

Low-Resolution Self-Attention for Semantic Segmentation

Yu-Huan Wu, Shi-Chen Zhang, Yun Liu, Le Zhang, Xin Zhan, Daquan Zhou, Jiashi Feng, Ming-Ming Cheng, and Liangli Zhen

Abstract—Semantic segmentation tasks naturally require high-resolution information for pixel-wise segmentation and global context information for class prediction. While existing vision transformers demonstrate promising performance, they often utilize high-resolution context modeling, resulting in a computational bottleneck. In this work, we challenge conventional wisdom and introduce the Low-Resolution Self-Attention (LRSA) mechanism to capture global context at a significantly reduced computational cost. Our approach involves computing self-attention in a fixed low-resolution space regardless of the input image’s resolution, with additional 3×3 depth-wise convolutions to capture fine details in the high-resolution space. We demonstrate the effectiveness of our LRSA approach by building the LRFormer, a vision transformer with an encoder-decoder structure. Extensive experiments on the ADE20K, COCO-Stuff, and Cityscapes datasets demonstrate that LRFormer outperforms state-of-the-art models. The code will be made publicly available.

Index Terms—Low-Resolution Self-Attention, Semantic Segmentation, Vision Transformer

arXiv:2310.05026v1 [cs.CV] 8 Oct 2023

1 INTRODUCTION

As a fundamental computer vision problem, semantic segmentation [1]–[3] aims to assign a semantic label to each image pixel. Semantic segmentation models [4], [5] usually rely on pretrained backbone networks [6], [7] for feature extraction, which is then followed by specific designs for pixel-wise predictions. In the last decade, the progress in feature extraction via various backbone networks has consistently pushed forward state-of-the-art semantic segmentation [8]–[10]. This paper improves the feature extraction for semantic segmentation from a distinct perspective.

It is commonly believed that semantic segmentation, as a dense prediction task, requires high-resolution features to ensure accuracy. In contrast, image classification typically infers predictions from a very small feature map, such as $1/32$ of the input resolution. Semantic segmentation models with convolutional neural networks (CNNs) usually decrease the strides of backbone networks to increase the feature resolution [11]–[14], *e.g.*, $1/8$ of the input resolution. This attribute is also well preserved in transformer-based semantic segmentation, demonstrating that high-resolution is still necessary for semantic segmentation.

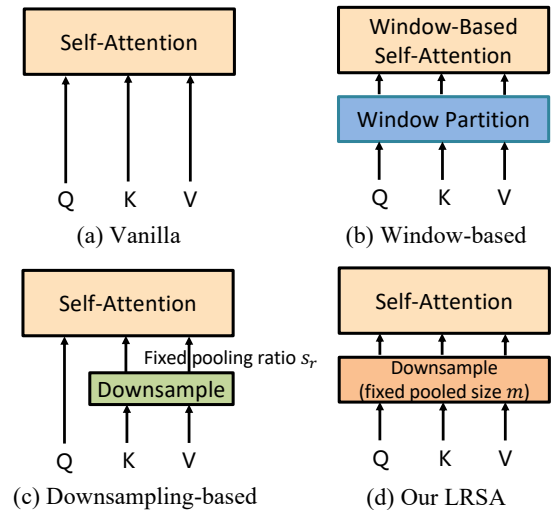


Fig. 1. Comparison of existing and our proposed paradigms for the self-attention calculation in the vision transformer. Representatives include (a) ViT [15], DeiT [16]; (b) Swin [17], CSwin [18]; (c) PVT [19], SegFormer [9], P2T [20]; and (d) our LRFormer.

High-resolution features are powerful for capturing the local details, while context information pertains to the broader understanding of the scene. Contextual features discern the interrelations between various scene components [21], mitigating the ambiguity inherent in local features. Thus, considerable research efforts [1], [22] have been devoted to extending the receptive field of CNNs. Conversely, vision transformers inherently facilitate the computation of global relationships by introducing self-attention with a global receptive field. Nonetheless, this comes at a significant computational cost, as vanilla attention mechanisms exhibit quadratic complexity to input length. Intriguingly, seminal studies [9], [20], [23] made a remarkable effort by judiciously downsampling some of the features (*i.e.*, key and value) during the self-attention computation for reduced computational complexities.

- Y.-H. Wu was with Nankai University when he started this research. Y.-H. Wu and L. Zhen are currently with the Institute of High Performance Computing (IHPC), A*STAR, Singapore. (E-mail: {wu_yuhuan,zhen_liangli}@ihpc.a-star.edu.sg)
- S.-C. Zhang and M.-M. Cheng are with VCIP, Nankai University, Tianjin, China. (E-mail: zhangshichen@mail.nankai.edu.cn and cmm@nankai.edu.cn)
- Y. Liu is with the Institute of Infocomm Research (I2R), A*STAR, Singapore. (E-mail: liu_yun@i2r.a-star.edu.sg)
- L. Zhang is with the University of Electronic Science and Technology of China. (E-mail: zhangleuestic@gmail.com)
- X. Zhan is with Udeer AI, Hangzhou, China. (E-mail: zhanxin@udeer.ai)
- D. Zhou and J. Feng are with Bytedance Inc., Singapore. (E-mail: zhoudaqun21@gmail.com, jshfeng@bytedance.com)
- M.-M. Cheng is the corresponding author.

Nevertheless, we observe that the computational overhead of self-attention remains a non-negligible bottleneck for existing vision transformers, as evidenced by Tab. 11. Consequently, we aim to delve deeper into the downsampling in the core component of the transformer, *i.e.*, self-attention. Diverging from prior works that only downsample the key and value features [9], [20], [23], we propose to downsample all constituents—query, key, and value features. In this way, the output of self-attention would be in a low-resolution so that the mainstream of the transformer would contain low-resolution. Furthermore, we adopt a fixed downsampling size rather than a downsampling ratio to attain a very low computational complexity for self-attention. The proposed method is called **Low-Resolution Self-Attention (LRSA)**.

Fig. 1 depicts the differences between existing self-attention approaches and our LRSA. Vanilla self-attention [15] (Fig. 1(a)) directly computes the global feature relations in the original resolution, which is quite expensive. Window-based methods [17], [18], [24], [25] (Fig. 1(b)) divide the features into small windows and perform local self-attention within each window. Downsampling-based methods [9], [19], [20], [26], [27] (Fig. 1(c)) keep the size of the query unchanged, and they downsample the key and value features with a fixed pooling ratio. The lengths of key and value features increase linearly with the input resolution. In contrast, our LRSA (Fig. 1(d)) downsamples all query, key, and value to a small fixed size, leading to very low complexity regardless of the input resolution. More analysis of the computational complexity can refer to §3.1.

While LRSA significantly boosts efficiency in capturing global context, we recognize that maintaining fine-grained details is another critical aspect for optimal performance in semantic segmentation. To address this duality, we employ LRSA to capture global context information in a purely low-resolution domain, while simultaneously integrating small kernel (3×3) depth-wise convolution to capture local details in the high-resolution space. Based on these foundational principles, we build a new backbone network for feature extraction and a simple decoder to aggregate the extracted multi-level features for semantic segmentation. This new model is dubbed as **Low-Resolution Transformer (LRFormer)**. We evaluate LRFormer on popular benchmarks, including ADE20K [28], COCO-Stuff [29], and Cityscapes [30]. Experimental results (*e.g.*, Fig. 2) demonstrate the superiority of LRFormer over state-of-the-art models. Besides, LRFormer also achieves competitive performance for image classification on the ImageNet dataset [31], compared with recent strong baselines.

