This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Slide-Transformer: Hierarchical Vision Transformer with Local Self-Attention

Xuran Pan* Tianzhu Ye* Zhuofan Xia Shiji Song Gao Huang†
Department of Automation, BNRist, Tsinghua University

Abstract

Self-attention mechanism has been a key factor in the recent progress of Vision Transformer (ViT), which enables adaptive feature extraction from global contexts. However, existing self-attention methods either adopt sparse global attention or window attention to reduce the computation complexity, which may compromise the local feature learning or subject to some handcrafted designs. In contrast, local attention, which restricts the receptive field of each query to its own neighboring pixels, enjoys the benefits of both convolution and self-attention, namely local inductive bias and dynamic feature selection. Nevertheless, current local attention modules either use inefficient Im2Col function or rely on specific CUDA kernels that are hard to generalize to devices without CUDA support. In this paper, we propose a novel local attention module, **Slide Attention**, *which leverages common convolution operations to achieve high efficiency, flexibility and generalizability. Specifically, we first re-interpret the* column-based *Im2Col function from a new* row-based *perspective and use Depthwise Convolution as an efficient substitution. On this basis, we propose a deformed shifting module based on the re-parameterization technique, which further relaxes the fixed key/value positions to deformed features in the local region. In this way, our module realizes the local attention paradigm in both efficient and flexible manner. Extensive experiments show that our slide attention module is applicable to a variety of advanced Vision Transformer models and compatible with various hardware devices, and achieves consistently improved performances on comprehensive benchmarks.*

1. Introduction

Transformer was originally proposed for natural language processing [5,29] and has gained increasing research interest in recent years. With the advent of Vision Transformer [9], researchers begin to realize its great potential on processing vision data, and further extend Transformer models to a variety of vision tasks including image classification [20, 21, 30], semantic segmentation [26, 36], object detection [2, 16, 22, 39], and multi-modal tasks [23, 24].

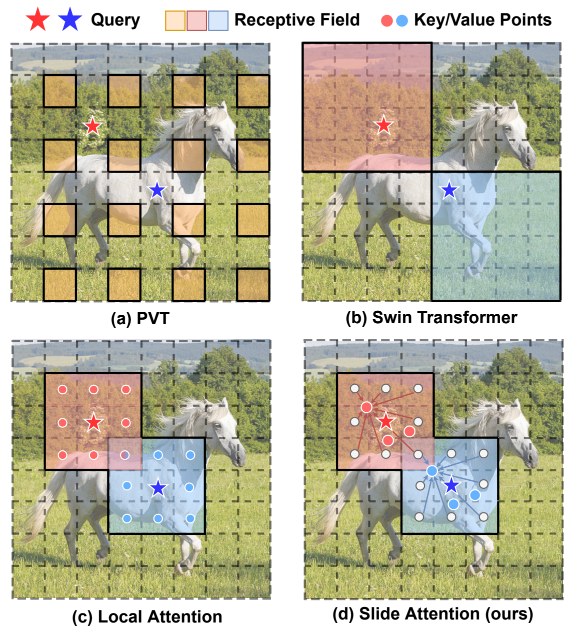


Figure 1. **Comparison of our model and other attention patterns.** Comparing to the global attention in PVT and window attention in Swin-Transformer, we propose a novel Slide Attention module that not only imposes local inductive bias like local attention, but also has high efficiency and flexibility.

Nevertheless, adapting Transformer to vision is a non-trivial task. The computation complexity of self-attention with global receptive field grows quadratically with the sequence length, which leads to excessive computation costs and makes it impractical for vision models that require high-resolution inputs and large memory consumption.

To overcome this challenge, existing works have proposed to limit the global receptive field to smaller regions. For example, PVT [30] and DAT [33] use sparse global attention to select sparse key and value positions from the feature map and share them across all queries. Another line of research including Swin Transformer [20] and CSwin Transformer [8] follow the window attention paradigm. The input is divided into specially designed windows, where features are extracted and aggregated within. Despite be-

*Equal contribution.
†Corresponding author.

ing efficient, these carefully designed attention patterns still suffer from several limitations. On one hand, sparse global attention tends to be inferior in capturing local features, and is susceptible to key and value positions where informative features in other regions may be discarded. On the other hand, window attentions may hinder cross-window communication, and involve extra designs like window shifts that set restrictions on the model structure.

Instead of shrinking the global receptive field, a natural and effective alternative is adopting local attention by constraining receptive field of each query in its own neighboring pixels, where similar pattern has been widely used in traditional convolution design [6, 13]. Compared with the aforementioned attention patterns, local attention has the advantages of convolution with translation-equivariance and local inductive bias, while also enjoying the flexibility and data-dependency of the self-attention mechanism. Several works have already investigated applying local attention to modern convolution or Transformer models. However, they either use the inefficient Im2Col function [25] which results in huge increase in inference time, or rely on carefully written CUDA kernels [11, 37] which restrict the applicability on devices without CUDA support. Therefore, developing a local attention module with both high efficiency and high generalizability remains challenging.

In this paper, we present a novel local attention module, dubbed *Slide Attention*, that can be efficiently integrated with various Vision Transformer models and hardware devices. We target the inefficient Im2Col function that was adopted in the previous local attention module and view the process from a new perspective. Specifically, the original Im2Col generates the key and value matrix from a **column-based view**, where each column represents a local region centered at a certain position of the input. Alternatively, we re-formulate the key and value matrix from a **row-based view** and show that each row corresponds to the input feature shifted in different directions. This new insight gives us the chance to take a further step, that allows us to replace the shifting operation with carefully designed Depthwise Convolutions. In this way, the Im2Col function is replaced with standard convolution operations, which can be realized in a more efficient manner and easily implemented on different hardware devices. To further enhance flexibility, we introduce a novel deformed shifting module that relaxes fixed key and value positions (Fig.1(c)) to deformed features within the local region (Fig.1(d)). By using a re-parameterization technique, we effectively increase the model capacity while preserving inference efficiency.

