones. Lastly, similar to works like [50], [54], [55], the third design uses decoder heads only for features of the final stage, simplifying the process but increasing dependence on late-stage output, which can limit its efficacy in lightweight CNNs.

In this experiment, we integrate DSAN with various decoders to validate its general applicability. The obtained experimental results are tabulated in Tab. VII. DSAN-T w/ PSP/Uper/Ham represents the fusion of DSAN-T with either PSPNet [50], UperNet [51], or Hamburger [52] architectures. Notably, UperNet and PSPNet fall under the first and third categories of decoder designs respectively, while Hamburger (Ham) is classified as a member of the second category.

TABLE VI

RESULTS OF MASK R-CNN WITH DIFFERENT BACKBONES ON COCO VALIDATION SET 2017

Backbone	#Params (M)	FLOPs (G)	mAP^b	mAP$^b_{50}$	mAP$^b_{75}$	mAP^m	mAP$^m_{50}$	mAP$^m_{75}$
PVTv2-B0 [47]	23.5	196.0	38.2	60.5	40.7	36.2	57.8	38.6
VAN-T [4]	23.9	187.1	40.2	62.6	44.4	37.6	59.6	40.4
DSAN-T (ours)	**24.3**	**188.6**	**42.6**	**64.3**	**46.4**	**38.9**	**61.5**	**41.6**
VAN-B1 [5]	33.5	221.5	42.6	64.2	46.7	38.9	61.2	41.7
PVTv2-B1 [47]	33.7	243.7	41.8	64.3	45.9	38.8	61.2	41.6
ResNet50 []	44.2	260.1	38.2	58.8	41.4	34.7	55.7	37.2
VAN-B2 [4]	46.2	272.8	46.4	67.8	51.0	41.8	65.2	44.9
Swin-T [48]	48.0	267.0	42.7	65.2	46.8	39.3	62.2	42.2
ConvNeXt-T [49]	48.1	262.1	44.2	66.6	48.3	40.1	63.3	42.8
InternImage-T [8]	48.9	269.8	47.2	69.0	52.1	42.5	66.1	45.8
DSAN-S (ours)	**39.5**	**235.4**	**46.1**	**67.8**	**50.5**	**41.5**	**64.7**	**44.6**

DSAN-T, when combined with PSPNet, achieves a mIoU of 41.4%, while its collaboration with UperNet results in a mIoU of 45.1%. Consequently, our design consistently performs well across different types of decoders as well. The fact that DSAN with UperNet outperforms DSAN with Hamburger, also demonstrates that UperNet, as a heavyweight semantic segmentation decoder, can effectively enhance the performance of the backbone.

Cityscapes. The performance of DSAN on the Cityscapes validation set is presented in Tab. VIII, where the FLOPs is computed based on an RGB image with dimensions of $2,048 \times 1,024$. This experiment was designed to contrast our approach with the self-attention mechanism, which also possesses a global receptive field. The modified convolutional neural network is able to attain comparable or identical performance while maintaining a compact model size. Specifically, DSAN-T achieves an equivalent 80.0% mIoU as SegFormer-B1, yet has only 33.6% of its parameters and 21.6% of its computational requirements. Although DSAN-S-Ham does not outperform SegFormer-B2, the performance disparity is marginal at just 0.7% mIoU, and notably, its computational load constitutes only 25.3% in comparison to that of SegFormer-B2.

TABLE VIII
PARAMETERS AND RESULTS OF DIFFERENT METHODS ON CITYSCAPES VALIDATION SET

Method	#Params (M)	FLOPs (G)	mIoU (MS)
SegFormer-B0 [40]	3.8	125.5	78.1
SegFormer-B1 [40]	13.7	243.7	80.0
DSAN-T-Ham (ours)	**4.6**	**52.6**	**80.0**
SegFormer-B2 [40]	27.5	717.1	82.2
DSAN-S-Ham (ours)	**20.7**	**181.1**	**81.5**

E. Object Detection

To validate the versatility of our design, we also implemented it as the encoder in Mask R-CNN for object detection tasks. The details and performance metrics of various models on the COCO 2017 validation set are presented in Tab. VI, with the FLOPs being computed using an RGB image of size $800 \times 1,280$. In this experiment, we employed the Mask R-CNN framework to evaluate the performance of DSAN. Remarkably, with a total model size not exceeding 30M parameters, DSAN-T achieves a 42.6% mAP, which is a noteworthy 2.4% mAP improvement over VAN-T that utilizes dilation convolution as its primary operations within the large kernel attention module. Although there is a certain gap between the performance of DSAN-S and InternImage-T, it still has better performance compared to VAN-B, ResNet50, and ConvNeXt-T, which are constructed with vanilla convolution with more than 40M parameters.