2 RELATED WORK

2.1 Semantic Segmentation

Semantic segmentation is a fundamental task in computer vision. It is challenging due to the numerous variations like object sizes, textures, and lighting conditions in practical scenarios. FCN [32], the pioneering work in this area, proposed the adaptation of CNNs for semantic segmentation in an end-to-end manner. Since then, numerous studies have been built upon FCN [32], with major efforts focused on enriching multi-scale representations [4], [5], [33], enhancing

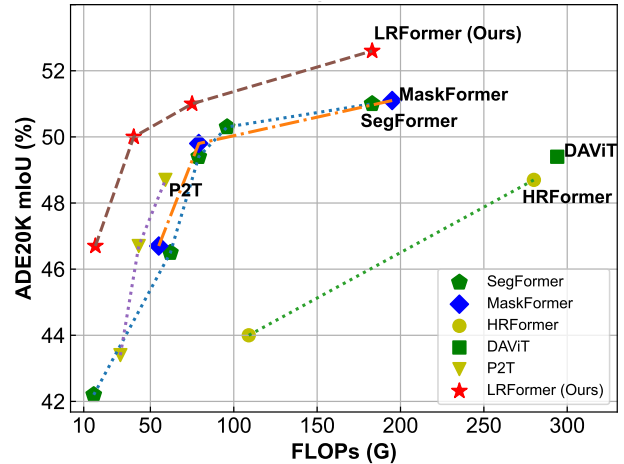


Fig. 2. Experimental comparisons on ADE20K [28] dataset.

boundary perception [34]–[37], contextual representations [21], [38] and introducing visual attention [2], [3], [8], [11], [14]. These studies deeply explored the semantic head design upon FCN [32] and achieved great progress. Among these, many approaches [1]–[5], [8], [11]–[14] are greatly benefited from the high-resolution features, performing prediction in the 1/8 of the input resolution to ensure high accuracy.

More recently, many works [9], [10], [39]–[41] showed that vision transformers [15] can largely improve the performance of semantic segmentation. This is mainly attributed to the strong global capability of vision transformers, which happens to be a crucial property required for semantic segmentation. For example, SETR [39] first adapted ViT as an encoder followed by multi-level feature aggregation. SegFormer [9] introduced a novel pyramid vision transformer encoder with an MLP mask decoder. MaskFormer [40] revolutionized mask decoders with transformer-based mask classification. More discussions on vision transformers can refer to §2.3.

2.2 Convolutional Neural Networks

Given that CNN-based semantic segmentation models rely on CNN backbones for feature extraction, we discuss some notable CNN architectures. Since the emergence of AlexNet [42], many techniques have been developed to strengthen the CNN representations and achieved great success. For example, VGG [43], GoogleNet [44], ResNets [6] and DenseNets [45] developed increasingly deep CNNs to learn more powerful representations. ResNeXts [7], Res2Nets [46], and ResNeSts [47] explored the cardinal design in ResNets [6]. SENet [48] and SKNet [49] introduced different attention architectures for selective feature learning. Very recently, CNNs with large kernels are proven powerful in some works [50]–[52]. To ensure the high-resolution of feature maps for accurate semantic segmentation, semantic segmentation models usually decrease the strides of these CNNs and uses the dilated convolutions [22] to keep larger receptive field. Motivated by this, HRNet [53] was proposed to directly learn high-resolution CNN features. Despite the numerous successful stories, CNNs are limited in capturing global and long-range relationships, which are of vital importance for semantic segmentation.

2.3 Vision Transformers

Transformers are initially proposed in natural language processing (NLP) [54]. Through multi-head self-attention (MHSA), transformers are capable of modeling global relationships. Thanks to this characteristic, transformers may also be powerful for computer vision tasks that require global information for a whole understanding of the visual scenarios. To bridge this gap, ViT [15] transformed an image to tokens via a 16×16 pooling operation and adopts the transformer to process these tokens, achieving better performance than CNNs in image recognition. After that, vision transformers are developed rapidly by leveraging knowledge distillation [55], overlapping patch embedding [56] or convolutions [57], [58]. Recently, pyramid vision transformers [10], [17], [19], [20], [23], [26], [59], [60] are proven to be powerful for image recognition tasks like semantic segmentation. For example, PVT [19] and MViT [26] proposed to build a pyramid vision transformer pipeline via performing downsampling on key and value features. Liu *et al.* [17] created a window-based vision transformer with shifted windows. Yuan *et al.* [10] presented HRFormer to learn high-resolution features for dense prediction using the vision transformer. Xia *et al.* [61] proposed DAT with deformable attention, conducting deformable sampling on key and value features.

Despite their reported effectiveness, it is still commonly believed that high-resolution features are crucial for self-attention to effectively capture contextual information in semantic segmentation. Window-based vision transformers [17], [18], [24], [25] calculate self-attention within each local windows to reduce the computational complexity so that they can keep the high-resolution of feature maps. Downsampling-based vision transformers [9], [19], [20], [23], [26], [27] keep the size of the query while partially conduct the downsampling on the key and value features with a fixed pooling ratio. Such strategy greatly reduces the complexity compared with vanilla attention so as to keep high-resolution features, making themselves computationally non-negligible especially for high-resolution inputs (Tab. 11). In contrast, we question the necessity of keeping high-resolution for capturing context information via self-attention. We study this question by proposing LRFormer with LRSA. The good performance on several public benchmarks suggest the superiority of our LRFormer for semantic segmentation.

3 METHODOLOGY

In this section, we first introduce the Low-Resolution Self-Attention (LRSA) mechanism in §3.1. Then, we build Low-Resolution Transformer (LRFormer) using LRSA for semantic segmentation in §3.2. The decoder of LRFormer is presented in §3.3. Finally, we provide the implementation details in §3.4.

3.1 Low-Resolution Self-Attention

Unlike existing vision transformers that aim to maintain high-resolution feature maps during self-attention, our proposed LRSA computes self-attention in a low-resolution space, significantly reducing computational costs. Before

Scheme	Global	Spatial Corr.	Complexity
Window-based [17]	✗	✓	$O(NC^2)$
Factorized [62]	✓	✗	$O(NC^2)$
Downsampling-based [9]	✓	✓	$O(N^2C + NC^2)$
LRSA (Ours)	✓	✓	$O(NC + C^2)$

TABLE 1

Comparison of various self-attention schemes. N is the length of the flattened features and C is the number of feature channels. ‘‘Spatial Corr.’’ denotes the spatial correlation. We omit constant factors for simplicity, like the window size in window-based methods and the downsampled size of LRSA.

delving into our proposed LRSA, let us first revisit the vision transformer architecture.

Revisiting self-attention in transformers. The vision transformer [15] has been demonstrated to be very powerful for computer vision [10], [17]–[20], [23]–[26]. It consists of two main parts: the multi-head self-attention (MHSA) and the feed-forward network (FFN). We continue by elaborating on MHSA. Given the input feature F_{in} , the query Q , key K and value V are obtained with a linear transformation from F_{in} . Then, we can calculate MHSA as

$$\text{Attention}(F_{in}) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (1)$$

where d_k is the number of channels of F_{in} . We omit the multi-head operation for simplicity. The overall computational cost of vanilla self-attention is $O(N^2C + C^2N)$, where N and C are the number of tokens and the number of channels of $F_{in} \in \mathbb{R}^{N \times C}$, respectively. As the number of tokens of natural images are usually very large, the computational cost of vanilla self-attention is very high.