We empirically validate our module on image classification, semantic segmentation, and object detection tasks under five advanced Vision Transformer models, and show consistent improvements over all baselines. When adopted on devices without CUDA support like Metal Performance

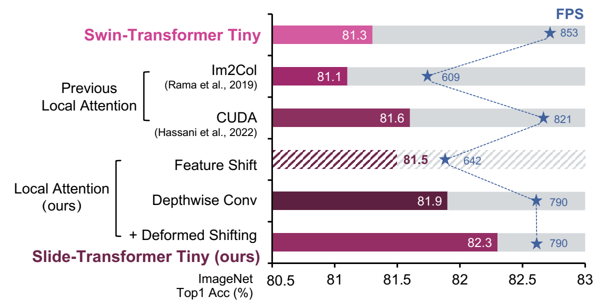


Figure 2. **Performance and inference speed comparison on local attention implementations.** Results are based on Swin-Tiny [20]. Previous works mainly use Im2Col function [25] or carefully designed CUDA kernels [11], where the former is highly inefficient and the latter only shows marginal improvements over other attention patterns, *e.g.,* window attention in Swin-Transformer, and hard to generalize to other devices. Our work first re-interprets the Im2Col function as feature shift operations, then substitute shifts with more efficient depthwise convolutions. When further equipped with a deformed shifting module, our model achieves significant improvements over baselines under competitive inference time. FPS is tested on an RTX3090 GPU.

Shader (MPS) or iPhone 12, our method also proves to be efficient. For instance, our Slide Attention based on Swin-small outperforms the vanilla Swin-base model while achieving 1.7x inference speedup on iPhone 12.

2. Related Works

2.1. Vision Transformer

Transformer and the self-attention mechanism have shown great progress in the field of Natural Language Processing [5, 29] and successfully applied to vision tasks thanks to the pioneering work of Vision Transformer [9]. Following its path, researchers have extended Vision Transformer models along various directions, including data efficiency [27], position encoding [32], and optimization [35]. To better adapt Vision Transformers to downstream tasks, several works focused on investigating pyramid model structures, and show advanced performances over convolution-based approaches. PVT [30, 31] considers sampling sparse locations in the feature map as key and value pairs. DAT [33] takes a further step and shifts fixed locations toward different directions in a data-dependent way. MViT [10, 17] considers the pooling function on the input to obtain key and value pairs, which can be seen as a lower resolution of the feature map. Other approaches adopt an alternative strategy and restrict the attention to carefully designed patterns. Swin Transformer [20] designs non-overlapped windows and shifts windows between consecutive blocks. On this basis, CSwin Transformer [8] adopts a cross-shape window to further improve model capacity.

2.2. Local Attention

By constraining the attention receptive field of each query in its own neighboring pixels, local attention inherits the advantages from traditional convolution including local inductive bias and translation-equivariance [25]. Researchers follow this path and target improving the efficiency of local attention. HaloNet [28] combines window attention with local attention by first dividing the input into blocks and considering neighborhood windows instead of pixels. Another direction is to design CUDA kernels with high inference speed. SAN [37] designs a novel patchwise attention pattern and achieves better performances based on convolution architectures. NAT [11] adopts neighborhood attention and specifically considers situations for corner pixels. Nevertheless, current local attention models either use inefficient Im2Col function and endure huge increase in inference time, or rely on carefully written CUDA kernels that restrict applicability on CUDA-free devices.

3. Overview of Self-Attention

In this section, we first provide an overview of the self-attention module and its various forms. Compared to the widely used sparse global attention and window attention paradigm, local attention tends to be the most natural implementation while suffering from efficiency limitations.

3.1. Multi-Head Self-Attention

Multi-head self-attention (MHSA) is the core component of Transformer models, which is also the most distinct part among the numerous Transformer researches. In general, an MHSA block with M heads can be formulated as:

$$q = xW_q,\ k = xW_k,\ v = xW_v, \tag{1}$$

$$z^{(m)} = \sigma(q^{(m)} \cdot k_{[r_q]}^{(m)\top} / \sqrt{d}) \cdot v_{[r_q]}^{(m)}, m=1,\dots,M, \tag{2}$$

$$z = \text{Concat}\left(z^{(1)},\dots,z^{(M)}\right) W_o, \tag{3}$$

where $\sigma(\cdot)$ denotes the SoftMax function, and d is the channel dimension of each head. In particular, we denote r_q as the receptive field of a specific query q, and denote $k_{[r_q]}^{(m)}$ and $v_{[r_q]}^{(m)}$ as the corresponding key and value pairs respectively.

3.2. Attention Patterns

The implementation of self-attention in the field of computer vision is never a trivial task. Like a coin has two sides, the high flexibility of the self-attention mechanism leads to higher computation complexity and lower efficiency on hard-wares. Therefore, to achieve a better trade-off between performance and efficiency, previous works have investigated injecting different inductive biases into vanilla self-attention paradigm by designing different attention patterns.

(1) Sparse Global Attention [30, 33] considers selecting a sparse set of key and value pairs instead of the dense feature map. However, this also restricts the potential of feature extraction into a limited subset of input. Also, the key and value pairs are the same for all queries. This query-agnostic selection strategy may lead to a homogenization of features throughout the whole feature map.

(2) Window Attention [8, 20] is another option to carefully divide input into particular windows where features are extracted within. Although partially addressing the limitation of query-agnostic key and value pairs, the designed patterns may lead to unnatural circumstances where features at the edge of different windows are totally isolated despite being close in the feature map. Also, window patterns need to shift between consecutive blocks to facilitate connections across windows, involving extra designs in model structure.

