SDPT: Semantic-Aware Dimension-Pooling Transformer for Image Segmentation

Hu Cao, Guang Chen, Hengshuang Zhao, *Member, IEEE*, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, *Fellow, IEEE*, and Alois Knoll, *Fellow, IEEE*

***Abstract*— Image segmentation plays a critical role in autonomous driving by providing vehicles with a detailed and accurate understanding of their surroundings. Transformers have recently shown encouraging results in image segmentation. However, transformer-based models are challenging to strike a better balance between performance and efficiency. The computational complexity of the transformer-based models is quadratic with the number of inputs, which severely hinders their application in dense prediction tasks. In this paper, we present the semantic-aware dimension-pooling transformer (SDPT) to mitigate the conflict between accuracy and efficiency. The proposed model comprises an efficient transformer encoder for generating hierarchical features and a semantic-balanced decoder for predicting semantic masks. In the encoder, a dimension-pooling mechanism is used in the multi-head self-attention (MHSA) to reduce the computational cost, and a parallel depth-wise convolution is used to capture local semantics. Simultaneously, we further apply this dimension-pooling attention (DPA) to the decoder as a refinement module to integrate multi-level features. With such a simple yet powerful encoder-decoder framework, we empirically demonstrate that the proposed SDPT achieves excellent performance and efficiency on various popular benchmarks, including ADE20K, Cityscapes, and COCO-Stuff. For example, our SDPT achieves 48.6% mIOU on the ADE20K dataset, which outperforms the current methods with fewer computational costs. The codes can be found at https://github.com/HuCaoFighting/SDPT.**

***Index Terms*— Image segmentation, vision transformer, dimension-pooling attention, semantic-balanced decoder, scene understanding.**

I. Introduction

IMAGE segmentation is a fundamental task in computer vision that involves partitioning an image into multiple regions or segments. Each segment typically represents a meaningful part of the image, such as objects, boundaries, or regions with similar properties. Early approaches often relied on simple methods such as thresholding, edge detection, and region growing [2]. These techniques were limited in their ability to handle complex images with varying lighting conditions, textures, and object orientations. Subsequently, region-based segmentation algorithms gained popularity. These methods divide an image into regions based on similarities in color, intensity, texture, or other low-level features. Region growing [3], watershed transformation [4], and mean-shift clustering [5] are examples of region-based segmentation techniques. While effective for certain types of images, region-based methods often struggle with handling noise, occlusions, and overlapping objects. Edge-based segmentation techniques focus on detecting boundaries or edges between different image regions. Edge detection algorithms, such as the Canny edge detector [6], Sobel operator [7], and Prewitt operator [8], identify abrupt changes in pixel intensity, which often correspond to object boundaries. While edge-based methods are sensitive to noise and may produce fragmented segmentations, they are useful for tasks like object detection and contour extraction. In recent years, the field of image segmentation has been revolutionized by the widespread adoption of machine learning techniques, particularly deep learning. Convolutional neural networks (CNNs) have emerged as powerful tools for learning feature representations directly from image data, enabling end-to-end segmentation pipelines [9]. Architectures like U-Net [10], FCN (Fully Convolutional Network) [11], and Mask R-CNN [12] have demonstrated state-of-the-art (SOTA) performance in tasks such as semantic segmentation and instance segmentation. As shown in Fig. 1, semantic segmentation assigns a class label to each pixel in an image, effectively partitioning the image into semantically meaningful regions. Instance segmentation, on the other hand, goes a step further by not only identifying object categories but also distinguishing individual object instances within the same category. These advanced segmentation tasks have numerous applications in autonomous driving [13], [14], medical imaging [15], video surveillance [16], remote sensing [17], augmented reality [18], robotic perception [19], aerial semantic segmentation [20], [21], and more.

Image segmentation is an important technology in the field of autonomous driving, enabling vehicles to perceive and understand their surroundings with great precision [22], [23].

Manuscript received 27 May 2023; revised 6 May 2024; accepted 28 May 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62372329, in part by Shanghai Scientific Innovation Foundation under Grant 23DZ1203400, in part by Shanghai Rising Star Program under Grant 21QC1400900, in part by Tongji–Qomolo Autonomous Driving Commercial Vehicle Joint Laboratory Project, and in part by Xiaomi Young Talents Program. The Associate Editor for this article was V. Chamola. *(Corresponding author: Guang Chen.)*

Hu Cao and Alois Knoll are with the Chair of Robotics, Artificial Intelligence and Real-Time Systems, Technical University of Munich, 80333 Munich, Germany.

Guang Chen is with the Department of Computer Science and Technology, Tongji University, Shanghai 200070, China (e-mail: guangchen@tongji.edu.cn).

Hengshuang Zhao is with the Department of Computer Science, The University of Hong Kong, Hong Kong.

Dongsheng Jiang, Xiaopeng Zhang, and Qi Tian are with Huawei Technologies, Shanghai 200122, China.

Digital Object Identifier 10.1109/TITS.2024.3417813

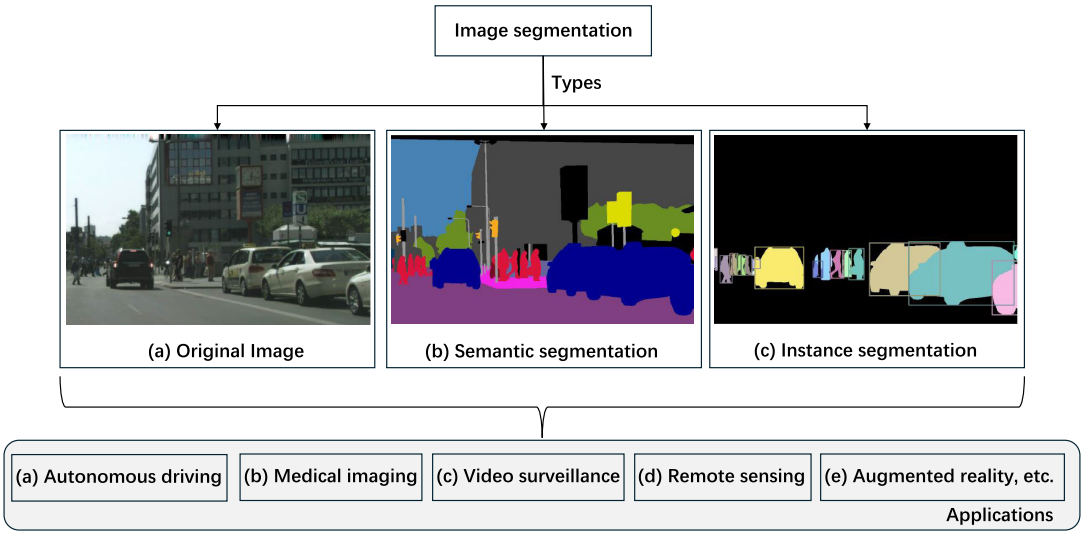


Fig. 1. An overview of image segmentation reveals its diverse types, such as semantic segmentation and instance segmentation. These techniques are integral to a wide range of applications, including autonomous driving, medical imaging, video surveillance, remote sensing, augmented reality, and beyond. Images are selected from the cityscapes [1] dataset.

This fine-grained segmentation allows self-driving cars to gain a detailed understanding of their environment, effectively distinguishing key elements such as roads, pedestrians, traffic signs, and vehicles [24]. The accurate identification and differentiation provided by image segmentation enable self-driving cars to make informed decisions, navigate through complex situations, and ultimately improve road safety. With its ability to provide high-resolution environmental understanding, image segmentation plays a key role in the development and implementation of self-driving cars, promising a future where transportation is more efficient, reliable, and safe [25].

CNNs with intrinsic inductive bias serve as the prevalent backbone in image segmentation [11], [26], [27], [28]. However, CNNs are good at modeling local visual features (*e.g.*, edges and corners) but are not suitable for modeling long-range information dependencies. The transformers, the *de-facto* dominant model in natural language processing (NLP) studies, have been introduced to vision tasks [29], [30], [31]. The key idea behind transformer is its strong ability to model long-range dependencies through the self-attention mechanism [32]. Benefiting from transformer's powerful representational learning capabilities, researchers have achieved excellent results in image classification [29], [30], [33], object detection [34], [35], [36], and image segmentation [15], [37], [38], [39], etc.

In the field of segmentation task, the performance gains achieved by transformer-based models compared to CNN-based approaches are mainly attributed to their powerful backbone networks as encoders. The main weakness of transformer-based encoders is that their self-attention mechanism has higher time and memory costs compared to convolutional operations. The complexity of the transformer-based model is $O(w^2h^2)$ for $w \times h$ inputs, which severely limits its application in dense prediction tasks (*e.g.*, semantic segmentation). To alleviate this limitation, Swin Transformer [31] adopts window-based self-attention to reduce the computation cost. Restricting attention computation to local windows is efficient, but it compromises the transformer's ability to model long-range dependencies. PVT [35], SegFormer [37], and PVTv2 [36] down-sample the spatial structure of *key* and *value* to reduce the computational complexity. Moreover, P2T [40] uses pyramid pooling to reduce the computational cost while capturing powerful contextual features. On the other hand, the spatial reduction strategy uses down-sampling to improve the computational efficiency but sacrifices the spatial structure of the feature map. In this work, we introduce a dimension-pooling mechanism to improve the efficiency of self-attention while preserving the feature map's spatial structure. Based on the dimension-pooling attention (DPA), we design a novel hierarchical transformer encoder to generate multi-scale features.