F. Visualization

The most advantage of DSA is the globally deformable receptive field. We demonstrate this through two visualization methods. The first method is Grad-CAM [56] based on gradient localization, which illustrates the better shape adaptability of DSA compared to conventional convolutional modules through the CAM visual results of the models in the ablation study on the ImageNet validation set. To confirm this, we select various categories of objects images with irregular shapes, including natural organisms, man-made objects, and natural landscapes. All CAM visual maps are produced by DSAN-T and the size of a image is 256×256. From the CAM maps, the more accurate location with the various shapes of DSA with DSCN proves its various deformable receptive fields, as shown in Fig. 7. The globally deforamble receptive field is especially shown in the second to fifth lines, which also confirms our judgment on the sparsity sampling of DCNv3, such as DCNv3 with small kernel size (3×3) do not sample all targets in images.

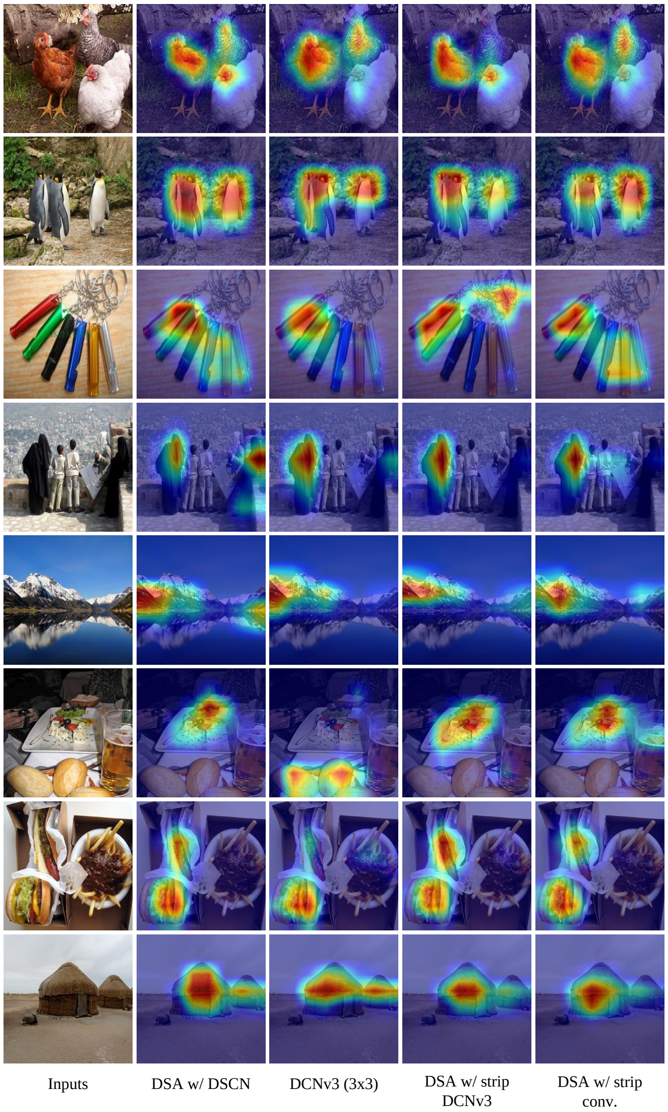


Fig. 7. CAM visual results of DCNv3, and DSA with different strip convolution, including strip vanilla convolution, strip DCNv3 and DSCN. The inputs come from ImageNet1K validation set, which produced by Grad-CAM.

The second method is to visualize the semantic segmentation masks, anchor boxes and instance segmentation masks.