Previous solutions. To alleviate the computational cost while keeping the high-resolution of feature maps, downsampling-based vision transformers [9], [19], [20], [23], [26], [27] change the self-attention computation to

$$\text{Attention}(F_{in}) = \text{Softmax}\left(\frac{QK_s^T}{\sqrt{d_k}}\right)V_s, \quad (2)$$

in which K_s and V_s are the downsampled key K and value V with a fixed downsampling ratio s_r , respectively. The $1D \leftrightarrow 2D$ feature reshaping is omitted for convenience. The length of K_s and V_s is $1/s_r^2$ of the original K and V . If the original length of K and V is too large, the K_s and V_s will also be long sequences, introducing considerable computational cost in self-attention. Here, we only introduce downsampling-based transformers because they are most relevant to our method.

Our solution. Instead, we tackle the heavy computation of vanilla self-attention from a new perspective: we do not keep the high-resolution of feature maps but process the features in a very low-resolution space. Specifically, the proposed LRSA downsamples the input feature F_{in} to a fixed size m . Then, multi-head self-attention is applied:

$$\text{Attention}(F_{in}) = \text{Softmax}\left(\frac{Q_p K_p^T}{\sqrt{d_k}}\right)V_p, \quad (3)$$

where Q_p , K_p and V_p are obtained by a linear transformation from the downsampled F_{in} . Q_p , K_p and V_p are with a fixed size m , regardless of the resolution of the input F_{in} . Compared with vanilla self-attention and previous

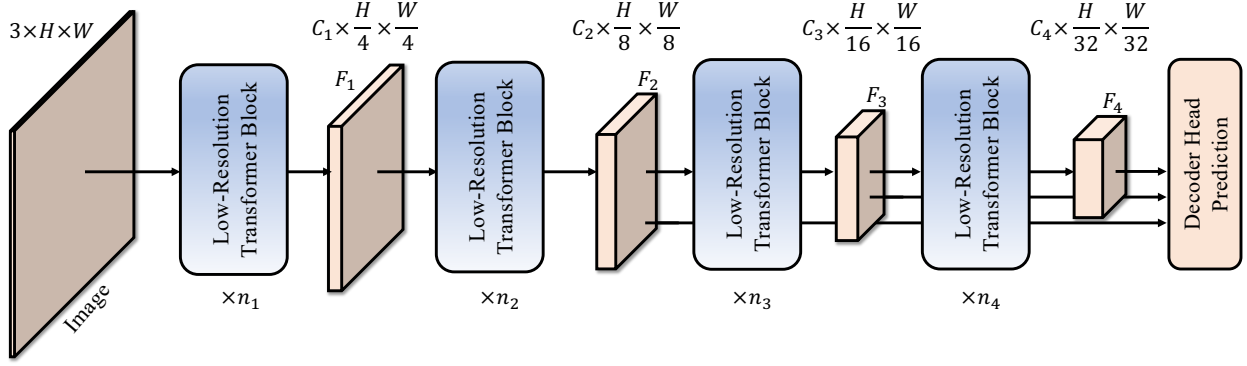


Fig. 3. Pipeline of the proposed LRFormer. F_2 , F_3 and F_4 are fed into the decoder head for semantic segmentation.

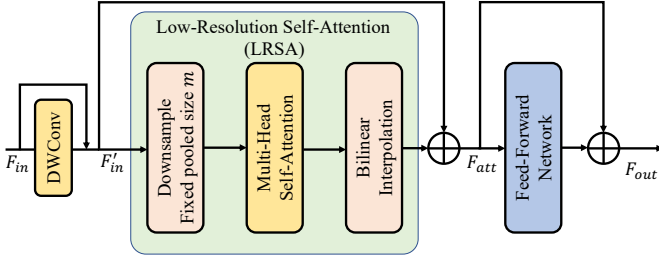


Fig. 4. Illustration of a basic block of our LRFormer. The symbol “ \oplus ” denotes the element-wise addition. We add a 3×3 depth-wise convolution (DWConv) with a residual connection before LRSA, which is also applied between the two linear layers of the FFN.

solutions, our LRSA has a much lower computational cost. LRSA can also facilitate attention optimization due to the much shorter token length. To fit the size of the original F_{in} , we then perform a bilinear interpolation after the self-attention calculation.

Complexity and characteristics. The computational complexity of LRSA is much lower than existing self-attention mechanisms for vision transformers. We summarize the main characteristics and computational complexity of recent popular self-attention mechanisms and our LRSA in Tab. 1. Spatial correlation means that self-attention is carried out in the spatial dimension, and some factorized transformers [62] compute self-attention in the channel dimension for reducing complexity. As can be observed from Tab. 1, other methods often face trade-offs among complexity, global receptive field, and spatial correlation. In contrast, our LRSA offers advantages in all these aspects.

Let us continue by analyzing the computational complexity of LRSA. For convenience, we do not include the $1D \leftrightarrow 2D$ feature reshaping. LRSA first downsamples the input features $F_{in} \in \mathbb{R}^{N \times C}$ to a fixed size $m \times C$ with a 2D pooling operation, whose computational cost is $O(NC)$. Then, LRSA performs linear transformations and self-attention on the pooled features, which costs $O(mC^2)$. The computation of self-attention costs $O(m^2C)$. The final upsampling operation has the same computational cost as downsampling. Overall, the computational complexity of LRSA is $O(NC + mC^2 + m^2C)$. As m is a constant number (e.g., 16^2) regardless of the value of N , we can simplify the complexity of LRSA to $O(NC + C^2)$, which is much smaller

than existing methods.

3.2 Low-Resolution Transformer

In this part, we build the LRFormer for semantic segmentation by incorporating the proposed LRSA. The overall architecture of LRFormer is illustrated in Fig. 3, with an encoder-decoder architecture.

Encoder-decoder. Taking a natural image as input, the encoder first downsamples it by a factor of $1/4$, following prevailing literature in this field [17], [19], [20], [23], [26]. The encoder consists of four stages with a pyramid structure, each comprising multiple stacked basic blocks. In between every two stages, we include a patch embedding operation to reduce the feature size by half. This results in the extraction of multi-level features F_1, F_2, F_3, F_4 with strides of 4, 8, 16, and 32, respectively. We resize F_2, F_3, F_4 to the same size as F_2 before concatenating and squeezing them to smaller channels. The resulting features are then fed into our decoder head, which performs further semantic reasoning and outputs the final segmentation map via a 1×1 convolution layer. The details of our decoder head are presented in §3.3.