For segmentation tasks, an effective decoder is important to capture the high-level semantics. Four representative designs of decoder structures are: (*i*) the output of the encoder is directly fed into a heavy decoder, such as ASPP [27], PSP [41], and DANet [42]; (*ii*) a symmetric decoder is used to up-sample the features from the encoder, such as U-Net [10] and V-Net [43]; (*iii*) a simple MLP-based decoder, such as SegFormer [37]; and (*iv*) a transformer-based decoder is used to model global context, such as Segmenter [44] and MaskFormer [45]. Despite the excellent performance achieved by these methods, two key challenges remain when aggregating multi-level features of the encoder; namely, how to maintain semantic consistency within the same level of features and how to bridge the context across different levels of features. Deep high-level features contain more abstract semantic information, while shallow low-level features provide more content descriptions [46]. Previous work, such as FPN [47], PANet [48], and SETR [38], utilized lateral connections for feature interaction. These methods demonstrate that high-level features and low-level features are complementary, but they focus more on adjacent level features and less on other level features. Inspired

by [49], we use the balanced semantic features to strengthen the multi-level features. The DPA is deployed to refine the balanced semantic features to be more discriminative. Each level of features can then obtain equal information from the other levels of features, balancing the information flow. Combined with our hierarchical transformer encoder, a simple yet effective encoder-decoder framework is established.

Our segmentation model consists of a novel hierarchical transformer encoder and a semantic-balanced decoder, named SDPT. The proposed SDPT achieves the best trade-off between segmentation performance and efficiency compared to the previous transformer-based methods.

Our main contributions can be summarized as follows:

- In order to generate multi-scale features and improve efficiency, a novel hierarchical transformer encoder with dimension-pooling attention (DPA) is implemented.
- A semantic-balanced decoder is introduced to strengthen the multi-level features. In the decoder, DPA is further used as a refinement module to make the balanced features more discriminative.
- Our method outperforms current segmentation models on three publicly available semantic segmentation datasets (including ADE20K [50], Cityscapes [1], and COCO-Stuff [51]) with lower computational complexity.

II. Related Work

A. Traditional Segmentation Methods

Classic approaches such as thresholding, edge detection, region-based segmentation, and clustering have laid the groundwork for subsequent research in computer vision. Early methods like global thresholding [52] and the Sobel operator [7] paved the way for more sophisticated techniques, including adaptive thresholding [53], Canny edge detection [6], region-growing [3], and clustering [14] algorithms. These methods, though simple in concept, have demonstrated effectiveness in segmenting images with well-defined features and distinct boundaries. Moreover, research efforts have extended to hybrid approaches that combine multiple segmentation techniques to achieve enhanced performance and robustness. While traditional segmentation methods excel in certain scenarios, their efficacy is often challenged by complex image structures, noise, and variability in lighting conditions.

B. CNN-Based Segmentation Methods

CNN-based segmentation methods have emerged as SOTA techniques for image segmentation tasks. Leveraging the power of deep learning, CNNs have revolutionized the field by learning hierarchical representations directly from raw image data [26]. Seminal works such as the FCN [11], U-Net [10], DeepLab series [27], [28], Strip pooling [54], ECANet [55], and Mask RCNN [12] have demonstrated remarkable performance in semantic segmentation, omni-directional segmentation, and instance segmentation. These architectures leverage convolutional layers to capture spatial dependencies and learn feature representations at multiple scales, enabling accurate and efficient segmentation of complex scenes. Moreover, advancements in CNN architectures, such as the integration of skip connections [56], atrous convolutions [27], and attention mechanisms [20], [54], [55], [57], have further improved segmentation performance and robustness. In [57], the authors introduce a criss-cross attention module aimed at gathering contextual information from full-image dependencies in a more efficient and effective manner. The concept of strip pooling, proposed in [54], focuses on capturing long-range dependencies while retaining attention to local details. Additionally, the Horizontal Segment Attention (HSA) module, developed in ECANet [55], is designed to facilitate omnidirectional semantic segmentation. Recently, SegNeXt [58] is propsed based on multi-scale convolutional attention (MSCA) module. Despite their success, CNN-based segmentation methods still face challenges related to data scarcity, overfitting, and generalization to diverse image domains.

C. Transformer-Based Segmentation Methods

ViT [29] is the first work that demonstrates transformer-based methods can achieve comparable performance in the vision task. DeiT [30] is proposed to facilitate training by bringing the idea of distilling knowledge from CNNs to Transformers. However, both ViT and DeiT are columnar architectures that maintain the same spatial scale across layers. Referring to the hierarchical structure of CNNs, more multi-scale transformer models such as PVT [35], Swin [31], PVTv2 [36], Shunted Transformer [59] and P2T [40] are proposed to perform dense prediction tasks. Various CNN-based approaches [60], [61], [62] have been explored to address the real-time segmentation problem. Previously, researchers have used pre-trained ViTs as encoders to improve segmentation performance [38], [44], [63]. Compared to CNN-based methods, the computational complexity of transformer-based approaches grows quadratically with the number of input tokens. Swin Transformer [31] addresses this challenge by introducing window-based self-attention, reducing computational costs. However, confining attention computation to local windows, although efficient, compromises the model's ability to capture long-range dependencies. Techniques employed by PVT [35], PVTv2 [36], and SegFormer [37] involve downsampling the spatial structure of *key* and *value* to mitigate computational complexity. Additionally, P2T [40] employs pyramid pooling to lower computational costs while retaining powerful contextual features. Conversely, spatial reduction strategies using downsampling enhance computational efficiency, but at the expense of spatial structure in the feature map. Recently, TopFormer [64], PIDNet [65], SeaFormer [66], AFFormer [67], and SCTNet [68] were developed for real-time semantic segmentation. SegViT [39] and VWFormer [69] are proposed to achieve excellent segmentation performance. Moreover, the authors of CMX [70] presented a multimodal fusion segmentation method based on the backbone of SegFormer [37]. In this work, we introduce a dimension-pooling attention (DPA) to improve the efficiency of the transformer-based encoder. We further demonstrate that applying our DPA to the encoder and decoder can achieve outstanding performance-efficiency trade-offs.

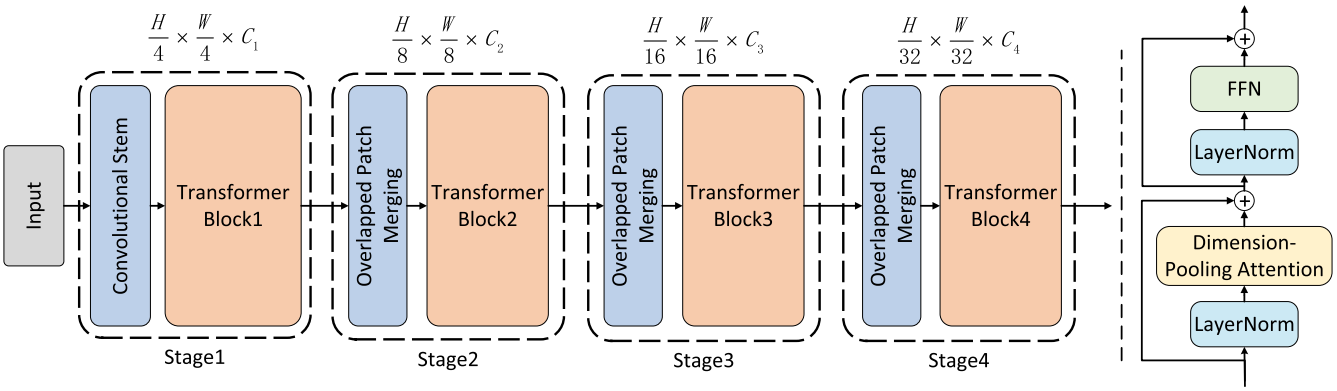


Fig. 2. **Left:** the overall architecture of our hierarchical transformer encoder. The input is passed through a hierarchical transformer encoder to generate multi-scale features. This encoder is structured into four stages. Within each stage, a convolutional stem and overlapped patch merging layer down-sample features, while a transformer block conducts representation learning. **Right:** details of the proposed transformer block. It consists of three main components: layer normalization, dimension-pooling attention (DPA), and a feed-forward network (FFN). The DPA efficiently captures long-range relationships between input tokens, while the expanded FFN is employed to learn wider representations.

III. Method

In this section, we introduce the framework of the proposed SDPT. Following the popular encoder-decoder architecture, the SDPT consists of an efficient transformer encoder and a semantic-balanced decoder. The encoder aims to extract multi-scale features, and the decoder aggregates these multi-level features to perform semantic mask prediction.

A. Encoder

The encoder comprises four stages for generating multi-scale features, as shown in Fig. 2. We use a convolutional stem and overlapped patch merging layers to down-sample features and a transformer block to perform representation learning. In the following, we elaborate on each module in detail.

1) Transformer Block: As shown in Fig. 2 right, the proposed transformer block contains a layer norm, a dimension-pooling attention (DPA), and a feed-forward network (FFN). The DPA is used to efficiently model long-range relationships between the input tokens and the expanded FFN is utilized to learn wider representations.