(3) Local Attention constrains the receptive field of each query in its own neighboring pixels, sharing a similar pattern with convolution. Compared to former patterns, local attention enjoys the advantages from both convolution and self-attention: 1) Local inductive bias from a query-centric attention pattern; 2) Translation-equivariance like traditional convolution, showing robustness towards shift variances of input; 3) Involving little human design, which sets the least restrictions on the model architecture design.

3.3. Local Attention Implementation

Despite being effective, the local receptive field also poses difficulties in practical implementation. Specifically, due to the fact that the receptive region is different for each query in the feature map, special technique, *i.e.,* Im2Col function needs to be adopted to sample keys and values for all the queries respectively. As illustrated in Fig.3(1), local window is centered at a particular query and represents the region of its corresponding key/value pairs. The windows are then flattened into **columns** and consist of the final key/value matrix. However, the process of sampling windows is mainly achieved by independently slicing the feature map, which practically breaks data locality and leads to huge time consumption. In the case of convolution, special tricks like Winograd [15] can be adopted where a portion of computations can be pre-computed before the inference stage. However, the tricks fail to generalize to local attention, since the *'kernel weights'* are computed by the dot product of queries and keys in a data-dependent way.

Another line of research [11, 37] focuses on improving the efficiency of local attention by writing CUDA kernels to replace the inefficient Im2Col function. Albeit effective, this inevitably sets restrictions on the potential applicability, making it impractical on hardwares without CUDA support, especially on edge devices like smartphones.

To show a thorough comparison between the aforementioned approaches, we practically analyze the performance

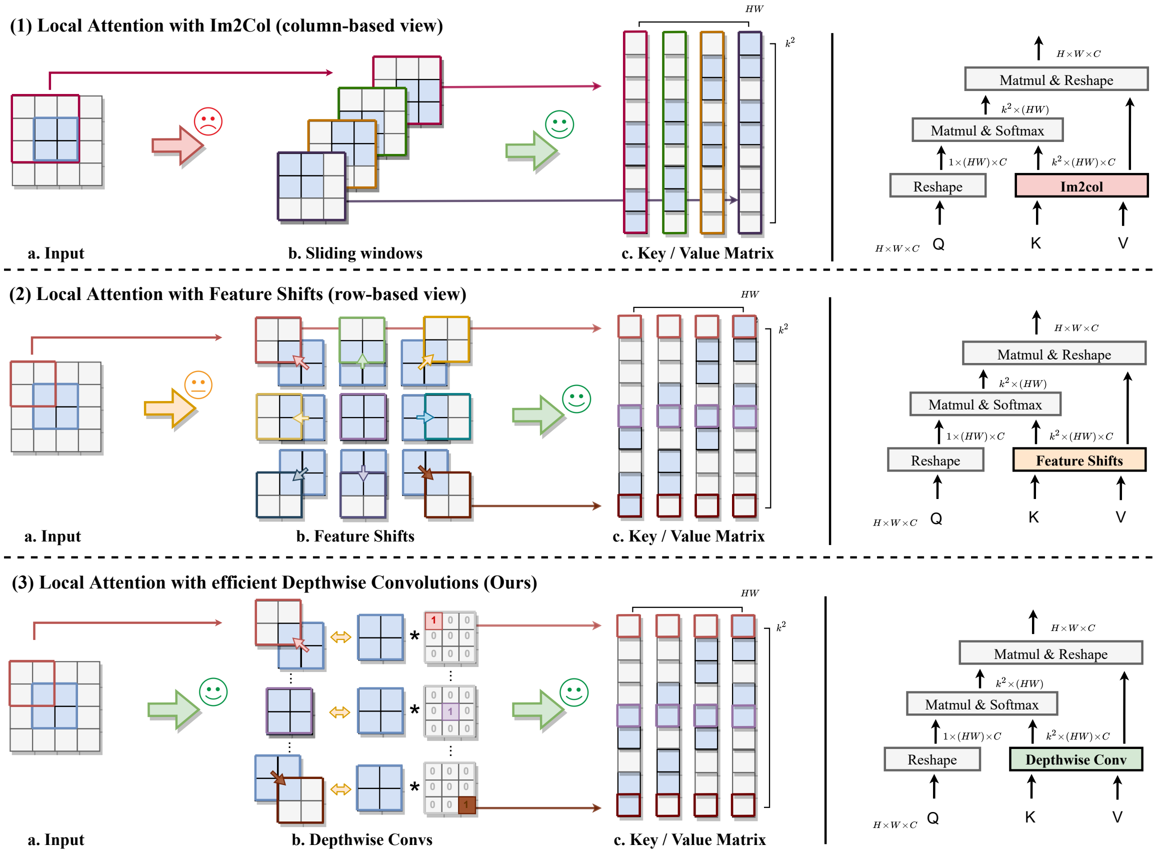


Figure 3. **Different implementation on the local attention module.** We take 3x3 local attention on a 2x2 feature map (in blue) with [1,1] padding (in gray) as an example. **Sub-figure(1)**: Im2Col function is viewed in a *column-based* way, where each column of the key/value matrix corresponds to the local region of a particular query (1.b). The process of sampling windows breaks data locality and leads to inefficiency ✗. **Sub-figure(2)**: we view the key/value matrix in a *row-based* way, where each row is equivalent to the input feature, only after shifting towards certain directions (2.b). Nevertheless, shifting toward different directions is also inefficient when compared with common operators ✗. **Sub-figure(3)**: we take a step forward, and substitute shifting operations with carefully designed depthwise convolutions, which is not only efficient but also friendly to different hardware implementations ✓. Best viewed in color.

and runtime of these two implementations of local self-attention and compare it with the window attention in Swin-Transformer. As illustrated in Fig.2, Im2Col-based local attention is less favorable in both efficiency and performance. The CUDA-based approaches can maintain comparable inference speed with vectorized operations like window attention, while only achieving marginal improvements. Considering the difficulty of adopting CUDA kernels on different hardwares, we still lack a local attention module that has both high efficiency and high generalizability.