Basic block. The basic block is illustrated in Fig. 4. Like previous transformer blocks [17], [19], the basic block of our LRFormer is composed of a self-attention module and an FFN. The FFN is generally an MLP layer composed of two linear layers with GELU [63] activation in between. Differently, we renovate the self-attention module with our proposed LRSA. As LRSA is computed in a very low-resolution space, attaining a low complexity regardless of the input resolution. However, the low-resolution space may lose the spatial locality of the input features. Inspired by recent works [20], [58], we further introduce depth-wise convolution (DWConv) in both positional encoding and FFN, assisting the feature extraction via capturing spatial local details. That is, we insert a 3×3 DWConv layer with short connection followed by our LRSA, providing conditional positional encoding [58]. This strategy is also applied between the two linear layers of the FFN. Therefore, our basic block can be simply formulated as below:

$$\begin{aligned} F'_{in} &= F_{in} + \text{DWConv}(F_{in}), \\ F_{att} &= F'_{in} + \text{LRSA}(\text{LayerNorm}(F'_{in})), \\ F_{out} &= F_{att} + \text{FFN}(\text{LayerNorm}(F_{att})), \end{aligned} \quad (4)$$

Stage	Output Size	LRFormer-T	LRFormer-S	LRFormer-B	LRFormer-L
1	$F_1 : \frac{H}{4} \times \frac{W}{4}$	$C = 48, E = 8$ $C_h = 24, n_1 = 2$	$C = 64, E = 8$ $C_h = 32, n_1 = 3$	$C = 80, E = 8$ $C_h = 40, n_1 = 4$	$C = 96, E = 8$ $C_h = 48, n_1 = 4$
2	$F_2 : \frac{H}{8} \times \frac{W}{8}$	$C = 96, E = 8$ $C_h = 24, n_2 = 2$	$C = 128, E = 8$ $C_h = 32, n_2 = 3$	$C = 160, E = 8$ $C_h = 40, n_2 = 4$	$C = 192, E = 8$ $C_h = 48, n_2 = 6$
3	$F_3 : \frac{H}{16} \times \frac{W}{16}$	$C = 240, E = 4$ $C_h = 24, n_3 = 6$	$C = 320, E = 4$ $C_h = 32, n_3 = 12$	$C = 400, E = 4$ $C_h = 32, n_3 = 15$	$C = 480, E = 4$ $C_h = 32, n_3 = 18$
4	$F_4 : \frac{H}{32} \times \frac{W}{32}$	$C = 384, E = 4$ $C_h = 24, n_4 = 3$	$C = 512, E = 4$ $C_h = 32, n_4 = 3$	$C = 512, E = 4$ $C_h = 40, n_4 = 8$	$C = 640, E = 4$ $C_h = 48, n_4 = 8$

TABLE 2

Detailed settings of the encoders for different LRFormer variants. C , C_h , E , and n_i denote the number of feature channels, channels of each attention head, expansion ratio of FFN, and the number of basic blocks for the i -th stage, respectively.

where F_{in} , F_{att} and F_{out} represent the input, output of LRSA, and output of the basic block, respectively.

Architecture setting. To fit the budgets of different computational resources, we design four variants of LRFormer, namely LRFormer-T/S/B/L, stacking different numbers of basic blocks for each stage in the encoder. We summarize the detailed settings of their encoders in Tab. 2. In terms of ImageNet pretraining [31], the computational cost of LRFormer-T/S/B/L is similar to ResNet-18 [6] and Swin-T/S/B [17], respectively.

3.3 Decoder Head

In semantic segmentation, it is suboptimal to predict the results based solely on the final output of the encoder, as multi-level information is useful in perceiving objects with various scales and aspect ratios [9], [64]. Thus, we design a simple decoder for LRFormer to aggregate multi-level features efficiently and effectively. To this end, we note that an MLP aggregation can achieve good performance in the state-of-the-art work SegFormer [9]. However, it does not consider the spatial correlation between the features from different levels. Therefore, we encapsulate our LRSA into our decoder for feature refinement, strengthening the semantic reasoning of LRFormer.

As mentioned above, F_2, F_3, F_4 are resized to the same size as F_1 and then concatenated together. We apply a 1×1 convolution on the concatenated feature to squeeze the number of channels. Then, a basic block (LRSA + FFN) is adopted to refine the squeezed feature. As we know, the feature from the top of the encoder, *i.e.*, F_4 , could be the most semantic meaningful. To avoid the loss of semantic information in the aggregation of high-level (F_4) and low-level (F_2, F_3) features, we concatenate the refined feature with F_4 to enhance the semantics. After that, another basic block is connected for further feature refinement. Finally, we infer the segmentation prediction from the refined feature with a simple 1×1 convolution. The experiments demonstrate that our simple decoder with LRSA can do better than previous state-of-the-art decoder heads for semantic segmentation, as shown in Tab. 9.

3.4 Implementation Details

In LRFormer, We apply the overlapped patch embedding, *i.e.*, a 3×3 convolution with a stride of 2, to downsample the features by half between each stage. To strengthen multi-scale learning of LRSA with negligible cost, we use pyramid

pooling [20] to extract multi-scale features when computing the key and value features in LRSA. The desired fixed downsampling size m for generating the query, key and value is 16^2 for semantic segmentation. Such size is changed to 7^2 for ImageNet pretraining because $m = 16^2$ is too large for image classification. For the number of channels in the decoder, we set it to 256/384/512/640 for LRFormer-T/S/B/L, respectively.

4 EXPERIMENTS

4.1 Experimental Setup

Datasets. We perform experiments on three well-established datasets. ADE20K [28] is a very challenging scene parsing dataset that contains 150 semantic classes with diverse foreground and background, consisting of 20K, 2K, and 3.3K images for training, validation, and testing, respectively. COCO-Stuff [29] labels both things and stuffs with a total of 171 fine-grained semantic labels, with 164K, 5K, 20K, and 20K images for training, validation, test-dev, and test challenge. Cityscapes [30] is a high-quality dataset for street scene parsing that contains 3K, 0.5K, and 1.5K driving images for training, validation, and testing. These datasets cover a wide range of semantic categories and pose different challenges for semantic segmentation models.

ImageNet pretraining. We adopt the popular *timm* package to implement our network. Following other networks, we first pretrain the backbone encoder of LRFormer on the ImageNet-1K dataset, which has 1.3M training and 50K validation images with 1K object categories. During ImageNet pretraining, the decoder head of LRFormer is omitted. To regularize the training process, we follow the standard data augmentation techniques and optimization strategy used in previous works [9], [17], [55]. We use AdamW [65] as the default optimizer with a learning rate of 0.001, weight decay of 0.05, a *cosine* learning rate adjustment schedule, and a batch size of 1024. No model EMA is applied. The backbone encoder is pretrained for 300 epochs, and we apply layer scale [66] to alleviate the overfitting of large networks, as suggested by recent works [50], [66]. For LRFormer-L, we follow [17], [50] additionally pretrain the network on the full ImageNet-22K dataset for 90 epochs and then finetune it on ImageNet-1K dataset for 30 epochs. In the finetuning, the learning rate is set as $5e-5$, and each mini-batch has 512 images.

Method	FLOPs ↓	#Params ↓	mIoU ↑
SegFormer [9]	16G	14M	42.2%
HRFormer-S [10]	109G	14M	44.0%
LRFormer-T	17G	13M	46.7%
Mask2Former [64]	74G	47M	47.7%
SegFormer-B2 [9]	62G	28M	46.5%
LRFormer-S	40G	32M	50.0%
HRFormer-B [10]	280G	56M	48.7%
SegFormer-B3 [9]	96G	47M	49.4%
LRFormer-B	75G	69M	51.0%
DPT-Hybrid [67]	308G	124M	49.0%
DAViT-B [68]	294G	121M	49.4%
MaskFormer [40]	195G	102M	51.3%
SegFormer-B5 [9]	183G	85M	51.0%
LRFormer-L	183G	113M	52.6%
SETR-MLA [†] [39]	-	302M	48.6%
MaskFormer [†] [64]	195G	102M	53.1%
CSWin-B [†] [18]	463G	109M	51.8%
LRFormer-L [†]	183G	113M	54.2%

TABLE 3

Comparisons with recent transformer-based methods on the ADE20K dataset [28]. The results of our method are marked as **bold**. “†” indicates the result pretrained on ImageNet-22K.