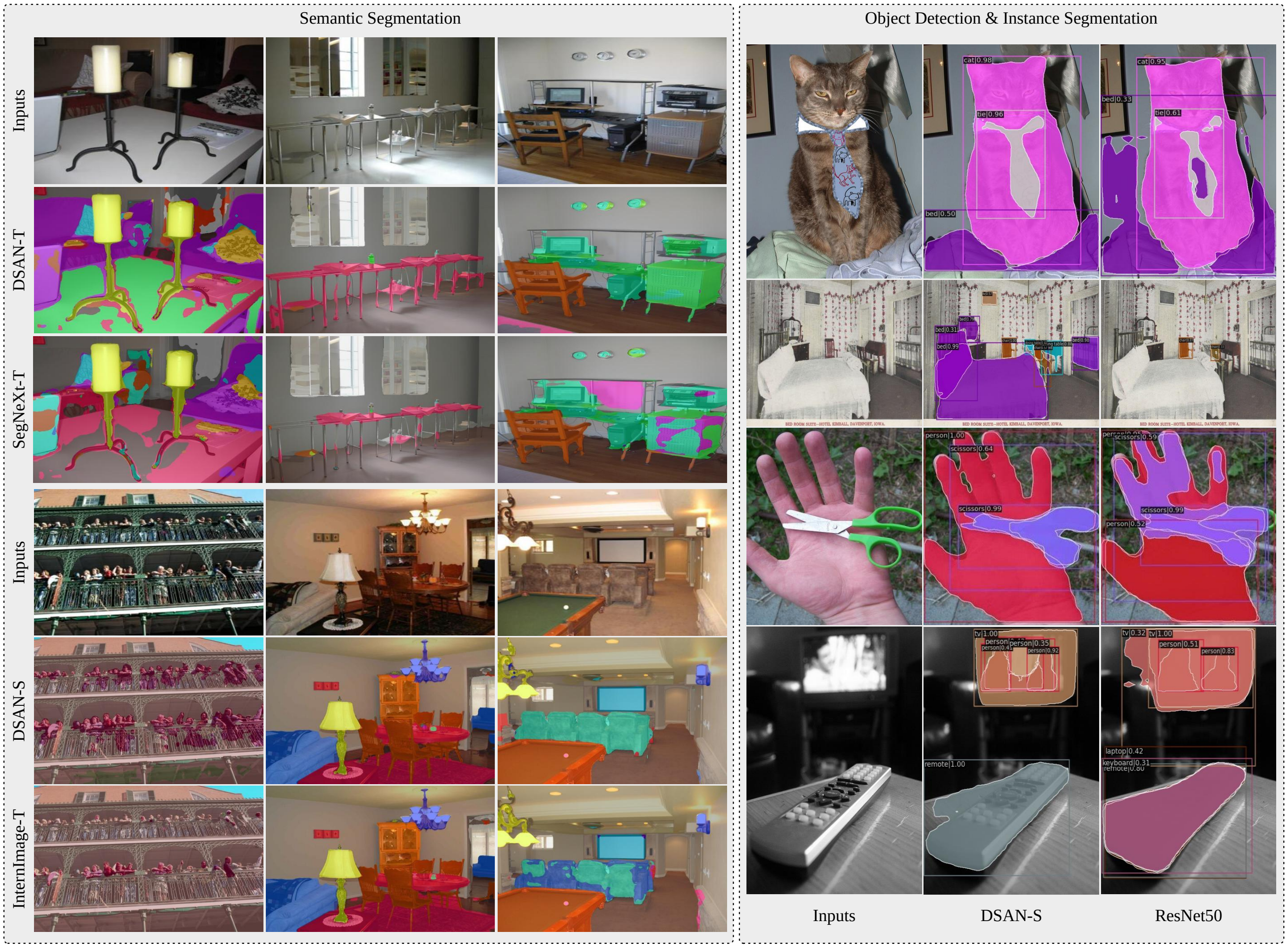


Fig. 8. he visualization showcases distinct downstream tasks. The left column presents semantic segmentation masks from the ADE20K validation set, while another column exhibits anchor boxes and instance segmentation masks of the COCO 2017 validation set.

The visualization results of these masks allow us to indirectly judge the influence of the receptive field on the entire model through the segmentation effect of semantic targets, such as clearer segmentation edges. The visualizations of the segmentation masks and anchor boxes are presented in Fig. 8.