4. Method

As analyzed above, local attention suffers from the efficiency problem that prevents it from practical implementation. In this section, we first show that the inefficient Im2Col function can be re-interpreted from another perspective and proved to be equivalent to a group of simple feature shifts. On this basis, we substitute feature shift operations with efficient depthwise convolutions. Equipped with a novel deformed shifting module to relax the fixed local key/value positions to deformed features, we finally propose a local attention module, dubbed *Slide Attention*, with high efficiency and flexibility.

4.1. New Perspective on Im2Col

For better understanding, we first review the process of the Im2Col function. We take the operations on keys as an example in the following section, and the case for values is exactly the same. Let $\mathbf{K} \in \mathcal{R}^{H\times W\times C}$ denote the keys of the self-attention module and k denote the local window size, the output of Im2Col can be represented as:

$$O_k[u*k+v, i*H+j] = \mathbf{K}[i+u, j+v], \quad (4)$$

$$\text{for } i\in[0, W-1], j\in[0, H-1], u, v\in[-\lfloor k/2\rfloor, \lfloor k/2\rfloor]. \quad (5)$$

From the **column-based** view, as illustrated in Fig.3[1], the key/value matrix contains HW columns where each column corresponds to a local window centered at a particular query. Specifically, if we carefully check each column of the output, the above equations can be reformulated as:

$$O_k[:, i*H+j] = \mathrm{Column}^{(i,j)}, \tag{6}$$

$$\text{where } \mathrm{Column}^{(i,j)}(u,v) = \mathbf{K}[i+u, j+v] \tag{7}$$

represents a local window centered at (i, j). This is in accordance with motivation of Im2Col function, where receptive windows of all queries are sampled and placed in order.

However, an interesting observation is we can also view the Im2Col function in a different way. From the **row-based** view, as illustrated in Fig.3(2), the key/value matrix contains k^2 rows, where each row corresponds to shifting input towards a certain direction. Specifically, we focus on each row of the output and reformulate the above equations as:

$$O_k[u*k+v, :] = \mathrm{Row}^{(u,v)}, \tag{8}$$

$$\text{where } \mathrm{Row}^{(u,v)}(i,j) = \mathbf{K}[i+u, j+v] \tag{9}$$

is equivalent to shifting the original feature map towards a certain direction $(u, v) \in [-\lfloor k/2 \rfloor, \lfloor k/2 \rfloor]$.

In this way, we offer a new alternative to understanding the Im2Col function by substituting the **column-based** view with a novel **row-based** view. Take $k = 3$ as an example, if we first shift the original feature map towards 9 different directions (Fig.3(2.b)), then flatten these features into rows and finally concatenate them in column (Fig.3(2.c)), the obtained key/value matrix is proved equivalent to HW local windows which can recover the exact same output of the original Im2Col function (Fig.3(1.c)).

4.2. Shift as Depthwise Convolution

Although the re-interpretation in Sec.4.1 provides us a new way to understand the Im2Col function, simply shifting features towards different directions still involves inefficient slicing operations, which provide little help in promoting the efficiency of local attention. Nevertheless, unlike sampling windows of all queries in Im2Col, feature shifting can be achieved in a more efficient way.

Specifically, we resort to applying depthwise convolution with designed kernels as a replacement for the inefficient feature shifts, as shown in Fig.3(3). Take $u = -1$ and $v = -1$ in Eq.(9) as an example, for a input $f \in \mathcal{R}^{H \times W \times C}$, shifting towards direction $(-1, -1)$ can be formulated as:

$$\tilde{f}[i, j, :] = f[i-1, j-1, :], \ \forall i, j. \tag{10}$$

On the other hand, if we denote the depthwise convolution kernel (kernel size $k = 3$) as:

$$K[:, :, c] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ \forall c, \tag{11}$$

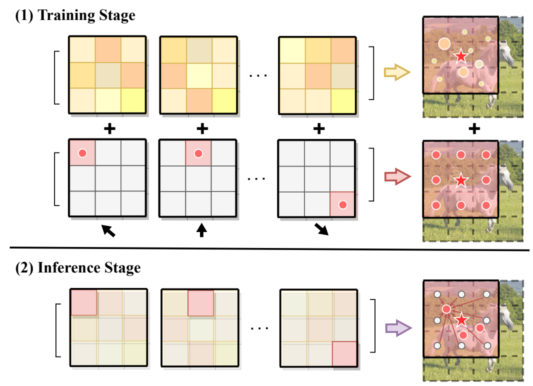


Figure 4. **Deformed shifting module with re-parameterization.** (1) At the training stage, we maintain two paths, one with designed kernel weights to perform shifting towards different directions, and the other with learnable parameters to enable more flexibility. (2) At the inference stage, we merge these two convolution operations into a single path with re-parameterization, which improves the model capacity while maintaining the inference efficiency.