In a vanilla transformer [29], it builds long-range dependence through multi-head self-attention (MHSA), which can be formulated as follows:

$$Attention = SoftMax(\frac{QK^T}{\sqrt{D_h}})V \tag{1}$$

where Q, K, and V are query, key, and value tensors, respectively. D_h denotes the head dimension. The computational complexity of the original MHSA module on an image of $h \times w$ patch tokens is:

$$\Omega(MHSA) = 4hwC^2 + 2(hw)^2C \tag{2}$$

Low computational complexity is crucial for the semantic segmentation task. However, the cost of MHSA is quadratic with the number of patch tokens ($O(h^2w^2)$), which severely limits its application in semantic segmentation. The Swin Transformer [31] restricts self-attention within the local window and PVT [35], PVTv2 [36], and SegFormer [37] down-sample the spatial structure of K and V to improve the efficiency of MHSA. Unlike the previous works, we use a dimension-pooling mechanism to reduce the computational cost of MHSA. To shorten the input sequence while preserving spatial structure, we employ global average pooling to shrink the input tokens of K and V from the size of HW to the size of $H + W$, as shown in Fig. 3.

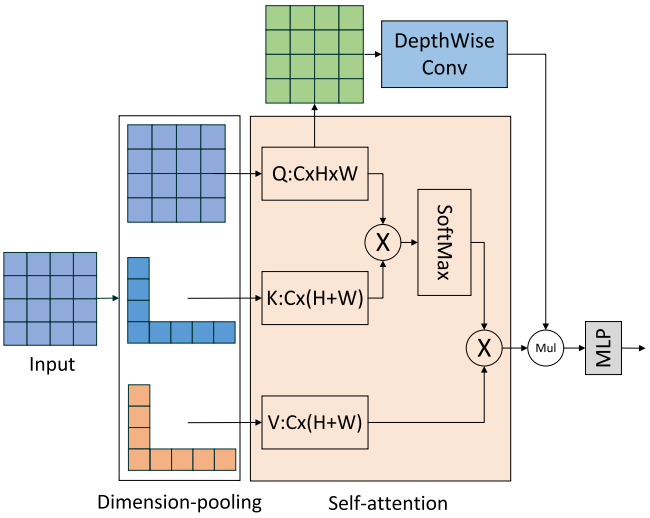


Fig. 3. The structure of dimension-pooling attention (DPA). To improve the efficiency of the attention operation, the input tokens of K and V are pooled from the size of HW to the size of $H+W$. A parallel depth-wise convolution is further deployed on the query tensors Q to provide high-frequency local information.

2) Dimension-Pooling Attention (DPA): The inputs are encoded along with the horizontal and lateral directions by two spatial extents with pooling kernels $(H, 1)$ and $(1, W)$. The encoded horizontal average tensor H_n and lateral average tensor W_n are then concatenated together for attention operation. The whole calculation process can be expressed as follows:

$$\begin{aligned} H_n &= \frac{1}{W} \sum_{0 \le i < W} x_n(h, i), n \in (K, V) \\ W_n &= \frac{1}{H} \sum_{0 \le j < H} x_n(j, w), n \in (K, V) \\ P_n &= Concat(H_n, W_n), n \in (K, V) \end{aligned} \tag{3}$$

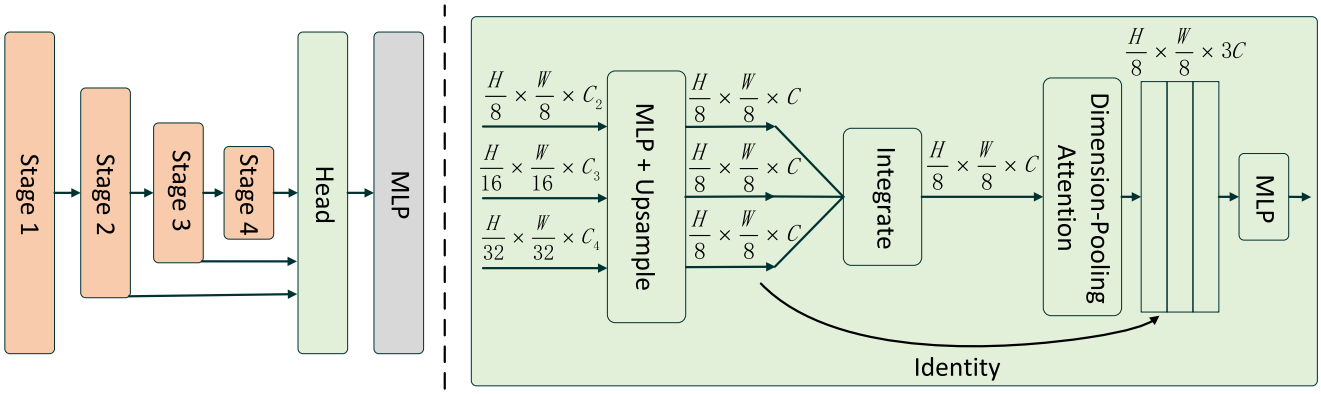


Fig. 4. **Left:** the overall architecture of our SDPT. The features extracted from stages 2, 3, and 4 are fed into the decoder head for semantic mask prediction. **Right:** details of our decoder. The proposed DPA is further utilized in the decoder to refine the features.

where x_n and P_n are input features and concatenated pooling tensors, respectively. Through this transformation, the global context can be captured in spatial direction while improving computational efficiency. Finally, the DPA is computed as follows:

$$Attention(Q, P_K, P_V) = SoftMax(\frac{QP_K^T}{\sqrt{D_h}})P_V \quad (4)$$

Furthermore, we use a parallel depth-wise convolution on the query tensors Q to provide high-frequency local information to compensate for the information loss caused by pooling operations.Â The final output of the attention module, x_{att}, is as follows:

$$x_{att} = MLP(Mul(Attention(Q, P_K, P_V), DWConv(Q))) \quad (5)$$

The computational complexity of our DPA is expressed as follows (only for attention operation):

$$\Omega(DPA) = 2(h + w + hw)C^2 + 2(hw)(h + w)C \quad (6)$$

3) FFN: Similar to [37] and [40], we add a depth-wise convolution with a kernel size of 3×3 and a padding size of 1 between the first MLP layer and the GELU non-linear activation in the feed-forward network (FFN). The computation can be defined using the following formula:

$$\begin{aligned} x_1 &= MLP_1(x_{att}) \\ x_2 &= DWConv(x_1) + x_1 \\ x_{out} &= MLP_2(GELU(x_2)) + x_{att} \end{aligned} \quad (7)$$

where x_{att} is the feature from the DPA module, x_1 is the output of the first MLP layer, x_2 is the value of x_1 after the convolution operation and residual connection, and x_{out} is the feature of the second MLP layer and residual connection.

4) Convolutional Stem: In the first stage, the convolutional stem is used to transform an input of size $H \times W \times 3$ into patch tokens of size 4×4. Unlike [29], [31], and [35], which directly use a convolutional layer with a large kernel size to split the input into non-overlapped patch tokens, we employ four consecutive convolutional blocks with a kernel size of 3×3 to form a convolutional stem. The reason for this design is that an early convolutional stem helps transformers be more robust and better, which has been demonstrated in [71]. Moreover, compared to the convolutional layer with a large kernel size used in the SegFormer [37], sequential convolution with a small kernel size can reduce the parameters without compromising the receptive field. Each convolutional block is made up of a convolution with a kernel size of 3×3, a BatchNorm (BN) layer, and the GELU non-linearity in between each convolutional layer.

5) Overlapped Patch Merging: In the next three stages, we use one-layer 3×3 convolution with a stride of 2 and 1 padding to perform overlapped patch merging to produce CNN-like multi-scale feature maps. Take the second stage as an example. The feature maps are shrunk from $F_1(\frac{W}{4} \times \frac{H}{4} \times C_1)$ to $F_2(\frac{W}{8} \times \frac{H}{8} \times C_2)$ while reserving the local continuity, and the remaining multi-scale feature maps are generated in the same manner.

6) Model Details: Following previous backbone designs [31], [35], [36], [37], [56], we built our encoder in four stages. As the network depth deepens, the resolution of feature maps decreases, whereas the channel dimension of feature maps increases. Hence, the encoder generates four feature maps: F_1, F_2, F_3, and F_4. F_i has a dimension of $\frac{W}{2^{i+1}} \times \frac{H}{2^{i+1}} \times C_i$, where $i \in \{1, 2, 3, 4\}$. Totally, we devised three encoder models with different sizes, named SDPT-Tiny, SDPT-Small, and SDPT-Base. In Table I, the detailed network settings are listed.