Training for semantic segmentation. We use *mmsegmentation* framework to train our network for semantic segmentation. AdamW [65] is adopted as the default optimizer, with learning of 0.00006, weight decay of 0.01, and *poly* learning rate schedule with factor 1.0. Following [9], [17], the weight decay of LayerNorm [69] layers is set as 0. Regarding the data augmentation, we use the same strategy as mentioned in [9], [17]. That is we construct the pipeline of image resizing (0.5 ~ 2×), random horizontal flipping, followed by a random cropping of size 512×512, 512×512, and 1024×1024 for ADE20K, COCO-Stuff, and Cityscapes datasets, respectively. Note that for our largest model LRFormer-L in ADE20K, the cropped size remains 640×640, consistent with recent works. The mini-batch size is set to 16, 16, and 8 images for ADE20K, COCO-Stuff, and Cityscapes datasets, respectively. We train our network for 160K, 80K, and 160K iterations for ADE20K, COCO-Stuff, and Cityscapes datasets, respectively. We only use the cross-entropy loss for training and do not employ any extra losses like the auxiliary loss [4] and OHem [70].

Testing for semantic segmentation. During testing, we maintain the original aspect ratio of the input image and resize it to a shorter size of 512 and a longer size not exceeding 2048 for the ADE20K and COCO-Stuff datasets. We follow the suggestion of [9] and resize the input size of LRFormer-L for the ADE20K dataset to a shorter size of 640 and a longer size not exceeding 2560. In the Cityscapes dataset, we apply a crop size of 1024×1024 with sliding window testing strategy following [9].

4.2 Comparisons

ADE20K. Results are shown in Tab. 3. LRFormer is compared with several recent transformer-based methods in different complexity levels. We can observe that our LRFormer exhibits strong superiority over other methods. For example, LRFormer-T/S/B/L are 4.5%/3.5%/2.6%/1.6% better than SegFormer-B1/B2/B4/B5 [9], [17]. LRFormer-T is 2.3%

Method	FLOPs ↓	#Params ↓	mIoU ↑
HRFormer-S [10]	109G	14M	37.9%
SegFormer-B1 [9]	16G	14M	40.2%
LRFormer-T	17G	13M	43.9%
SegFormer-B2 [9]	62G	28M	44.6%
LRFormer-S	40G	32M	46.4%
HRFormer-B [10]	280G	56M	42.4%
SegFormer-B3 [9]	79G	47M	45.5%
SegFormer-B5 [9]	112G	85M	46.7%
LRFormer-B	75G	69M	47.2%
LRFormer-L	122G	113M	47.9%

TABLE 4

Comparisons with recent transformer-based methods on the full COCO-Stuff dataset [29]. Results of our method are marked as **bold**.

Method	FLOPs ↓	#Params ↓	mIoU ↑
HRFormer-S [10]	872G	14M	80.0%
SegFormer-B1 [9]	244G	14M	78.5%
LRFormer-T	122G	13M	80.7%
SegFormer-B2 [9]	717G	28M	81.0%
LRFormer-S	295G	32M	81.9%
HRFormer-B [10]	2240G	56M	81.9%
SegFormer-B3 [9]	963G	47M	81.7%
LRFormer-B	555G	67M	83.0%
SegFormer-B5 [9]	1460G	85M	82.4%
LRFormer-L	908G	111M	83.2%

TABLE 5

Comparisons with recent transformer-based methods on the Cityscapes dataset [30]. Results of our method are marked as **bold**. FLOPs are calculated for an input size of 1024 × 2048.

better than Swin-T-based Mask2Former [64] with near half FLOPs. With ImageNet-22K pretraining, LRFormer is 1.1% and 2.4% better than the strongest Swin-B-based MaskFormer [17], [40] and UperNet-based CSwin [18], [71] with fewer FLOPs.

COCO-Stuff. We elaborate the results in Tab. 4. We evaluated our method on different network scales and compared it against recent popular methods. LRFormer achieved the highest mIoU on all network scales, outperforming the other methods. Specifically, our LRFormer-T model achieved a mIoU of 43.9%, which is 3.7% higher than HRFormer-S and 3.7% higher than SegFormer-B1. Similarly, our LRFormer-S and LRFormer-B models outperformed the corresponding SegFormer models by 1.8% and 1.7%. Our LRFormer-L model outperforms SegFormer-B5 by 1.2%. These experimental comparisons demonstrate the superiority of LRFormer on the COCO-Stuff dataset.

Cityscapes. Tab. 5 presents the experimental comparisons between LRFormer and recent popular methods on the Cityscapes dataset. LRFormer outperforms SegFormer and HRFormer in all cases. We can observe that due to large input size, FLOPs of other methods are much higher than ours. For example, SegFormer-B2 costs 717G FLOPs while our LRFormer-S only spends 41% FLOPs with 0.9% improvement. More complexity analysis can refer to Tab. 11.

ImageNet. Since we pretrained our backbone encoder on ImageNet, we also evaluate our network on ImageNet classification for reference. Results are shown in Tab. 6. We divide them to five groups. The first four groups are

Model	FLOPs ↓	#Params ↓	Size	Top-1 Acc. ↑
ResNet-18 [6]	1.8G	12M	224 ²	68.5%
PVTv2-B1 [23]	2.1G	13M	224 ²	78.7%
P2T-Tiny [20]	1.8G	12M	224 ²	79.8%
LRFormer-T	1.8G	13M	224 ²	80.8%
ResNet-50 [6]	4.1G	26M	224 ²	78.5%
Swin-T [17]	4.5G	28M	224 ²	81.5%
ConvNeXt-T [50]	4.5G	29M	224 ²	82.1%
MViTv2-T [27]	4.7G	24M	224 ²	82.3%
CSwin-T [18]	4.3G	23M	224 ²	82.7%
LRFormer-S	4.7G	30M	224 ²	83.5%
Swin-S [17]	8.7G	50M	224 ²	83.0%
ConvNeXt-S [50]	8.7G	50M	224 ²	83.1%
DAT-S [61]	9.0G	50M	224 ²	83.7%
P2T-Large [20]	9.8G	55M	224 ²	83.9%
LRFormer-B	9.3G	62M	224 ²	84.5%
DeiT-B [16]	17.5G	86M	224 ²	81.8%
RegNetY-16G [72]	16.0G	84M	224 ²	82.9%
RepLKNet-31B [51]	15.3G	79M	224 ²	83.5%
SwinT-B [17]	15.4G	88M	224 ²	83.5%
ConvNeXt-B [50]	15.4G	89M	224 ²	83.8%
FocalNet-B [73]	15.4G	89M	224 ²	83.9%
DAT-B [61]	15.8G	88M	224 ²	84.0%
CSwin-B [18]	15.0G	78M	224 ²	84.2%
LRFormer-L	15.7G	101M	224 ²	85.0%
Swin-B [†] [17]	15.4G	88M	224 ²	85.2%
ConvNeXt-B [†] [50]	15.4G	89M	224 ²	85.8%
LRFormer-L [†]	15.7G	101M	224 ²	86.4%
ConvNeXt-B [†] [50]	45.1G	89M	384 ²	86.8%
Swin-B [†] [17]	47.0G	88M	384 ²	86.4%
LRFormer-L [†]	46.3G	101M	384 ²	87.2%

TABLE 6

Classification results on ImageNet-1K [31] dataset. Results of our method are marked as **bold**. Results marked with “[†]” are pretrained on ImageNet-22K dataset.