Upon visualizing the semantic segmentation masks, we observe that DSAN-T consistently exhibits superior shape segmentation capabilities for irregular objects like candle holders and connected desks in comparison to SegNeXt-T, which possesses a regular receptive field. In the present analysis, when comparing the semantic segmentation masks produced by DSAN-S against InternImage-T, it is clear that using InternImage-T based on DCNv3 with smaller kernels alongside a compact parameter set often results in insufficient sampling within certain complex scenes. This insufficiency leads to incomplete segmentation, as seen with crowds not fully segmented within railings. In the object detection and instance segmentation visualization results, we find that DSAN-S is more adaptive than ResNet50 to some confused scenarios, such as a hand with fingers spread open and a pair of scissors in the same picture. The visualization results show that DSCN and DSA can effectively sample globally, improving the shape adaptability of lightweight CNNs.

V. CONCLUSION

Our method provides a solution for using DCNv3 in lightweight CNNs. To tackle the slow training speed and high memory usage of DCNv3 with large kernel, we optimize it at two levels: core operations and visual feature extraction units. At the core operations level, we deconstruct the deformation sampling core operation in DCNv3 into two strip deformable convolution core operations. We change the bilinear interpolation calculation for deformation sampling to linear interpolation and remove modulation mask-related computations, leading to a reduction in computational load. At the feature extraction unit level, we find that the mask branch in DCNv3 is similar to spatial attention. Hence, we replace the modualtion mask branch with spatial attention to cut down on parameters and memory consumption. Based on these improvements, we design DSCN and a more suitable visual feature extraction module DSA for lightweight CNNs. This design maintains the globally deformable receptive field from DCNv3 with large kernel while avoiding its drawbacks. To validate the effectiveness of DSCN and DSA, we construct a lightweight CNN backbone named DSAN, using DSA as the main visual feature extraction module. Through ablation studies, we confirm that DCNv3 does not fully demonstrate its inherent advantages when applied to lightweight CNNs and validate effectiveness of our designs. Moreover, we test the generality of DSAN across various vision tasks such as image classification, semantic segmentation, and object detection. Our designs improve training speed and reduce memory consumption during training. Our work is successful in semantic segmentation tasks, where DSAN-S can achieve better segmentation results with fewer parameters and computational requirements than InternImage-T. The disadvantage is that on other vision tasks, there is still a slight gap between InternImage-T and DSAN-S. Overall, our designs for DSCN and DSA effectively enhance the shape adaptability of lightweight CNNs while maintaining their lightweight state.

REFERENCES

[1] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.

[2] X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31x31: Revisiting large kernel design in cnns," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 11 963–11 975.

[3] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Sep. 2016.

[4] M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, and S.-M. Hu, "Visual attention network," arXiv: 2202.09741, Feb. 2022.

[5] M.-H. Guo, C.-Z. Lu, Q. Hou, Z.-N. Liu, M.-M. Cheng, and S. min Hu, "Segnext: Rethinking convolutional attention design for semantic segmentation," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Nov. 2022.

[6] Z. Xia, X. Pan, S. Song, L. E. Li, and G. Huang, "Vision transformer with deformable attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 4784–4793.

[7] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convnets v2: More deformable, better results," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9300–9308.

[8] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li *et al.*, "Internimage: Exploring large-scale vision foundation models with deformable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 14 408–14 419.

[9] Y. Qi, Y. He, X. Qi, Y. Zhang, and G. Yang, "Dynamic snake convolution based on topological geometric constraints for tubular structure segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 6070–6079.

[10] R. Lin, J. C. L. Li, J. Zhou, B. Huang, J. Ran, and N. Wong, "Lite it fly: An all-deformable-butterfly network," *IEEE Trans. Neural Netw. Learn. Syst.*, Nov. 2023, early access, doi: 10.1109/TNNLS.2023.3333562.

[11] C. Xing, Y. Cong, C. Duan, Z. Wang, and M. Wang, "Deep network with irregular convolutional kernels and self-expressive property for classification of hyperspectral images," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 10 747–10 761, 2023.

[12] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Yang, and X.-C. Yin, "Kernel proposal network for arbitrary shape text detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 8731–8742, 2023.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.