B. Decoder

In segmentation models, a decoder is deployed on the encoder to capture high-level semantics. Current methods usually utilize convolution, MLP, or attention-based modules as decoders to integrate multi-level features, such as SETR [38], SegFormer [37], MaskFormer [45], and SegNeXt [58]. Different from these methods, we introduce a decoder to strengthen multi-level features using the same balanced semantic features. The detailed structure is depicted in Fig. 4. The features from Stage 1 consume more computational resources but bring little performance improvement due to too much low-level information and higher resolution. Therefore, we aggregate the features from the last three stages of the encoder. First,

TABLE I

DETAILED SETTINGS OF DIFFERENT SIZES OF THE PROPOSED SDPT. C DENOTES THE OUTPUT CHANNEL NUMBER, E REPRESENTS THE EXPANSION RATIO IN FFN, AND L INDICATES THE NUMBER OF TRANSFORMER BLOCKS. 'PARAMETERS' ARE CALCULATED ON THE ADE20K DATASET [50]

Stage	Output Size	Head	Layer Name	SDPT		
				Tiny	Small	Base
1	$\frac{H}{4} \times \frac{W}{4} \times C_1$	1	Convolutional Stem	$C_1 = 32$	$C_1 = 48$	$C_1 = 64$
			Transformer Block	$E_1 = 8$, $L_1 = 2$	$E_1 = 8$, $L_1 = 2$	$E_1 = 8$, $L_1 = 2$
2	$\frac{H}{8} \times \frac{W}{8} \times C_2$	2	Overlapped Patch Merging	$C_2 = 64$	$C_2 = 96$	$C_2 = 128$
			Transformer Block	$E_2 = 8$, $L_2 = 2$	$E_2 = 8$, $L_2 = 2$	$E_2 = 8$, $L_2 = 2$
3	$\frac{H}{16} \times \frac{W}{16} \times C_3$	5	Overlapped Patch Merging	$C_3 = 160$	$C_3 = 240$	$C_3 = 320$
			Transformer Block	$E_3 = 4$, $L_3 = 2$	$E_3 = 4$, $L_3 = 6$	$E_3 = 4$, $L_3 = 9$
4	$\frac{H}{32} \times \frac{W}{32} \times C_4$	8	Overlapped Patch Merging	$C_4 = 256$	$C_4 = 384$	$C_4 = 512$
			Transformer Block	$E_4 = 4$, $L_4 = 2$	$E_4 = 4$, $L_4 = 3$	$E_4 = 4$, $L_4 = 3$
Decoder dimension				256	256	768
Parameters (M)				3.6	11.9	28.6

multi-level features F_i from the encoder are fed into an MLP layer to unify the channel dimension. Then, these unified features are up-sampled to $\frac{H}{8} \times \frac{W}{8}$ and integrated together. The balanced semantic features F are obtained by the averaging operation. We further utilize our dimension-pooling attention (DPA) to refine the balanced semantic features to be more discriminative. The refined features are then used to strengthen the original features through residual connections. In this manner, each level feature gets equal information from the others. The computation can be defined as follows:

$$\begin{aligned} F_i' &= MLP(C_i, C)(F_i), i \in (2,3,4) \\ F_i' &= Upsample(\frac{W}{8} \times \frac{H}{8})(F_i'), i \in (2,3,4) \\ F &= \frac{1}{N}\sum_{i=2}^{4} F_i', i \in (2,3,4) \\ F &= DPA(F) \\ Y_i &= F_i' + F, i \in (2,3,4) \end{aligned} \tag{8}$$

where $MLP(C_{in}, C_{out})$ denotes a MLP layer with C_{in} and C_{out} as input and output vector dimension, respectively. N represents the number of multi-level features. By fusing these balanced semantic features, the final output of the decoder is expressed as follows:

$$\begin{aligned} Y &= MLP(3C, C)(Concat(Y_i)), i \in (2,3,4) \\ M &= MLP(C, N_{cls})(Y) \end{aligned} \tag{9}$$

where N_{cls} and M denote the number of categories and the predicted semantic mask, respectively.

IV. EXPERIMENTAL SETTINGS

A. Datasets

Following previous methods, we pre-train each encoder variant using the ImageNet-1K dataset [72]. ImageNet-1K is a popular dataset with 1000 categories for image classification. For semantic segmentation, we use three publicly available datasets to evaluate our SDPT, including ADE20K [50], Cityscapes [1], and COCO-Stuff [51]. ADE20K is a challenging scene-parsing dataset covering 150 semantic classes. In this dataset, there are 20210 images for training, 2000 images for validation, and 3352 images for the test. Cityscapes is a driving dataset that contains 5000 high-resolution images in 19 categories. It consists of 2975 images in the training set, 500 images in the validation set, and 1525 images in the test set. The COCO-Stuff dataset includes 164k images with 172 semantic categories; 118k of these images are utilized for training, 5k for validation, 20k for test development, and 20k for test challenge.

TABLE II

TRAINING DETAILS FOR THE THREE DATASETS

Dataset	Crop Size	Batch Size	Iterations
ADE20K [50]	512×512	16	160K
Cityscapes [1]	1024×1024	8	160K
COCO-Stuff [51]	512×512	16	80K

B. Implementation Details

We implement our models based on the Pytorch [73], Timm [74], and Mmsegmentation [75] libraries. All encoder variants are pre-trained on the ImageNet-1K dataset [72], and the decoder is randomly initialized. We train our models on a node with 8 Tesla V100 GPUs. For pre-training, we adopt the same training hyperparameters (e.g., data augmentation, learning rate, and regularization) used in DeiT [30]. For semantic segmentation experiments, we use random scaling (0.5–2.0), random horizontal flipping, and random cropping as data augmentation methods. AdamW [76] is used as the default optimizer. The batch size is set to 16 for ADE20K and COCO-Stuff, and 8 for Cityscapes. The learning rate is initialized as 0.00006 and the poly-learning rate decay policy is applied during the training. We train the models with 160K iterations for ADE20K and Cityscapes and 80K iterations for COCO-Stuff. We report the mean Intersection over Union (mIOU) to

TABLE III
IMAGE CLASSIFICATION RESULTS ON THE IMAGENET-1K DATASET [72]. 'ACC' REPRESENTS THE TOP-1 ACCURACY. 'FLOPS (G)' IS TESTED UNDER THE INPUT SIZE OF 224 × 224

Model	Params (M)	FLOPs (G)	Acc(%)
MiT-B0 [37] (NeurIPS 2021)	3.7	0.6	70.5
PVTv2-B0 [36] (CVM 2022)	3.4	0.6	70.5
SeaFormer-S [66] (ICLR 2023)	4.1	0.2	**73.3**
SDPT-Tiny	3.2	0.6	71.9
PVT-Tiny [35] (ICCV 2021)	13.2	1.9	75.1
MiT-B1 [37] (NeurIPS 2021)	14.0	2.1	78.7
PVTv2-B1 [36] (CVM 2022)	13.1	2.1	78.7
P2T-Tiny [40] (TPAMI 2022)	11.6	1.8	79.8
SeaFormer-B [66] (ICLR 2023)	8.7	0.3	76.0
SDPT-Small	11.6	1.9	**80.6**
PVT-Small [35] (ICCV 2021)	24.5	3.8	79.8
Swin-T [31] (ICCV 2021)	29.0	4.5	81.3
MiT-B2 [37] (NeurIPS 2021)	25.4	4.0	81.6
PVTv2-B2 [36] (CVM 2022)	25.4	4.0	82.0
ConvNeXt-T [77] (CVPR 2022)	28.6	4.5	82.1
P2T-Small [40] (TPAMI 2022)	24.1	3.7	82.4
SeaFormer-L [66] (ICLR 2023)	14.0	1.2	79.9
SDPT-Base	24.1	3.9	**82.7**

compare the segmentation performance. For a fair comparison, we keep the same training settings as SegFormer [37]. Detailed settings for the three benchmarks, including ADE20K [50], Cityscapes [1], and COCO-Stuff [51], are listed in Table II.

V. RESULTS

We pre-train all encoder models on the ImageNet-1K dataset [72] and evaluate SDPT on ADE20k [50], Cityscapes [1], and COCO-Stuff [51] for semantic segmentation. Ablation analysis is also conducted to show the effect of each component in our method.

A. Encoder Pre-Training on ImageNet

Similar to the previous segmentation methods [27], [37], [58], [81], we pre-train our encoder models on the ImageNet-1K dataset [72]. The corresponding results are summarized in Table III. Our SDPT surpasses the CNN-based approach, ConvNeXt [77], and outperforms the transformer-based methods, such as PVT [35], Swin Transformer [31], PVTv2 [36], MiT (the encoder of SegFormer [37]), and P2T [40]. In contrast to SeaFormer [66], our approach is more powerful in the small and base models, while SeaFormer is more powerful in the tiny model.