Method	Memory	Top-1 Acc. ↑	mIoU ↑
LRFormer-S	14.5GB	81.6%	48.5%
w/o DWConv (bef. LRSA)	13.8GB	81.4%	48.0%
w/o DWConv (FFN)	11.7GB	81.1%	47.1%

TABLE 7

Ablation study on the spatial locality capturing. “Memory” is the training memory in ImageNet pretraining.

divided by the FLOPs of approximate 2G, 4.5G, 9G, 16G, respectively. The last group includes the results pretrained on ImageNet-22K dataset. The backbone encoder of our LRFormer outperformed recent state-of-the-art CNN-based methods such as ConvNeXt [50] and RepLKNet [51], and transformer-based methods like CSwin [18] and P2T [20].

4.3 Visualization analysis.

To visually illustrate the effectiveness of our method, we pick segformer [9] as the model for intuitive comparison from ADE20K val set and Cityscapes val set, as shown in Fig. 5 and Fig. 6 respectively. The results indicate that LRFormer is capable of generating more precise segmentation maps, particularly in the areas highlighted by the red boxes. We discover that LRFormer offers significant advantages in terms of maintaining object segmentation integrity and capturing intricate details.



Fig. 5. **Qualitative Visualization on ADE20K val set.** The figures from left to right are input images, ground truth, segmentation maps of SegFormer [9], segmentation maps of our LRFormer. Significant improvements are indicated by red boxes on segmentation maps.

4.4 Ablation Study

In the following part, we conduct several ablation studies to analyze our LRFormer. Except for specifically mentioning, we use the following settings. LRFormer-S is set as the baseline and trained using 8 GPUs for both classification and semantic segmentation. For classification, our network is trained for 100 epochs in the ImageNet-1K [31] dataset. For semantic segmentation, our network is trained for 80K iterations in the ADE20K [28] dataset. Other settings are kept same as the setup in §4.1.

Locality capturing. Our LRSA only computes the attention in low-resolution space. Introducing spatial locality, 3×3 depth-wise convolution, to our network is beneficial for getting fine-grained semantic maps. In Tab. 7, we analyzed the effect of the two depth-wise convolution before LRSA and in FFN. We can observe that the ADE20K performance of our LRFormer is improved by 0.5% and 1.4% and when adding the depth-wise convolution before LRSA in FFN, with 5% and 24% training memory overhead. Therefore, we add both of them in our LRFormer.

Fixed pooled size. We reported the results in Tab. 8. For each basic block, the pooling operation will be omitted if the feature map size is smaller than the desired pooled size. Default fixed pooled size m is 16^2 for semantic segmentation. Results show larger pooled size ($m \geq 16^2$) achieves saturated performance. The default setting only introduces 5% training memory overhead and FLOPs compared with the pooled size of 8^2 for semantic segmentation. When

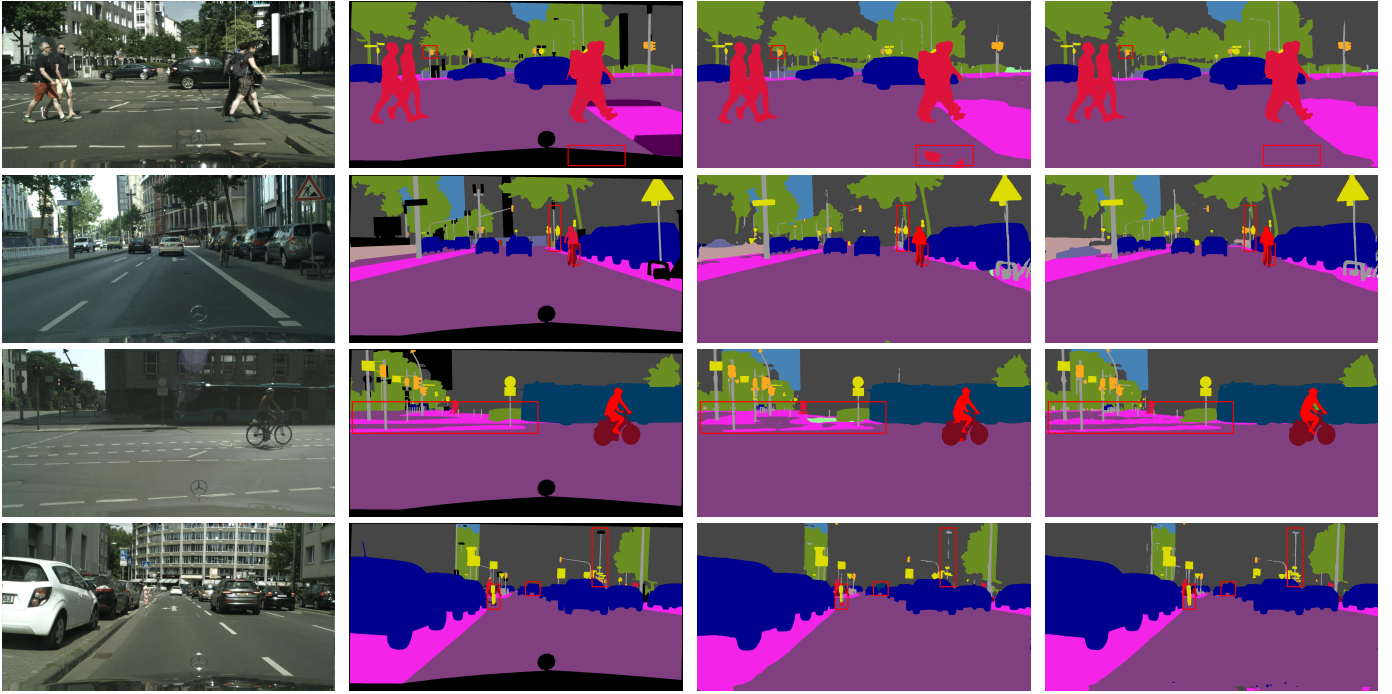


Fig. 6. **Qualitative Visualization on Cityscapes val set.** The figures from left to right are input images, ground truth, segmentation maps of SegFormer [9], segmentation maps of our LRFormer. The significant improvements are indicated by red boxes on segmentation maps.

Pooled Size ↓	FLOPs ↓	Training Memory ↓	mIoU ↑
8×8	38G (-5%)	4.0GB (-5%)	46.8%
16×16	40G	4.2GB	48.5%
32×32	52G (+30%)	5.3GB (+26%)	48.6%
48×48	74G (+85%)	7.4GB (+76%)	48.7%
64×64	108G (+170%)	10.9GB (+160%)	48.5%

TABLE 8

Experiments on the fixed pooled size settings of our LRSA. The performance is saturated when pooled size is larger than 16×16 .

Decoder Head	FLOPs ↓	#Params ↓	mIoU ↑
Ours	40G	32M	49.5%
w/ PPM [4]	82G	44M	48.4%
w/ DA [2]	94G	42M	48.9%
w/ CC [8]	84G	42M	48.6%
w/ OCR [21]	48G	34M	48.0%

TABLE 9

Comparisons of our decoder and other popular decoder heads.

increasing the pooled size to 32^2 , 48^2 , 64^2 , we obtain a minor improvement or even decreased performance on ADE20K semantic segmentation. We also observe that the FLOPs and training memory overhead are much more significant (26% ~ 170%) when the pooled size is larger than 16^2 . Considering the performance, FLOPs and training memory, a low-resolution setting in LRFormer is much more proper.