the corresponding output can be formulated as:

$$f^{(\mathrm{dwc})}[i, j, c] = \sum_{p,q \in \{0,1,2\}} K[p, q, c] f[i{+}p{-}1, j{+}q{-}1, c] \tag{12}$$

$$= f[i{-}1, j{-}1, c] = \tilde{f}[i, j, c], \ \forall i, j, c. \tag{13}$$

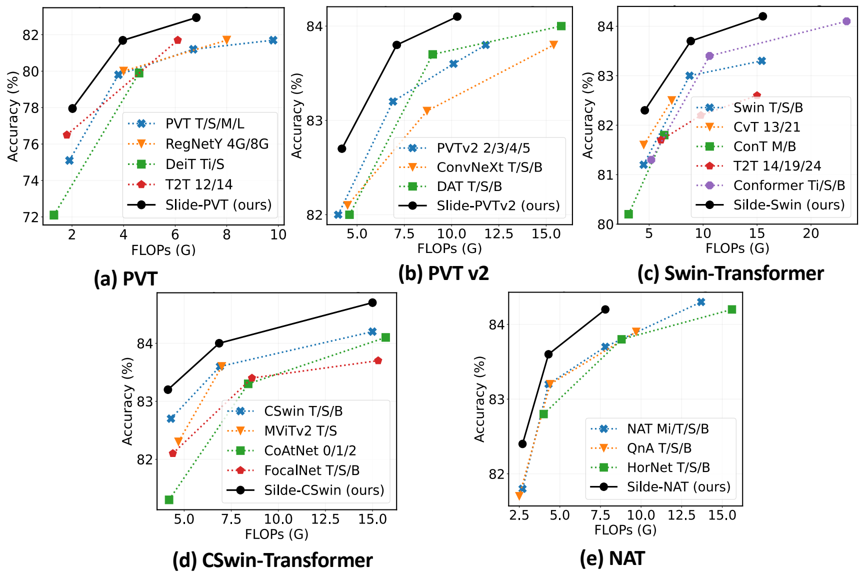
Therefore, with carefully designed kernel weights for different shift directions, the convolution outputs are equivalent to the simple feature shifts.

In general, we can integrate the findings from Sec.4.1 and Sec.4.2 and propose an efficient implementation of local attention. For local attention with window size k, we can re-implement the Im2Col function as k^2 carefully defined depthwise convolutions, alleviating the main overhead. Moreover, these depthwise convolutions can be further boiled down to a single group convolution, which not only avoids the inefficient slicing operation, but also can benefit from the optimized implementation of convolution operations on many hardwares [6, 15].

4.3. Deformed Shifting Module

By switching the original Im2Col function to depthwise convolutions, the efficiency of the local attention is greatly improved. Nevertheless, the carefully designed kernel weights still constrain keys and values to the fixed neighboring positions, which may not be the optimal solution for capturing diverse features.

Therefore, we propose a novel *deformed shifting module* to further enhance the flexibility of local attention. In specific, we take advantage of design paradigm in our shift-wise convolution, and introduce a parallel convolution path where the kernel parameters are randomly initialized and learnable in the training process. Compared to fixed kernels that shift features towards different directions, learn-



Method	Params	Flops	Top-1
PVT-T [30]	13.2M	1.9G	75.1
Slide-PVT-T	12.2M	2.0G	**78.0** (+2.9)
PVT-S	24.5M	3.8G	79.8
Slide-PVT-S	22.7M	4.0G	**81.7** (+1.9)
PVTv2-B1 [31]	13.1M	2.1G	78.7
Slide-PVTv2-B1	13.0M	2.2G	**79.5** (+0.7)
PVTv2-B2	25.4M	4.0G	82.0
Slide-PVTv2-B2	22.8M	4.2G	**82.7** (+0.7)
Swin-T [20]	29M	4.5G	81.3
Slide-Swin-T	29M	4.6G	**82.3** (+1.0)
Swin-S	50M	8.7G	83.0
Slide-Swin-S	51M	8.9G	**83.7** (+0.7)
Swin-B	88M	15.4G	83.5
Slide-Swin-B	89M	15.5G	**84.2** (+0.7)
CSwin-S [8]	35M	6.9G	83.6
Slide-CSwin-S	35M	6.9G	**84.0** (+0.4)
CSwin-B	78M	15.0G	84.2
Slide-CSwin-B	78M	15.0G	**84.7** (+0.5)
NAT-T [8]	28M	4.3G	83.2
Slide-NAT-T	28M	4.3G	**83.6** (+0.4)
NAT-S	51M	7.8G	83.7
Slide-NAT-S	51M	7.8G	**84.3** (+0.6)

Figure 5. Comparisons of FLOPS and parameters against accuracy on ImageNet classification task. Models in (a)(b) adapt from PVT and PVTv2 with global attention; Models in (c)(d) adapt from Swin-Transformer and CSwin-Transformer with window attention; Models in (e) adapt from NAT with local attention. See the full comparison table in Appendix.

able kernels can be interpreted as a linear combination of all local features. This is in analogy with the deformed receptive field in Deformable Convolutional Network [3], where our module practically relaxes the fixed key and value positions to deformed features in the local region.

As illustrated in Fig.4, the additional convolution path improves local attention module from several perspectives:

(1) The key and value pairs in the local attention are extracted by a more flexible module, that greatly improves model capacity and can capture diverse features.

(2) The learnable convolution kernel shows a resemblance with the deformable technique in DCN. Similar to the bilinear interpolation of four neighboring pixels in DCN, our deformed shifting module can be viewed as a linear combination of features within the local window. This finally contributes to augmenting the spatial sampling locations and model geometric transformation of inputs.

(3) We use the re-parameterization technique [7] to transform the two parallel paths into a single convolution. In this way, we can improve the model capacity while maintaining inference efficiency.

4.4. Implementation

On the basis of the aforementioned design, we propose a novel *Slide Attention* module that enables a highly efficient and flexible local attention pattern and poses little restriction on model architecture design. Our block can serve as a plug-in module and is easily adopted on a variety of modern vision Transformer architectures and hardware devices. As a showcase, we empirically implement our module on five advanced models including PVT [30], PVT-v2 [31], Swin Transformer [20], CSwin Transformer [8] and NAT [11], and conduct experiments on several environments including Nvidia GPU, Medal Performance Shader and iPhone 12.