B. Comparison With SOTA Methods

1) Comparison With Transformer-Based Methods: In Table IV, we compare SDPT with SOTA transformer-based methods on the ADE20K dataset [50]. The results show that our method performs better than other transformer-based segmentation models. Compared with SegFormer [37], HRFormer [78], MaskFormer [45], SegDeformer [79], Mask2Former [80], and VWFormer [69], the proposed models achieve superior performance in terms of accuracy and efficiency. In particular, SDPT-Base achieves the same segmentation performance as SETR-MLA† [38] while using

TABLE IV
COMPARISON WITH SOTA TRANSFORMER-BASED METHODS ON THE ADE20K DATASET [50]. 'FLOPS (G)' IS TESTED UNDER THE INPUT SIZE OF 512 × 512. † DENOTES MODELS PRE-TRAINED ON IMAGENET-22K

Model	Params (M)	FLOPs (G)	mIoU (%)
SegFormer-B0 [37] (NeurIPS 2021)	3.8	8.4	37.4
TopFormer-S [64] (CVPR 2022)	3.1	1.2	36.1
SeaFormer-S [66] (ICLR 2023)	4.0	1.1	38.1
SCTNet-S [68] (AAAI 2024)	4.7	-	37.7
VWFormer-B0 [69] (ICLR 2024)	3.7	5.8	38.9
SDPT-Tiny	3.6	5.7	**39.4**
SegFormer-B1 [37] (NeurIPS 2021)	13.7	15.9	42.2
HRFormer-S [78] (NeurIPS 2021)	13.5	109.5	44.0
SegDeformer-B1 [79] (ECCV 2022)	14.4	-	44.1
TopFormer-B [64] (CVPR 2022)	5.1	1.8	37.8
SeaFormer-B [66] (ICLR 2023)	8.6	1.8	40.2
SCTNet-B [68] (AAAI 2024)	17.4	-	43.0
VWFormer-B1 [69] (ICLR 2024)	13.7	13.2	43.2
SDPT-Small	11.9	12.7	**46.0**
SegFormer-B2 [37] (NeurIPS 2021)	27.5	62.4	46.5
MaskFormer [45] (NeurIPS 2021)	42	55	46.7
SETR-MLA† [38] (CVPR 2021)	310.6	-	48.6
SegDeformer-B2 [79] (ECCV 2022)	27.6	-	47.5
Mask2Former [80] (CVPR 2022)	47	74	47.7
SeaFormer-L [66] (ICLR 2023)	14.0	6.5	42.7
VWFormer-B2 [69] (ICLR 2024)	27.4	46.6	48.1
SDPT-Base	28.6	35.9	**48.6**

TABLE V
COMPARISON WITH SOTA METHODS ON THE COCO-STUFF DATASET [51]. 'FLOPS (G)' IS TESTED UNDER THE INPUT SIZE OF 512 × 512

Model	Params (M)	FLOPs (G)	mIoU (%)
SegFormer-B0 [37] (NeurIPS 2021)	3.8	8.4	35.6
VWFormer-B0 [69] (ICLR 2024)	3.7	5.8	36.2
SDPT-Tiny	3.6	5.8	**36.9**
SegFormer-B1 [37] (NeurIPS 2021)	13.7	15.9	40.2
HRFormer-S [78] (NeurIPS 2021)	13.5	109.5	37.9
VWFormer-B1 [69] (ICLR 2024)	13.7	13.2	41.5
SDPT-Small	11.9	12.8	**43.0**
SegFormer-B2 [37] (NeurIPS 2021)	27.5	62.4	44.6
HRFormer-B [78] (NeurIPS 2021)	56.2	280.0	42.4
VWFormer-B2 [69] (ICLR 2024)	27.4	46.6	45.2
SDPT-Base	28.6	35.9	**45.6**

fewer parameters (28.6 vs. 310.6). Furthermore, our SDPT outperforms efficient methods such as TopFormer [64], SeaFormer [66], and SCTNet [68].

2) Generalization Ability: To further validate the effectiveness of our SDPT, we conducted experiments on the COCO-Stuff [51] and Cityscapes [1] datasets. Table V presents the results on the COCO-Stuff dataset. Our SDPT-Tiny and SDPT-Small achieve better segmentation performance than other transformer-based methods with fewer parameters and FLOPs. And SDPT-Base significantly reduces computational complexity while retaining better performance. When testing on high-resolution images from the Cityscapes dataset, our method achieves excellent performance, as shown in Table VI. For example, SDPT-Base achieves comparable performance with CMX [70] (the multimodal fusion segmentation method) and VWFormer [69].

3) Comparison With Real-Time Methods: We further compare our SDPT with other SOTA real-time methods on the ADE20K dataset [50]. As shown in Table VII, all the proposed

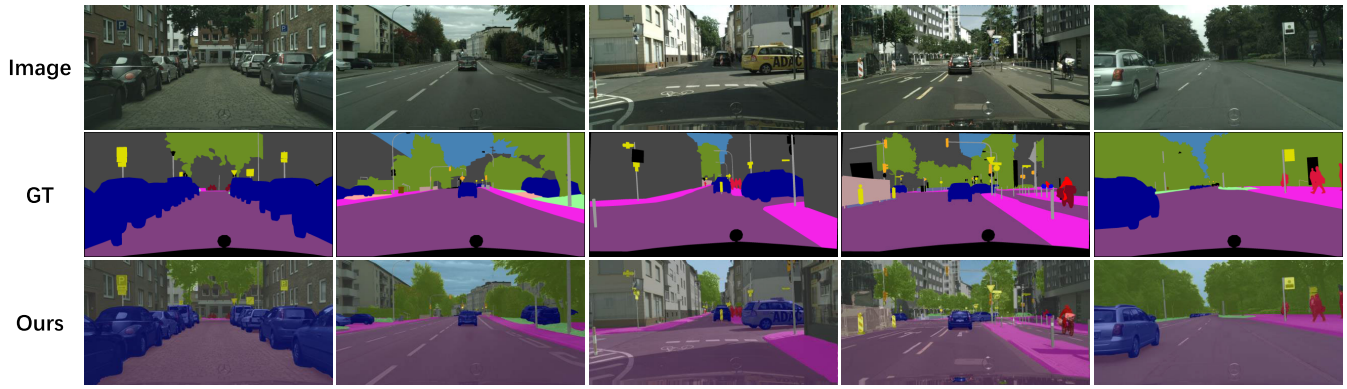


Fig. 5. The visualization showcases prediction segmentation results using the Cityscapes dataset [1]. The original images are displayed in the first row, while ground truths (GT) are depicted in the second row. The predicted results are presented in the last row.

TABLE VI
COMPARISON WITH SOTA METHODS ON THE CITYSCAPES DATASET [1]. 'FLOPS (G)' IS TESTED UNDER THE INPUT SIZE OF 2048 × 1024. † DENOTES MODELS PRE-TRAINED ON IMAGENET-22K

Model	Params (M)	FLOPs (G)	mIoU (%)
SegFormer-B0 [37] (NeurIPS 2021)	3.8	125.5	76.2
TopFormer-B (h) [64] (CVPR 2022)	-	2.7	70.7
SeaFormer-S [66] (ICLR 2023)	-	8.0	76.1
VWFormer-B0 [69] (ICLR 2024)	3.7	-	77.2
SDPT-Tiny	3.6	63.4	**77.3**
SegFormer-B1 [37] (NeurIPS 2021)	13.7	243.7	78.5
HRFormer-S [78] (NeurIPS 2021)	13.5	835.7	80.0
TopFormer-B (f) [64] (CVPR 2022)	-	11.2	75.0
SeaFormer-B [66] (ICLR 2023)	-	13.7	77.7
PIDNet-S [65] (CVPR 2023)	7.6	47.6	78.8
VWFormer-B1 [69] (ICLR 2024)	13.7	-	79.0
SDPT-Small	11.9	131.3	**80.4**
SegFormer-B2 [37] (NeurIPS 2021)	27.5	717.1	81.0
SETR-MLA† [38] (CVPR 2021)	310.6	-	79.3
CMX-B2 [70] (TITS 2023)	-	-	81.6
PIDNet-M [65] (CVPR 2023)	34.4	197.4	80.1
VWFormer-B2 [69] (ICLR 2024)	27.4	-	**81.7**
SDPT-Base	28.5	333.0	81.6

SDPT-Tiny, SDPT-Small, and SDPT-Base achieve real-time performance with 63.8 FPS, 46.8 FPS, and 32.2 FPS on the ADE20K dataset, respectively. The results demonstrate that the proposed method can strike a balance between segmentation performance and efficiency.

4) Comparison With CNN-Based Methods: We compare our SDPT with SOTA CNN-based methods such as FCN [11], EncNet [83], PSPNet [41], CCNet [57], DeepLabV3+ [28], OCRNet [84], and SegNeXt [58] on the ADE20K dataset [50]. The results are summarized in Table VII. Our SDPT surpasses the popular OCRNet (HRNet) [84] and SegNeXt [58]. In addition, our method is significantly faster (FPS) than the majority of CNN-based competitors.

5) Instance Segmentation: To further demonstrate the effectiveness of our SDPT backbone. We conducted experiments on the MS COCO dataset [85] for the instance segmentation task. We integrated the proposed SDPT backbone into the Mask RCNN [12] framework to conduct the experiments. The corresponding results are summarized in Table VIII. Our SDPT outperforms other methods including ViL [86], PVT [35], PVTv2 [36], Swin [31], Twins [87] and P2T [40]. As presented in Fig. 6, we visualize the instance segmentation results by using our trained model. The results show that our method can generate excellent instance-predictions.

6) Results on Mask Predictions: In Fig. 5, we visualize the competitive segmentation results selected from the Cityscapes dataset [1]. The original images are showcased in the first row, followed by the ground truths (GT) depicted in the second row. The predicted results using the proposed model are presented in the last row. It is clear that the proposed SDPT can produce satisfactory segmentation results by single-scale inference.

C. Ablation Study

We conduct ablation experiments on the ADE20K dataset [50]. SDPT-Tiny is used as the baseline model, and 'FLOPs (G)' is tested under the input size of 512 × 512.

1) Effect of Our DPA: We investigate the effect of introducing dimension-pooling attention (DPA) by replacing attention module with different attention mechanisms in the encoder. The results are summarized in Table IX. Our DPA outperforms the linear spatial reduction attention (linear SRA) used in PVTv2 [36] and the convolutional spatial reduction attention (SRA) used in SegFormer [37]. Compared with the original self-attention [32], the proposed DPA achieves comparable performance with significantly lower computational complexity.