Comparisons of different decoder heads. Our decoder head aims to predict the semantic maps from multi-level feature maps effectively and efficiently with LRSA. To validate the LRSA of our decoder head, we compare it with several popular decoder heads. These popular decoder heads are designed for CNNs, whose output feature maps are usually $1/8$ of the original image. However, the backbone

encoder of our LRFormer can output features of the $1/32$ of the original image. To make a fair comparison, we first upsample features of the last stages and concatenate them together. Then, we feed them to the popular decoder heads. Other processes keep unchanged in these popular decoder heads. Tab. 9 summarized the results on ADE20K semantic segmentation. The backbone is pretrained for 300 epochs on ImageNet-1K. Compared with PPM [4], DA [2], and CC [8], our LRFormer achieves 1.1%, 0.6%, and 0.9% improvement, respectively, with only fewer than 50% FLOPs. Compared with OCR [21], our LRFormer obtains 1.5% performance gain, with 83% FLOPs. Therefore, our default setting is more efficient and effective than other popular decoder heads.

Bilinear interpolation. Although the self-attention is computed in a low-resolution manner, a bilinear interpolation is needed to fit the size as requested by the residual connection. However, we find that using LRFormer-S with an input size of 512^2 , the bilinear interpolation only has a latency of 0.1ms, constituting a negligible 0.8% of the overall network's latency.

Dimensions of the decoder head. To optimize the performance and computational cost of the decoder head, we employ a 1×1 convolution to reduce the dimension of the concatenated multi-level features before feeding them into the decoder. We conducted experiments with various dimension settings and compared their results in Tab. 10. The backbone is pretrained for 300 epochs on ImageNet-1K. The experiments show that a dimension setting of 512 achieves the best performance. However, setting the dimension to 384 results in only a 0.1% drop in mIoU performance, while saving 25% FLOPs. Therefore, we set the dimension of the decoder head to 384 in our LRFormer-S, reflecting the optimal trade-off between performance and computational

Dimension	FLOPs ↓	#Params ↓	mIoU ↑
128	27G	30M	47.8%
256	32G	31M	49.2%
384	40G	32M	49.5%
512	50G	37M	49.6%
768	79G	46M	49.2%
1024	117G	58M	49.2%

TABLE 10

Discussions on the dimensions of the decoder. When the dimension of the decoder is larger than 384, the performance will be saturated or even decreased.

Method	Size, Batch Size	Memory ↓	FLOPs ↓	Att. FLOPs ↓
LRFormer-S	512×512, 2	4.2GB	40G	0.4G
SegFormer-B2	512×512, 2	7.2GB	62G	3.9G
LRFormer-S	1024×1024, 1	5.7GB	145G	0.4G
SegFormer-B2	1024×1024, 1	18.8GB	291G	62G
LRFormer-L	1024×1024, 1	12.8GB	430G	2.0G
SegFormer-B5	1024×1024, 1	28.8GB	563G	162G

TABLE 11

Analysis of the memory usage and FLOPs for different input size. "Att. FLOPs" indicates the FLOPs of MHSA operations. "Memory" is the training memory for semantic segmentation.

cost.

Memory and FLOPs. Our LRSA has a very low computational complexity of only $O(C^2 + CN)$. We numerically analyze the efficiency of our LRFormer for different input sizes, as well as the comparisons with the representative method SegFormer [9]. The analyzed results on FLOPs, attention FLOPs, and training memory are shown in Tab. 11. Our LRFormer-S costs much less memory and FLOPs than SegFormer-B2. Given input size of 1024×1024 , the number of FLOPs of MHSA operations in our LRFormer is dramatically lower than (0.4G vs. 62G) the self-attention in SegFormer. The training memory SegFormer-B5 is close to 32GB, which is close to the memory limit of a 32GB V100 GPU. Instead, our LRFormer-L only costs 12.8GB memory.

5 CONCLUSION

In this paper, we presented a novel approach to semantic segmentation via introducing the low-resolution self-attention. LRSA computes the self-attention in a fixed low-resolution space, regardless of the size of the input image, making the self-attention highly efficient. Extensive experiments on ADE20K [28], COCO-Stuff [29] and Cityscapes [30] datasets show that LRFormer outperforms state-of-the-art models, suggesting that LRSA is adequate to keep global receptive field with negligible computational cost. This study provides evidence for the effectiveness of LRSA and opens a new direction for future research.

Acknowledgements. This work is funded by NSFC (NO. 62225604, 62176130), and the Fundamental Research Funds for the Central Universities (Nankai University, 070-63233089). Computation is supported by the Supercomputing Center of Nankai University.

REFERENCES

[1] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convo-

lutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2017.

[2] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 3146–3154.

[3] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Int. Conf. Comput. Vis.*, 2019, pp. 593–602.

[4] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 2881–2890.

[5] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, "DenseASPP for semantic segmentation in street scenes," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 3684–3692.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 770–778.

[7] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 1492–1500.

[8] Z. Huang, X. Wang, Y. Wei, L. Huang, H. Shi, W. Liu, and T. Huang, "CCNet: Criss-cross attention for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[9] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers," *Adv. Neural Inform. Process. Syst.*, vol. 34, 2021.

[10] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang, "Hrformer: High-resolution vision transformer for dense predict," *Adv. Neural Inform. Process. Syst.*, vol. 34, 2021.

[11] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. C. Loy, D. Lin, and J. Jia, "PSANet: Point-wise spatial attention network for scene parsing," in *Eur. Conf. Comput. Vis.*, 2018, pp. 267–283.

[12] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, "Context encoding for semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 7151–7160.

[13] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Learning a discriminative feature network for semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 1857–1866.

[14] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-scnn: Gated shape cnns for semantic segmentation," in *Int. Conf. Comput. Vis.*, 2019, pp. 5229–5238.

[15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Int. Conf. Learn. Represent.*, 2021.

[16] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," *arXiv preprint arXiv:2012.12877*, 2020.

[17] 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 *Int. Conf. Comput. Vis.*, 2021.

[18] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo, "Cswin transformer: A general vision transformer backbone with cross-shaped windows," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 12 124–12 134.

[19] 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 *Int. Conf. Comput. Vis.*, 2021.

[20] Y.-H. Wu, Y. Liu, X. Zhan, and M.-M. Cheng, "P2T: Pyramid pooling transformer for scene understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.

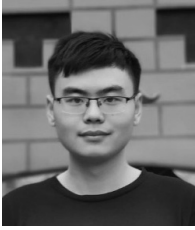
[21] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *Eur. Conf. Comput. Vis.* Springer, 2020, pp. 173–190.

[22] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Int. Conf. Learn. Represent.*, 2016.

[23] 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.

[24] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong et al., "Swin transformer v2: Scaling up capacity and resolution," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 12 009–12 019.