Also, previous works [35] have demonstrated that the locality and translation-equivariant property in convolutions are beneficial at early stages of vision Transformers. Considering the similar design pattern and characteristics between our module and traditional convolution, we simply adopt the slide attention block at the early stages of vision Transformer models, and keep the rest of the block unchanged. The detailed architectures are shown in Appendix.

5. Experiments

We conduct experiments on several datasets to verify the performance of our Slide Attention module. We show comparison results on ImageNet [4] classification, ADE20K [38] semantic segmentation and COCO [19] object detection tasks. We also provide a detailed comparison with other local attention modules based on two representative model structures. In addition, ablation studies are conducted to show the effectiveness of the designs in our module. See Appendix for detailed dataset and training configurations.

5.1. ImageNet-1K Classification

ImageNet-1K [4] contains 1.28M images for training and 50K images for validation. We practically implement our module on five advanced Vision Transformer models, and compare with various state-of-the-art models.

We show the classification results in Fig.5. It is shown that our method achieves consistent improvements against baseline models under comparable FLOPs or parameters.

Method	FLOPs	#Param	Schedule	AP^b	AP^b_{50}	AP^b_{75}	AP^b_s	AP^b_m	AP^b_l	AP^m	AP^m_{50}	AP^m_{75}	AP^m_s	AP^m_m	AP^m_l
(a) Mask R-CNN Object Detection & Instance Segmentation on COCO															
PVT-T	240G	33M	1x	36.7	59.2	39.3	21.6	39.2	49.0	35.1	56.7	37.3	19.5	37.4	48.5
Slide-PVT-T	219G	32M	1x	40.4	63.4	43.8	25.3	42.8	53.0	38.1	60.4	41.0	20.0	40.1	55.2
PVT-S	305G	44M	1x	40.4	62.9	43.8	22.9	43.0	55.4	37.8	60.1	40.3	20.4	40.3	53.6
Slide-PVT-S	269G	42M	1x	42.8	65.9	46.7	26.6	45.5	57.3	40.1	63.1	43.1	20.3	42.4	59.0
PVT-M	392G	64M	1x	42.0	64.4	45.6	24.4	44.9	57.9	39.0	61.6	42.1	21.3	42.0	55.2
Slide-PVT-M	357G	62M	1x	44.4	66.9	48.6	28.9	47.0	59.4	40.8	63.9	43.8	25.0	43.5	55.9
PVTv2-B1	244G	34M	1x	41.8	64.3	45.9	26.4	44.9	54.3	38.8	61.2	41.6	20.2	41.3	56.1
Slide-PVTv2-B1	222G	33M	1x	42.6	65.3	46.8	27.4	45.6	55.7	39.7	62.6	42.6	24.1	42.9	53.7
PVTv2-B2	309G	45M	1x	45.3	67.1	49.6	28.8	48.4	59.5	41.2	64.2	44.4	22.0	43.7	59.4
Slide-PVTv2-B2	274G	43M	1x	46.0	68.2	50.3	28.8	49.4	61.0	41.9	65.1	45.4	24.6	45.2	57.2
Swin-T	267G	48M	3x	46.0	68.1	50.3	31.2	49.2	60.1	41.6	65.1	44.9	25.9	45.1	56.9
Slide-Swin-T	268G	48M	3x	46.8	69.0	51.6	31.7	50.4	60.1	42.3	66.0	45.8	23.5	45.8	60.8
(b) Cascade Mask R-CNN Object Detection & Instance Segmentation on COCO															
Method	FLOPs	#Param	Schedule	AP^b	AP^b_{50}	AP^b_{75}	AP^b_s	AP^b_m	AP^b_l	AP^m	AP^m_{50}	AP^m_{75}	AP^m_s	AP^m_m	AP^m_l
Swin-T	745G	86M	3x	50.4	69.2	54.7	33.8	54.1	65.2	43.7	66.6	47.3	27.3	47.5	59.0
Slide-Swin-T	747G	86M	3x	51.1	69.8	55.4	35.2	54.4	65.8	44.3	67.4	48.0	28.0	48.0	59.2
Swin-S	838G	107M	3x	51.9	70.7	56.3	35.2	55.7	67.7	45.0	68.2	48.8	28.8	48.7	60.6
Slide-Swin-S	838G	107M	3x	52.5	71.3	57.2	35.6	56.1	68.0	45.4	68.9	49.6	29.1	49.2	60.6
Swin-B	981G	145M	3x	51.9	70.5	56.4	35.4	55.2	67.4	45.0	68.1	48.9	28.9	48.3	60.4
Slide-Swin-B	983G	145M	3x	52.7	71.2	57.2	37.0	56.1	68.0	45.5	68.8	49.6	30.1	48.8	60.9

Table 1. Results on COCO dataset. The FLOPs are computed over backbone, FPN and detection head with input resolution of 1280×800.

RetinaNet Object Detection on COCO (Sch. 1x)

Method	FLOPs	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
PVT-T	221G	36.7	56.9	38.9	22.6	38.8	50.0
Slide-PVT-T	200G	40.1	61.1	42.2	25.9	43.3	54.2
PVT-S	286G	38.7	59.3	40.8	21.2	41.6	54.4
Slide-PVT-S	251G	42.4	63.9	45.0	26.8	45.6	56.9
PVT-M	373G	41.9	63.1	44.3	25.0	44.9	57.6
Slide-PVT-M	338G	43.5	64.7	46.1	26.3	47.1	58.5
PVTV2-B3	379G	45.9	66.8	49.3	28.6	49.8	61.4
Slide-PVTv2-B3	343G	46.8	67.7	50.3	30.5	51.1	61.6

Table 2. Results on COCO object detection with RetinaNet [18]. The FLOPs are computed over backbone, FPN, and detection head with an input resolution of 1280×800.