2) Importance of Convolutional Stem in Early Stage: To explore the influence of convolutions in the early stages, we evaluate our convolutional stem with non-overlap patch embedding in ViT [29] and overlap patch embedding in SegFormer [37]. As shown in Table X, the results show that early convolutional stem helps transformers learn better, which is consistent with [71]. With this design, our method can bring significant performance improvement.

3) Effect of Each Component in DPA: We investigate the effect of each component in dimension-pooling attention (DPA) by replacing the attention module with different configurations in the encoder. The results are summarized in Table XI. Only the convolution or attention branches achieve relatively poor performance, while combining attention with lightweight depth-wise convolution (DWConv) can lead to performance gains. This is due to the fact that DWConv can provide local detail information, which facilitates image classification and downstream tasks (semantic segmentation).

TABLE VII

COMPARISON WITH SOTA REAL-TIME METHODS AND CNN-BASED METHODS ON THE ADE20K DATASET [50]. 'FLOPS (G)' IS TESTED UNDER THE INPUT SIZE OF 512 × 512

Model	Encoder	Params (M)	FLOPs (G)	FPS	mIoU (%)
Real-time methods:					
FCN [11] (CVPR 2015)	MobileNetV2	9.8	39.6	64.4	19.7
BiSeNetV2 [82] (IJCV 2021)	-	14.8	12.4	75.7	26.8
PSPNet [41] (CVPR 2017)	MobileNetV2	13.7	52.9	57.7	29.6
DeepLabV3+ [28] (ECCV 2018)	MobileNetV2	15.4	69.4	43.1	34.0
SegFormer-B0 [37] (NeurIPS 2021)	MiT-B0	3.8	8.4	50.5	37.4
SegFormer-B1 [37] (NeurIPS 2021)	MiT-B1	13.7	15.9	55.8	42.2
SegNeXt-T [58] (NeurIPS 2022)	SegNeXt-T	4.3	6.6	60.3	41.1
TopFormer-B [64] (CVPR 2022)	TopFormer-B	5.1	1.8	96.2	37.8
AFFormer-B [67] (AAAI 2023)	AFFormer-B	3.0	4.6	49.6	41.8
SeaFormer-B [66] (ICLR 2023)	SeaFormer-B	8.6	1.8	44.5	40.2
SDPT-T (Ours)	SDPT-Tiny	3.6	5.7	63.8	39.4
SDPT-S (Ours)	SDPT-Small	11.9	12.7	46.8	**46.5**
CNN-based methods:					
FCN [11] (CVPR 2015)	ResNet-101	68.6	275.7	14.8	41.4
EncNet [83] (CVPR 2018)	ResNet-101	55.1	218.8	14.9	44.7
PSPNet [41] (CVPR 2017)	ResNet-101	68.1	256.4	15.3	44.4
CCNet [57] (ICCV 2019)	ResNet-101	68.9	278.4	14.1	45.2
DeepLabV3+ [28] (ECCV 2018)	ResNet-101	62.7	255.1	14.1	44.1
OCRNet [84] (ECCV 2020)	HRNet-W48	70.5	164.8	17.0	45.6
SegNeXt-S [58] (NeurIPS 2022)	SegNeXt-S	13.9	15.9	50.3	45.8
SegNeXt-B [58] (NeurIPS 2022)	SegNeXt-B	27.6	34.9	30.1	48.5
SDPT-B (Ours)	SDPT-Base	28.6	35.9	32.2	**48.6**

TABLE VIII

COMPARISON WITH INSTANCE SEGMENTATION RESULTS OF OTHER SOTA METHODS ON THE MS COCO DATASET [85]. 'FLOPS (G)' IS TESTED UNDER THE INPUT SIZE OF 800 × 1280

Backbone	Params (M)	FLOPs (G)	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
R-18 [56] (CVPR 2016)	31.2	209	34.0	54.0	36.7	31.2	51.0	32.7
ViL-Tiny [86] (ICCV 2021)	26.9	223	41.4	63.5	45.0	38.1	60.3	40.8
PVT-Tiny [35] (ICCV 2021)	32.9	223	36.7	59.2	39.3	35.1	56.7	37.3
PVTv2-B1 [36] (CVM 2022)	33.7	227	41.8	64.3	45.9	38.8	61.2	41.6
P2T-Tiny [40] (TPAMI 2022)	31.3	225	43.3	65.7	47.3	39.6	62.5	42.3
SDPT-Small (Ours)	31.3	233	**44.2**	**66.5**	**48.5**	**40.4**	**63.3**	**43.5**
R-50 [56] (CVPR 2016)	44.2	260	38.0	58.6	41.4	34.4	55.1	36.7
PVT-Small [35] (ICCV 2021)	44.1	280	40.4	62.9	43.8	37.8	60.1	40.3
Swin-T [31] (ICCV 2021)	47.8	264	42.2	64.6	46.2	39.1	61.6	42.0
ViL-Small [86] (ICCV 2021)	45.0	310	44.9	67.1	49.3	41.0	64.2	44.1
Twins-SVT-S [87] (NeurIPS 2021)	44.0	252	43.4	66.0	47.3	40.3	63.2	43.4
PVTv2-B2 [36] (CVM 2022)	45.0	285	45.3	67.1	49.6	41.2	64.2	44.4
P2T-Small [40] (TPAMI 2022)	43.7	279	45.5	67.7	49.8	41.4	64.6	44.5
SDPT-Base (Ours)	43.8	278	**45.7**	**67.8**	**50.2**	**41.5**	**64.9**	**44.5**

TABLE IX

THE PERFORMANCE OF DIFFERENT ATTENTION MECHANISMS IN THE ENCODER. WE REPORT TOP-1 ACCURACY AND MIOU ON THE IMAGENET-1K [72] AND ADE20K DATASETS [50], RESPECTIVELY. 'FLOPS (G)' IS TESTED UNDER THE INPUT SIZE OF 224 × 224 AND 512 × 512, RESPECTIVELY

Method	ImageNet-1K			ADE20K		
	Params (M)	FLOPs (G)	Top1 (%)	Params (M)	FLOPs (G)	mIoU (%)
MSA [32]	3.2	2.1	71.2	3.6	45.2	**40.4**
Linear [36]	3.4	0.6	71.7	3.8	5.1	38.3
Convolution [37]	3.7	0.6	70.7	4.0	6.1	39.2
DPA (Ours)	3.2	0.6	**71.9**	3.6	5.7	39.4

Compared to fusing with the add operation, multiplying features from convolution and attention branches performs better.

4) Influence of Decoder Structure: We compare different decoder structures for semantic segmentation. Specifically, we configured our SDPT-Tiny backbone with a pure MLP-based decoder [37], ASPP [27], and a lightweight Hamburger decoder [58], and our decoder variants ((a) and (b), see Fig. 7a and Fig. 7b) for segmentation experiments on the ADE20K dataset [50]. The results are listed in Table XII. It can be seen that SDPT (b) achieves the best performance compared to other decoder structures.

TABLE X
THE PERFORMANCE OF DIFFERENT PATCH EMBEDDING METHODS ON THE ADE20K DATASET [50]

Patch Embedding	Params (M)	FLOPs (G)	Top1	mIoU
Non-Overlap [29]	3.6	5.4	67.1	36.1
Overlap [37]	3.6	5.5	71.5	38.7
Conv-Stem (Ours)	3.6	5.7	**71.9**	**39.4**

TABLE XI
ABLATION STUDIES ON EACH COMPONENT OF DPA ON THE ADE20K DATASET [50]. (A) AND (B) DENOTE ATTN AND CONV ARE FUSED BY USING THE ADD AND MULTIPLY OPERATIONS, RESPECTIVELY

Architecture	Params (M)	FLOPs (G)	Top1	mIoU
Conv branch	3.0	4.7	68.0	36.7
Attn branch	3.6	5.7	70.9	38.4
Attn & Conv (a)	3.6	5.7	71.5	38.6
Attn & Conv (b)	3.6	5.7	**71.9**	**39.4**

TABLE XII
THE PERFORMANCE OF DIFFERENT DECODER STRUCTURES ON THE ADE20K DATASET [50]

Architecture	Params (M)	FLOPs (G)	mIoU (%)
SDPT w/ MLP [37]	3.4	9.0	38.5
SDPT w/ ASPP [27]	18.6	7.8	38.5
SDPT w/ Ham [58]	3.3	5.6	38.3
SDPT (a)	5.7	10.7	39.0
SDPT (b) (Ours)	3.6	5.7	**39.4**
SDPT (b) w/ Stage 1	3.7	13.4	39.2

TABLE XIII
THE PERFORMANCE OF DIFFERENT REFINEMENT MODULES ON THE ADE20K DATASET [50]

Method	Params (M)	FLOPs (G)	mIoU (%)
Integration	3.3	4.9	38.5
DWConv	3.3	4.9	38.5
NonLocal	3.6	14.6	38.4
DPA (Ours)	3.6	5.7	**39.4**

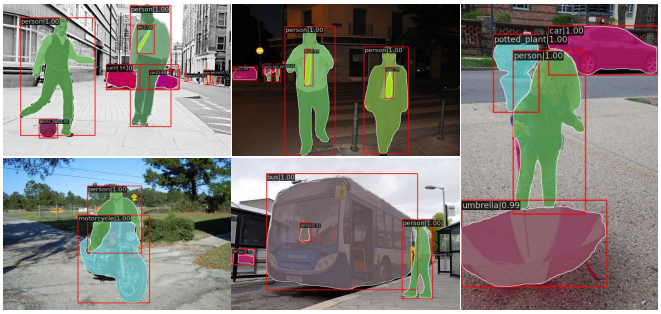
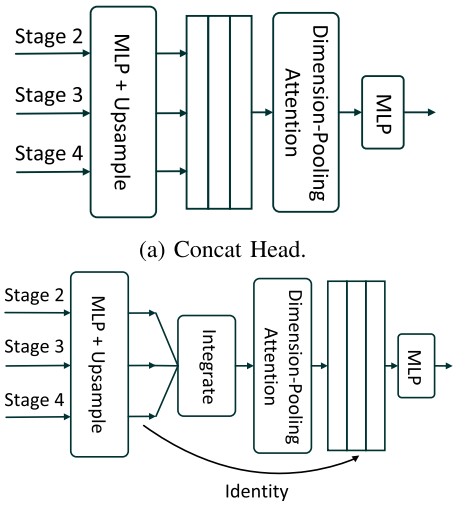


Fig. 6. The visualization of instance segmentation results. The results are generated by using the proposed model on the MS COCO dataset [85].