- [25] J. Yang, C. Li, P. Zhang, X. Dai, B. Xiao, L. Yuan, and J. Gao, "Focal self-attention for local-global interactions in vision transformers," 2021.
- [26] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in *Int. Conf. Comput. Vis.*, 2021, pp. 6824–6835.
- [27] Y. Li, C.-Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, and C. Feichtenhofer, "MViTv2: Improved multiscale vision transformers for classification and detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 4804–4814.
- [28] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 633–641.
- [29] H. Caesar, J. Uijlings, and V. Ferrari, "COCO-Stuff: Thing and stuff classes in context," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 1209–1218.
- [30] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3213–3223.
- [31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
- [32] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3431–3440.
- [33] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.
- [34] H. Ding, X. Jiang, A. Q. Liu, N. M. Thalmann, and G. Wang, "Boundary-aware feature propagation for scene segmentation," in *Int. Conf. Comput. Vis.*, 2019, pp. 6819–6829.
- [35] X. Li, X. Li, L. Zhang, G. Cheng, J. Shi, Z. Lin, S. Tan, and Y. Tong, "Improving semantic segmentation via decoupled body and edge supervision," in *Eur. Conf. Comput. Vis.* Springer, 2020, pp. 435–452.
- [36] Y. Yuan, J. Xie, X. Chen, and J. Wang, "Segfix: Model-agnostic boundary refinement for segmentation," in *Eur. Conf. Comput. Vis.* Springer, 2020, pp. 489–506.
- [37] M. Zhen, J. Wang, L. Zhou, S. Li, T. Shen, J. Shang, T. Fang, and L. Quan, "Joint semantic segmentation and boundary detection using iterative pyramid contexts," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 13 666–13 675.
- [38] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, "Ocnet: Object context network for scene parsing," *arXiv preprint arXiv:1809.00916*, 2018.
- [39] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr *et al.*, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021, pp. 6881–6890.
- [40] B. Cheng, A. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," *Adv. Neural Inform. Process. Syst.*, vol. 34, pp. 17 864–17 875, 2021.
- [41] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 1290–1299.
- [42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. Neural Inform. Process. Syst.*, vol. 25, pp. 1097–1105, 2012.
- [43] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [44] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 1–9.
- [45] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 4700–4708.
- [46] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 652–662, 2019.
- [47] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, H. Lin, Z. Zhang, Y. Sun, T. He, J. Mueller, R. Manmatha *et al.*, "Resnest: Split-attention networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 2736–2746.
- [48] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 7132–7141.
- [49] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 510–519.
- [50] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022.
- [51] X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31x31: Revisiting large kernel design in CNNs," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 11 963–11 975.
- [52] S. Liu, T. Chen, X. Chen, X. Chen, Q. Xiao, B. Wu, M. Pechenizkiy, D. Mocuano, and Z. Wang, "More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity," *arXiv preprint arXiv:2207.03620*, 2022.
- [53] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, 2020.
- [54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Adv. Neural Inform. Process. Syst.*, 2017, pp. 5998–6008.
- [55] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *Int. Conf. Mach. Learn.* PMLR, 2021, pp. 10 347–10 357.
- [56] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, F. E. Tay, J. Feng, and S. Yan, "Tokens-to-token ViT: Training vision transformers from scratch on ImageNet," in *Int. Conf. Comput. Vis.*, 2021, pp. 558–567.
- [57] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "CvT: Introducing convolutions to vision transformers," in *Int. Conf. Comput. Vis.*, 2021, pp. 22–31.
- [58] X. Chu, Z. Tian, B. Zhang, X. Wang, X. Wei, H. Xia, and C. Shen, "Conditional positional encodings for vision transformers," in *Int. Conf. Learn. Represent.*, 2023.
- [59] D. Zhou, Z. Yu, E. Xie, C. Xiao, A. Anandkumar, J. Feng, and J. M. Alvarez, "Understanding the robustness in vision transformers," in *Int. Conf. Mach. Learn.* PMLR, 2022, pp. 27 378–27 394.
- [60] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 10 819–10 829.
- [61] Z. Xia, X. Pan, S. Song, L. E. Li, and G. Huang, "Vision transformer with deformable attention," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 4794–4803.
- [62] W. Xu, Y. Xu, T. Chang, and Z. Tu, "Co-scale conv-attentional image transformers," in *Int. Conf. Comput. Vis.*, 2021, pp. 9981–9990.
- [63] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2020.
- [64] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," 2022.
- [65] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Int. Conf. Learn. Represent.*, 2018.
- [66] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou, "Going deeper with image transformers," in *Int. Conf. Comput. Vis.*, 2021, pp. 32–42.
- [67] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Int. Conf. Comput. Vis.*, 2021, pp. 12 179–12 188.
- [68] M. Ding, B. Xiao, N. Codella, P. Luo, J. Wang, and L. Yuan, "Davit: Dual attention vision transformers," in *Eur. Conf. Comput. Vis.* Springer, 2022, pp. 74–92.
- [69] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.
- [70] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 761–769.
- [71] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Eur. Conf. Comput. Vis.*, 2018, pp. 418–434.
- [72] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 10 428–10 436.
- [73] J. Yang, C. Li, and J. Gao, "Focal modulation networks," *arXiv preprint arXiv:2203.11926*, 2022.



Yu-Huan Wu received his Ph.D. degree from Nankai University in 2022, advised by Prof. Ming-Ming Cheng. Currently, he is a research scientist at the Institute of High Performance Computing (IHPC), Agency for Science, Technology and Research (A*STAR), Singapore. He has published 10+ papers on top-tier conferences and journals such as IEEE TPAMI/TIP/CVPR/ICCV. His research interests include computer vision, medical imaging and autonomous driving.



Daquan Zhou received the PhD degree from NUS, under the supervision of Prof. Jiashi Feng. He is currently a research scientist at Bytedance. His research interests include deep learning, neural network compression, neural network structure design, and AutoML.



Shi-Chen Zhang received his B.E. degree in computer science from Nankai University in 2023. Currently, he is a Ph.D. student in Media Computing Lab, Nankai University, supervised by Prof. Ming-Ming Cheng. His research interests include object detection and semantic segmentation.



Jiashi Feng received the PhD degree from NUS, in 2014. He is currently a research lead with ByteDance. Before joining ByteDance, he was assistant professor with the Department of Electrical and Computer Engineering, National University of Singapore. His research areas include deep learning and their applications in computer vision. He received the best technical demo award from ACM MM 2012, best paper award from TASK-CV ICCV 2015, best student paper award from ACM MM 2018.



Yun Liu received his B.E. and Ph.D. degrees from Nankai University in 2016 and 2020, respectively. Then, he worked with Prof. Luc Van Gool for one and a half years as a postdoctoral scholar at Computer Vision Lab, ETH Zurich. Currently, he is a senior scientist at the Institute for Infocomm Research (I2R), Agency for Science, Technology and Research (A*STAR), Singapore. His research interests include computer vision and machine learning.



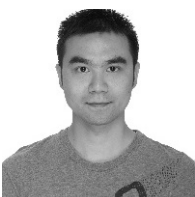
Ming-Ming Cheng received his Ph.D. degree from Tsinghua University in 2012. Then, he did two years research fellow with Prof. Philip Torr in Oxford. He is now a professor at Nankai University, leading the Media Computing Lab. His research interests include computer graphics, computer vision, and image processing. He received research awards, including ACM China Rising Star Award, IBM Global SUR Award, and CCF-Intel Young Faculty Researcher Program. He is on the editorial boards of IEEE TPAMI/TIP.



Le Zhang received his M.Sc and Ph.D. degree from Nanyang Technological University (NTU) in 2012 and 2016, respectively. Currently, he is a professor at UESTC. He served as TPC member in several conferences such as AAAI, IJCAI. He has served as a Guest Editor for Pattern Recognition and Neurocomputing; His current research interests include deep learning and computer vision.



Liangli Zhen received his Ph.D. degree in computer science from Sichuan University in 2018. He is a senior scientist and group manager at the Institute of High Performance Computing (IHPC), Agency for Science, Technology and Research (A*STAR), Singapore. His current research interests include machine learning and computer vision. He has published more than 30 papers in top tier journals and conferences including TPAMI/TNNLS/CVPR/ICCV.



Xin Zhan received his bachelor's and doctoral degrees from USTC in 2010 and 2015, respectively. He works as a researcher of Udeer AI. His research interests include perception for autonomous driving.