Semantic Segmentation on ADE20K

Backbone	Method	FLOPs	#Params	mIoU	mAcc
PVT-T	S-FPN	158G	17M	36.57	46.72
Slide-PVT-T	S-FPN	136G	16M	**38.43**	50.05
PVT-S	S-FPN	225G	28M	41.95	53.02
Slide-PVT-S	S-FPN	188G	26M	**42.47**	54.00
Swin-T	UperNet	945G	60M	44.51	55.61
Slide-Swin-T	UperNet	946G	60M	**45.67**	57.13
Swin-S	UperNet	1038G	81M	47.64	58.78
Slide-Swin-S	UperNet	1038G	81M	**48.46**	60.18

Table 3. Results of semantic segmentation. The FLOPs are computed over encoders and decoders with an input image at the resolution of 512×2048. S-FPN is short for SemanticFPN [14] model.

For example, based on PVT, our model achieves even 0.5% higher performance, with 60% FLOPs. Our model based on PVTv2 and Swin Transformer also achieve comparable performance with 60%-70% FLOPs of competitive baselines. These results demonstrate that our module is applicable to various model structures and shows a better trade-off between computation cost and model performance.

5.2. ADE20K Semantic Segmentation

ADE20K [38] is a widely adopted benchmark for semantic segmentation with 20K training and 2K validation images. We employ our model on two representative segmentation models, SemanticFPN [14] and UperNet [34]. The comparison results show that our model can be adopted on various segmentation frameworks and effectively improve the model performance on dense prediction task.

5.3. COCO Object Detection

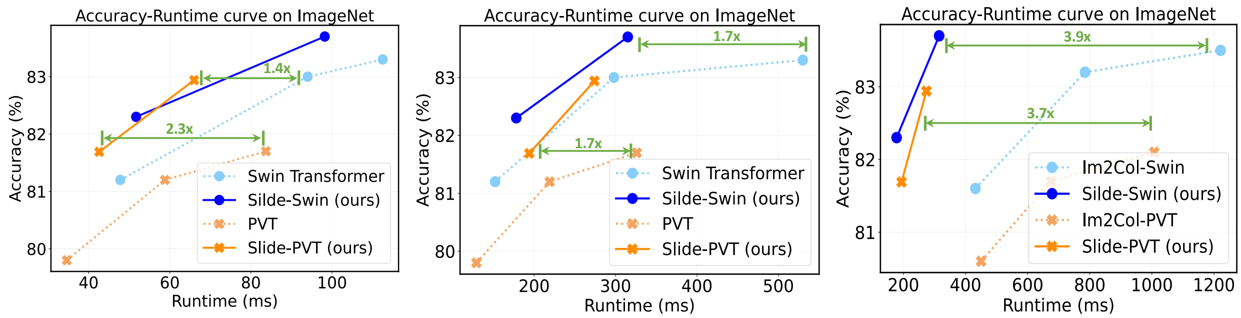
COCO [19] object detection and instance segmentation dataset has 118K training and 5K validation images. We use ImageNet pretrained model as the backbone in RetinaNet [18], Mask R-CNN [12] and Cascade Mask R-CNN [1] frameworks to evaluate the effectiveness of our method.

We conduct experiments on both 1x and 3x schedules with different detection heads and show results in Tab.1 and Tab.2. Our model shows better results under all settings. Also, our model achieves more significant improvements in detecting **small** objects (up to 4.5% improvement), which demonstrates the effectiveness of injecting local inductive bias towards Vision Transformer backbones.

5.4. Comparison with Other Local Attentions

To show a fair comparison with other local attention modules, we select two representative Vision Transformer models, Swin Transformer [20] and NAT [11], that are originally constructed based on window attention and local attention respectively. We adapt previous local attention approaches, including SASA [25], SAN [37], and NAT [11] into these two models, and compare performance with ours.

As shown in Tab.4, our model achieves significantly



(a) Metal Performance Shader (*v.s.* Baselines) (b) iPhone 12 (*v.s.* Baselines) (c) iPhone 12 (*v.s.* Im2Col)

Figure 6. Runtime comparison on Metal Performance Shader and iPhone 12 devices.

(a) Comparison on Swin-T Setting				
Local Attention	FLOPs	#Param	Acc.	FPS
SASA [25]	4.5G	29M	81.6	644
SAN [37]	4.5G	29M	81.4	670
NAT [11]	4.5G	29M	81.8	821
Ours	4.6G	30M	**82.3**	790
(b) Comparison on NAT-Mini Setting				
Local Attention	FLOPs	#Param	Acc.	FPS
SASA [25]	2.7G	20M	81.2	791
SAN [37]	2.7G	20M	81.1	815
NAT [11]	2.7G	20M	81.8	1045
Ours	2.7G	20M	**82.4**	998

Table 4. Comparison of different local attention modules on different model structures. We use Swin-Transformer and NAT as the basic settings. FPS is tested on a single RTX3090 GPU.

Stages w/ Slide Attention				FLOPs	#Param	Acc.	Diff.
Stage1	Stage2	Stage3	Stage4				
✓				4.5G	29M	81.8	-0.5
✓	✓			4.6G	29M	**82.3**	**Ours**
✓	✓	✓		4.6G	30M	82.2	-0.1
✓	✓	✓	✓	4.7G	30M	81.3	-1.0
Swin-T [20]				4.5G	29M	81.3	-1.0

Table 5. Ablation study on applying slide attention on different stages. All the models are based on the Swin-Tiny structure.

better results than Im2Col-based approach SASA. When comparing with CUDA-based approaches SAN and NAT, our model achieves higher performances (0.5%-1.3%) with comparable inference speed. This demonstrates the comprehensive superiority of our model on the accuracy-efficiency trade-off against other local attention approaches.