5) DPA for Decoder: We evaluate the impact of the different refinement modules in the decoder. As shown in Table XIII, our DPA can significantly reduce the computational cost compared to NonLocal [88], while the performance is superior and stable compared to DWConv.



(a) Concat Head.

(b) Balanced Head.

Fig. 7. Different segmentation heads in SDPT: (a) Concat head. The features from the encoder are concatenated and then passed through the DPA module. (b) Balanced head. Balanced features are obtained through averaging operations and then fed into the DPA module.

VI. CONCLUSION

In this paper, we present a simple yet effective encoder-decoder architecture based on dimension-pooling transformers, named SDPT. An efficient transformer encoder is elaborately designed to generate multi-scale features, and a semantic-balanced decoder is introduced to integrate multi-level features for predicting semantic masks. The highlight is that our SDPT performs better than current methods with less computational complexity, leading to a trade-off between accuracy and speed. The experimental results on the ImageNet-1k and MS COCO datasets show that using the proposed SDPT as a backbone can achieve excellent performance. Extensive experiments on the ADE20K, Cityscapes, and COCO-Stuff datasets have demonstrated the effectiveness of our SDPT.Â The limitation is that, despite having only 3.6 million parameters, it is unclear whether our SDPT-Tiny will work well in chip-based edge devices. Furthermore, it is also interesting to study how our approach extends to large-scale models and other vision tasks.

REFERENCES

[1] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[2] G. Phonsa and K. Manu, "A survey: Image segmentation techniques," in *Harmony Search and Nature Inspired Optimization Algorithms: Theory and Applications*. Singapore: Springer, 2019, pp. 1123–1140.
[3] S. A. Hojjatoleslami and J. Kittler, "Region growing: A new approach," *IEEE Trans. Image Process.*, vol. 7, no. 7, pp. 1079–1084, Jul. 1998.
[4] S. Beucher, "The watershed transformation applied to image segmentation," *Scanning Microsc.*, vol. 1992, no. 6, p. 28, 1992.
[5] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[6] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[7] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an image edge detection filter using the Sobel operator," *IEEE J. Solid-State Circuits*, vol. 23, no. 2, pp. 358–367, Apr. 1988.

[8] J. M. Prewitt, "Object enhancement and extraction," *Picture Process. Psychopictorics*, vol. 10, no. 1, pp. 15–19, 1970.

[9] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3523–3542, Jul. 2022.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. MICCAI*, 2015, pp. 234–241.

[11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

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

[13] P. Wang et al., "Understanding convolution for semantic segmentation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1451–1460.

[14] N. Sahu, V. Chamola, and R. R. Rajkumar, "A clustering and image processing approach to unsupervised real-time road segmentation for autonomous vehicles," in *Proc. IEEE Globecom Workshops*, Dec. 2022, pp. 160–165.

[15] H. Cao et al., "Swin-UNet: UNet-like pure transformer for medical image segmentation," in *Proc. ECCVW*, 2022, pp. 205–218.

[16] M. Gruosso, N. Capece, and U. Erra, "Human segmentation in surveillance video with deep learning," *Multimedia Tools Appl.*, vol. 80, no. 1, pp. 1175–1199, Jan. 2021.

[17] X. Yuan, J. Shi, and L. Gu, "A review of deep learning methods for semantic segmentation of remote sensing imagery," *Exp. Syst. Appl.*, vol. 169, May 2021, Art. no. 114417.

[18] L. Tanzi, P. Piazzolla, F. Porpiglia, and E. Vezzetti, "Real-time deep learning semantic segmentation during intra-operative surgery for 3D augmented reality assistance," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 16, no. 9, pp. 1435–1445, Sep. 2021.

[19] S. Ainetter and F. Fraundorfer, "End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from RGB," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 13452–13458.

[20] T. Anand, S. Sinha, M. Mandal, V. Chamola, and F. R. Yu, "AgriSegNet: Deep aerial semantic segmentation framework for IoT-assisted precision agriculture," *IEEE Sensors J.*, vol. 21, no. 16, pp. 17581–17590, Aug. 2021.

[21] A. S. Chakravarthy, S. Sinha, P. Narang, M. Mandal, V. Chamola, and F. R. Yu, "DroneSegNet: Robust aerial semantic segmentation for UAV-based IoT applications," *IEEE Trans. Veh. Technol.*, vol. 71, no. 4, pp. 4277–4286, Apr. 2022.

[22] W. Zhou, J. S. Berrio, S. Worrall, and E. Nebot, "Automated evaluation of semantic segmentation robustness for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1951–1963, May 2020.

[23] G. Chen, H. Cao, J. Conradt, H. Tang, F. Rohrbein, and A. Knoll, "Event-based neuromorphic vision for autonomous driving: A paradigm shift for bio-inspired visual sensing and perception," *IEEE Signal Process. Mag.*, vol. 37, no. 4, pp. 34–49, Jul. 2020.

[24] D. Feng et al., "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1341–1360, Mar. 2021.

[25] K. Muhammad et al., "Vision-based semantic segmentation in scene understanding for autonomous driving: Recent achievements, challenges, and outlooks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 22694–22715, Dec. 2022.

[26] J. Liao, L. Cao, W. Li, Y. Ou, C. Duan, and H. Cao, "Fully-supervised semantic segmentation networks: Exploring the relationship between the segmentation networks learning ability and the number of convolutional layers," in *Proc. 7th Int. Conf. Comput. Commun. (ICCC)*, Dec. 2021, pp. 1685–1692.

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

[28] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder–decoder with atrous separable convolution for semantic image segmentation," in *Proc. ECCV*, 2018, pp. 801–818.

[29] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. ICLR*, 2021, pp. 1–22.

[30] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou, "Training data-efficient image transformers distillation through attention," in *Proc. ICML*, 2021, pp. 10347–10357.

[31] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9992–10002.

[32] A. Vaswani et al., "Attention is all you need," in *Proc. NIPS*, 2017, pp. 1–11.

[33] L. Yuan et al., "Tokens-to-token ViT: Training vision transformers from scratch on ImageNet," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 538–547.

[34] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. ECCV*, 2020, pp. 213–229.

[35] W. Wang et al., "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 548–558.

[36] W. Wang et al., "PVTv2: Improved baselines with pyramid vision transformer," *Comput. Vis. Media*, vol. 8, no. 3, pp. 1–10, 2022.

[37] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Proc. NIPS*, 2021, pp. 12077–12090.

[38] S. Zheng et al., "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6877–6886.

[39] B. Zhang et al., "SegViT: Semantic segmentation with plain vision transformers," in *Proc. NIPS*, 2022, pp. 1–12.

[40] Y.-H. Wu, Y. Liu, X. Zhan, and M.-M. Cheng, "P2T: Pyramid pooling transformer for scene understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 99, pp. 1–12, 2022.

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

[42] J. Fu et al., "Dual attention network for scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3141–3149.

[43] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 565–571.

[44] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 7242–7252.

[45] B. Cheng, A. G. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," in *Proc. NIPS*, 2021, pp. 17864–17875.

[46] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. ECCV*, 2014, pp. 818–833.

[47] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.

[48] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.

[49] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra R-CNN: Towards balanced learning for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 821–830.

[50] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5122–5130.

[51] H. Caesar, J. Uijlings, and V. Ferrari, "COCO-stuff: Thing and stuff classes in context," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1209–1218.

[52] S. U. Lee, S. Y. Chung, and R. H. Park, "A comparative performance study of several global thresholding techniques for segmentation," *Comput. Vis., Graph., Image Process.*, vol. 52, no. 2, pp. 171–190, 1990.

[53] D. Bradley and G. Roth, "Adaptive thresholding using the integral image," *J. Graph. Tools*, vol. 12, no. 2, pp. 13–21, 2007.

[54] Q. Hou, L. Zhang, M.-M. Cheng, and J. Feng, "Strip pooling: Rethinking spatial pooling for scene parsing," in *Proc. CVPR*, 2020, pp. 4003–4012.