[14] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[15] H. Caesar, J. Uijlings, and V. Ferrari, "Coco-stuff: Thing and stuff classes in context," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 1209–1218.

[16] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2017, pp. 5987–5995.

[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Jan. 2021.

[18] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pvt v2: Improved baselines with pyramid vision transformer," *Computational Visual Media*, vol. 8, no. 3, pp. 415–424, 2022.

[19] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 7132–7141.

[20] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proc. Eur. Conf. on Comput. Vis. (ECCV)*, Sep. 2018.

[21] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 7794–7803.

[22] W. Cai, H. Sun, R. Liu, Y. Cui, J. Wang, Y. Xia, D. Yao, and D. Guo, "A spatial–channel–temporal-fused attention for spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, 2023, early access, doi: 10.1109/TNNLS.2023.3278265.

[23] Y. Zhou, Z. Chen, P. Li, H. Song, C. L. P. Chen, and B. Sheng, "Fsad-net: Feedback spatial attention dehazing network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7719–7733, 2023.

[24] R. Liu, Z.-A. Huang, Y. Hu, Z. Zhu, K.-C. Wong, and K. C. Tan, "Spatial–temporal co-attention learning for diagnosis of mental disorders from resting-state fmri data," *IEEE Trans. Neural Netw. Learn. Syst.*, 2023, early access, doi: 10.1109/TNNLS.2023.3243000.

[25] H. Wang, X. Xiang, Y. Tian, W. Yang, and Q. Liao, "Stdan: Deformable attention network for space-time video super-resolution," *IEEE Trans. Neural Netw. Learn. Syst.*, 2023, early access, doi: 10.1109/TNNLS.2023.3243029.

[26] C. Xu, K. Li, X. Luo, X. Xu, S. He, and K. Zhang, "Fully deformable network for multiview face image synthesis," *IEEE Trans. Neural Netw. Learn. Syst.*, 2022, early access, doi: 10.1109/TNNLS.2022.3216018.

[27] Q. Zhu, F. Chen, S. Zhu, Y. Liu, X. Zhou, R. Xiong, and B. Zeng, "Dvsrnet: Deep video super-resolution based on progressive deformable alignment and temporal-sparse enhancement," *IEEE Trans. Neural Netw. Learn. Syst.*, 2024, early access, doi: 10.1109/TNNLS.2023.3347450.

[28] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2017, pp. 5122–5130.

[29] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Dec. 2019.

[30] R. Wightman, "Pytorch image models," 2019. [Online]. Available: https://github.com/rwightman/pytorch-image-models

[31] MMSegmentation Contributors, "MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark," 2020. [Online]. Available: https://github.com/open-mmlab/mmsegmentation

[32] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," arXiv:1906.07155, Jun. 2019.

[33] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jul. 2021, pp. 10 347–10 357.

[34] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2019.

[35] I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with restarts," arXiv:1608.03983, Aug. 2016.

[36] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou, "Going deeper with image transformers," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 32–42.

[37] G. Larsson, M. Maire, and G. Shakhnarovich, "Fractalnet: Ultra-deep neural networks without residuals," arXiv:1605.07648, May 2016.

[38] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[39] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.

[40] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Nov. 2021.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[42] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, and S. Yan, "Metaformer is actually what you need for vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10 809–10 819.

[43] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 568–578.

[44] H. Liu, Z. Dai, D. So, and Q. V. Le, "Pay attention to MLPs," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Nov. 2021.

[45] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10 425–10 433.

[46] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10 012–10 022.

[47] W. Wang, E. Xie, X. Li, D. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pvtv2: Improved baselines with pyramid vision transformer," arXiv:2106.13797, Apr. 2021.

[48] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen, "Twins: Revisiting the design of spatial attention in vision transformers," *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Nov. 2021.

[49] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 11 966–11 976.

[50] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2017, pp. 6230–6239.

[51] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Proc. Eur. Conf. on Comput. Vis. (ECCV)*, Sep. 2018.

[52] Z. Geng, M.-H. Guo, H. Chen, X. Li, K. Wei, and Z. Lin, "Is attention better than matrix decomposition?" in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2021.

[53] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2017, pp. 5168–5177.

[54] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2018.

[55] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. on Comput. Vis. (ECCV)*, Sep. 2018.

[56] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.