5.5. Inference Time

We further investigate the practical inference time of our method under different hardware, including computation units like Metal Performance Shader (MPS) and edge devices like iPhone 12. We show comparison results with two competitive baselines in Fig.6. We can see that our module shows significantly better trade-off between runtime and model performance on different devices, and achieves up to 2.3x speed up on advanced Vision Transformer models. For other local attention modules, due to the fact that CUDA-based approaches cannot be implemented on these devices, we only compare our method with Im2Col-based approach. As shown in Fig.6(c), our model achieves 3.7x-3.9x speed up while maintaining higher performances.

5.6. Ablation Study

To further validate the effectiveness of the designs in our model, we conduct several ablation studies. As shown in Tab.5, we can see that our slide attention module shows better performances when adopted at the early stages of Transformer models. Considering that our module has a similar design pattern with convolution, we believe this result is in accordance with the previous finding in [35], that convolutions are more useful at the early stages of Vision Transformer. We also show the effectiveness of each module in slide attention in Fig.2, which contributes to better model performance or efficiency respectively.

6. Conclusion

In this paper, we revisit the local attention mechanism and address its efficiency overhead by proposing a novel Slide Attention module with only common convolution operations. By substituting the inefficient Im2Col function with depthwise convolutions and equipped with a deformed shifting module, our module realizes local attention in high efficiency, flexibility, and generalizability. Extensive experiments demonstrated that our module can be widely adopted on a variety of Vision Transformers and different hardware devices while achieving a better trade-off between computation efficiency and model performance.

Acknowledgement

This work is supported in part by the National Key R&D Program of China (2019YFC1408703), the National Natural Science Foundation of China (62022048, 62276150), Guoqiang Institute of Tsinghua University and Beijing Academy of Artificial Intelligence. We also appreciate generous donation of computing resources by High-Flyer AI.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade R-Cnn: Delving into High Quality Object Detection. In *Conference on Computer Vision and Pattern Recognition*, 2018. 7

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In *European Conference on Computer Vision*, 2020. 1

[3] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable Convolutional Networks. In *International Conference on Computer Vision*, 2017. 6

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In *Conference on Computer Vision and Pattern Recognition*, 2009. 6

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2019. 1, 2

[6] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in Cnns. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2, 5

[7] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making Vgg-Style Convnets Great Again. In *Conference on Computer Vision and Pattern Recognition*, 2021. 6

[8] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin Transformer: A General Vision Transformer Backbone with Cross-Shaped Windows. In *Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 3, 6

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021. 1, 2

[10] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale Vision Transformers. In *International Conference on Computer Vision*, 2021. 2

[11] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood Attention Transformer. In *arXiv preprint arXiv:2204.07143*, 2022. 2, 3, 6, 7, 8

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-Cnn. In *International Conference on Computer Vision*, 2017. 7

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016. 2

[14] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic Feature Pyramid Networks. In *Conference on Computer Vision and Pattern Recognition*, 2019. 7

[15] Andrew Lavin and Scott Gray. Fast Algorithms for Convolutional Neural Networks. In *Conference on Computer Vision and Pattern Recognition*, 2016. 3, 5

[16] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring Plain Vision Transformer Backbones for Object Detection. In *European Conference on Computer Vision*, 2022. 1

[17] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. MViTv2: Improved Multiscale Vision Transformers for Classification and Detection. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *International Conference on Computer Vision*, 2017. 7

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, 2014. 6, 7

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *International Conference on Computer Vision*, 2021. 1, 2, 3, 6, 7, 8

[21] Xuran Pan, Chunjiang Ge, Rui Lu, Shiji Song, Guanfu Chen, Zeyi Huang, and Gao Huang. On the Integration of Self-Attention and Convolution. In *Conference on Computer Vision and Pattern Recognition*, 2022. 1

[22] Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3D Object Detection with Pointformer. In *Conference on Computer Vision and Pattern Recognition*, 2021. 1

[23] Xuran Pan, Tianzhu Ye, Dongchen Han, Shiji Song, and Gao Huang. Contrastive Language-Image Pre-Training with Knowledge Graphs. In *Advances in Neural Information Processing Systems*, 2022. 1

[24] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning*, 2021. 1

[25] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-Alone Self-Attention in Vision Models. In *Advances in Neural Information Processing Systems*, 2019. 2, 3, 7, 8

[26] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for Semantic Segmentation. In *International Conference on Computer Vision*, 2021. 1

[27] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training Data-Efficient Image Transformers & Distillation Through Attention. In *International Conference on Machine Learning*, 2021. 2

[28] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling Local Self-Attention for Parameter Efficient Visual Back-

bones. In *Conference on Computer Vision and Pattern Recognition*, 2021. 3

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. In *Advances in Neural Information Processing Systems*, 2017. 1, 2

[30] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions. In *International Conference on Computer Vision*, 2021. 1, 2, 3, 6

[31] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved Baselines with Pyramid Vision Transformer. In *Computational Visual Media*, 2022. 2, 6

[32] Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and Improving Relative Position Encoding for Vision Transformer. In *International Conference on Computer Vision*, 2021. 2

[33] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision Transformer with Deformable Attention. In *Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 3

[34] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified Perceptual Parsing for Scene Understanding. In *European Conference on Computer Vision*, 2018. 7

[35] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early Convolutions Help Transformers See Better. In *Advances in Neural Information Processing Systems*, 2021. 2, 6, 8

[36] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and Efficient Design for Semantic Segmentation with Transformers. In *Advances in Neural Information Processing Systems*, 2021. 1

[37] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring Self-attention for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2020. 2, 3, 7, 8

[38] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic Understanding of Scenes Through the ADE20k Dataset. In *International Journal of Computer Vision*. Springer, 2019. 6, 7

[39] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *International Conference on Learning Representations*, 2021. 1