[55] K. Yang, J. Zhang, S. Reiß, X. Hu, and R. Stiefelhagen, "Capturing omni-range context for omnidirectional segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 1376–1386.

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

[57] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: Criss-cross attention for semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 603–612.

[58] M.-H. Guo, C.-Z. Lu, Q. Hou, Z. Liu, M.-M. Cheng, and S.-M. Hu, "SegNeXt: Rethinking convolutional attention design for semantic segmentation," in *Proc. NIPS*, 2022, pp. 1140–1156.

[59] S. Ren, D. Zhou, S. He, J. Feng, and X. Wang, "Shunted self-attention via multi-scale token aggregation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10843–10852.

[60] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," in *Proc. ECCV*, 2018, pp. 405–420.

[61] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. ECCV*, 2018, pp. 325–341.

[62] M. Fan et al., "Rethinking BiSeNet for real-time semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 9711–9720.

[63] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 12159–12168.

[64] W. Zhang et al., "TopFormer: Token pyramid transformer for mobile semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12073–12083.

[65] J. Xu, Z. Xiong, and S. P. Bhattacharyya, "PIDNet: A real-time semantic segmentation network inspired by PID controllers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 19529–19539.

[66] Q. Wan, Z. Huang, J. Lu, G. Yu, and L. Zhang, "Seaformer: Squeeze-enhanced axial transformer for mobile semantic segmentation," in *Proc. ICLR*, 2023, pp. 1–34.

[67] D. Bo, W. Pichao, and F. Wang, "Head-free lightweight semantic segmentation with linear transformer," in *Proc. AAAI*, 2023, pp. 516–524.

[68] Z. Xu, D. Wu, C. Yu, X. Chu, N. Sang, and C. Gao, "SCTNet: Single-branch CNN with transformer semantic information for real-time segmentation," in *Proc. AAAI*, 2024, pp. 6378–6386.

[69] H. Yan, M. Wu, and C. Zhang, "Multi-scale representations by varing window attention for semantic segmentation," in *Proc. ICLR*, 2024, pp. 1–17.

[70] J. Zhang, H. Liu, K. Yang, X. Hu, R. Liu, and R. Stiefelhagen, "CMX: Cross-modal fusion for RGB-X semantic segmentation with transformers," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 14679–14694, Dec. 2023.

[71] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Dollàr, and R. B. Girshick, "Early convolutions help transformers see better," in *Proc. NIPS*, 2021, pp. 30392–30400.

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

[73] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. NIPS*, 2019, pp. 1–13.

[74] R. Wightman, "PyTorch image models," *GitHub, GitHub Repository*, 2019, doi: 10.5281/zenodo.4414861. [Online]. Available: https://github.com/rwightman/pytorch-image-models

[75] MMSegmentation Contributors. (2020). *MMSegmentation: OpenMMLab Semantic Segmentation Toolbox and Benchmark*. [Online]. Available: https://github.com/open-mmlab/mmsegmentation

[76] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. ICLR*, 2019.

[77] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 11966–11976.

[78] Y. Yuan et al., "HRFormer: High-resolution vision transformer for dense predict," in *Proc. NIPS*, 2021, pp. 7281–7293.

[79] B. Shi et al., "A transformer-based decoder for semantic segmentation with multi-level context mining," in *Proc. ECCV*, 2022, pp. 624–639.

[80] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 1280–1289.

[81] J. Gu et al., "Multi-scale high-resolution vision transformer for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12084–12093.

[82] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "BiSeNet v2: Bilateral network with guided aggregation for real-time semantic segmentation," *Int. J. Comput. Vis.*, vol. 129, no. 11, pp. 3051–3068, Nov. 2021.

[83] H. Zhang et al., "Context encoding for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7151–7160.

[84] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *Proc. ECCV*, 2020, pp. 173–190.

[85] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. ECCV*, 2014, pp. 740–755.

[86] P. Zhang et al., "Multi-scale vision longformer: A new vision transformer for high-resolution image encoding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 2978–2988.

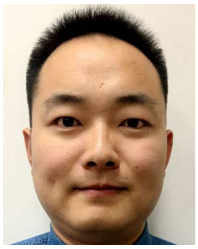
[87] X. Chu et al., "Twins: Revisiting the design of spatial attention in vision transformers," in *Proc. NIPS*, 2021, pp. 9355–9366.

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

Hu Cao received the Ph.D. degree in computer engineering from the Technical University of Munich (TUM) in 2023. He is currently a Research Associate with the Chair of Robotics, Artificial Intelligence, and Real-Time Systems, TUM, where he has also been a Research Assistant, since October 2019. During his studies, he stayed abroad with ETH Zürich and The University of Hong Kong (HKU), where he was involved in developing algorithms for dense prediction (classification, detection, and segmentation), autonomous driving, and robotic grasping. His current research interests include robotics, machine learning, computer vision, event-based vision, and embodied AI.

Guang Chen received the B.S. and M.Eng. degrees in mechanical engineering from Hunan University, China, and the Ph.D. degree from the Faculty of Informatics, Technical University of Munich, Germany, in 2016. He is a Professor with Tongji University and a Senior Research Associate (guest) with the Technical University of Munich. He is leading the Robotics and Embodied Artificial Intelligence Laboratory, Tongji University. He was a Research Scientist with Fortiss GmbH, a research institute of the Technical University of Munich, from 2012 to 2016; and a Senior Researcher with the Chair of Robotics, Artificial Intelligence, and Real-Time Systems, Technical University of Munich, from 2016 to 2018. His research interests include 3-D vision, embodied artificial intelligence, intelligent robotics, and autonomous driving. He was awarded as the Tongji Hundred Talent Research Professor 2018, the Shanghai Rising Star 2021, the Shanghai S&T 35U35 2021, and the National Distinguished Young Talents 2023. He is the Program Chair of IEEE MFI 2022. He serves as an associate editor for several international journals.

Hengshuang Zhao (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, supervised by Prof. Jiaya Jia. He is currently an Assistant Professor with the Department of Computer Science, The University of Hong Kong. Before that, he was a Post-Doctoral Researcher with the Computer Science and Artificial Intelligence Laboratory (CSAIL), MIT, working with Prof. Antonio Torralba; and the Torr Vision Group, Department of Engineering Science, University of Oxford, working with Prof. Philip Torr. His general research interests cover the broad areas of computer vision and machine learning.



Dongsheng Jiang received the Ph.D. degree in biomedical engineering from Fudan University. He is currently a Senior Researcher with Huawei Inc. His research interests include computer vision and medical image analysis.

Xiaopeng Zhang received the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 2017. He is currently a Senior Researcher with CLOUD&AI, Huawei Technologies. He was a Research Fellow with the Department of ECE, National University of Singapore, from 2017 to 2019; and a Visiting Researcher with the Department of Computer Science, The University of Texas at San Antonio, San Antonio, TX, USA, from 2015 to 2016. His current research interests include object recognition, weakly supervised detection, and self-supervised learning.

Qi Tian (Fellow, IEEE) received the Ph.D. degree in electronics and communication engineering from the University of Illinois at Urbana–Champaign (UIUC) in 2002. He is currently the Chief Scientist in artificial intelligence with Huawei Cloud & AI. He was the Chief Scientist in computer vision with the Huawei Noah's Ark Laboratory, from 2018 to 2020. Before joined Huawei, he was a Full Professor with the Department of Computer Science, The University of Texas at San Antonio (UTSA), from 2002 to 2019. He was listed in the Top 10 of the 2016 Most Influential Scholars in Multimedia by Aminer.org. He is an academician of the International Eurasian Academy of Sciences (IEAS) in 2021. He received the 2017 UTSA President Distinguished Award for Research Achievement, the 2016 UTSA Innovation Award in the first category, the 2014 Research Achievement Awards from the College of Science at UTSA, and the 2010 Google Faculty Research Award. He has served as a Founding Member for ICMR (2009–2014) and ACM MM (2009–2012); an International Steering Committee Member for ACM MIR (2006–2010), ACM ICIMCS (2013), ICME (2006 and 2009), PCM (2012), and IEEE International Symposium on Multimedia (2011); and the Chair for ACM Multimedia 2015. He is an Associate Editor of IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, *ACM Transactions on Multimedia Computing, Communications, and Applications*, *Multimedia Systems*, and *Journal of Machine Vision and Applications*.

Alois Knoll (Fellow, IEEE) received the diploma (M.Sc.) degree in electrical/communications engineering from the University of Stuttgart, Germany, in 1985, and the Ph.D. degree (summa cum laude) in computer science from the Technical University of Berlin (TU Berlin), Germany, in 1988. He was on the Faculty of the Computer Science Department, TU Berlin, until 1993. He joined the University of Bielefeld as a Full Professor and the Director of the Research Group of Technical Informatics in 2001. Since 2001, he has been a Professor with the Department of Informatics, TU München (TUM). He was also on the board of directors of the Central Institute of Medical Technology, TUM (IMETUM). From 2004 to 2006, he was the Executive Director of the Institute of Computer Science, TUM. His research interests include cognitive, medical, and sensor-based robotics; multi-agent systems; data fusion; adaptive systems; multimedia information retrieval; model-driven development of embedded systems, with applications to automotive software and electric transportation; and simulation systems for robotics and traffic.