A Transformer-based Decoder for Semantic Segmentation with Multi-level Context Mining

Bowen Shi[1†], Dongsheng Jiang[2†], Xiaopeng Zhang[2], Han Li[1], Wenrui Dai[1], Junni Zou[1], Hongkai Xiong[1] and Qi Tian[2*]

[1] Shanghai Jiao Tong University
{sjtu_shibowen,qingshi9974,daiwenrui,zoujunni,xionghongkai}@sjtu.edu.cn
[2] Huawei Cloud EI
dongsheng_jiang@outlook.com, zxphistory@gmail.com, tian.qi1@huawei.com

Abstract. Transformers have recently shown superior performance than CNN on semantic segmentation. However, previous works mostly focus on the deliberate design of the encoder, while seldom considering the decoder part. In this paper, we find that a light weighted decoder counts for segmentation, and propose a pure transformer-based segmentation decoder, named SegDeformer, to seamlessly incorporate into current varied transformer-based encoders. The highlight is that SegDeformer is able to conveniently utilize the tokenized input and the attention mechanism of the transformer for effective context mining. This is achieved by two key component designs, *i.e.,* the internal and external context mining modules. The former is equipped with internal attention within an image to better capture global-local context, while the latter introduces external tokens from other images to enhance current representation. To enable SegDeformer in a scalable way, we further provide performance/efficiency optimization modules for flexible deployment. Experiments on widely used benchmarks ADE20K, COCO-Stuff and Cityscapes and different transformer encoders (*e.g.*, ViT, MiT and Swin) demonstrate that SegDeformer can bring consistent performance gains. Code is available at https://github.com/lygsbw/segdeformer.

1 Introduction

Semantic segmentation is a fundamental computer vision task and has attracted broad interest for its wide applications. Current solutions for semantic segmentation usually follow an encoder-decoder architecture proposed in FCN [22], which enables efficient transferring from the classification pretrained backbone for segmentation via per-pixel classification. Recently, great performance benefits have been achieved in image classification by replacing Convolutional Neural Networks (CNN) with transformer-based networks [11,21,12], and many works [37,30,24] have pushed the segmentation performance by adapting these structures.

However, previous transformer-based methods mostly focus on designing the encoder, while ignoring the decoder part. These frameworks are usually equipped

[*] Corresponding author. [†] Equal contribution.

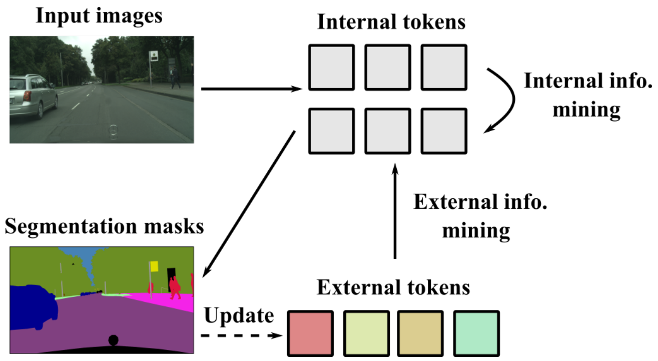

Fig. 1. The internal tokens from the current image and the introduced leanable external tokens can easily interacted with each other for multi-level context mining in a transformer-based framework.

with a cumbersome transformer encoder for better performance, which inevitably suffers high computational costs. Considering this issue, there have been several context modeling designs at the decoder for CNN-based models. Among them, ASPP [4] and PPM [35] enlarge the spatial scale of contexts to utilize multi-scale contexts. OCNet [32] and CCNet [15] augment the representation of a position by aggregating the representations of its contextual positions. The highlight is that, the decoder part has stronger feature integration capability compared to the encoder. Since there exist co-occurrent visual patterns among pixels, designing context modeling schemes in the decoder is a more direct and effective practice. Considering that the context introduced by these works all comes from pixels within a single image, some recent works [17,27] introduce cross-image context mining and validate that it helps improve feature representation.

Different from previous complex context mining schemes, this paper pursues a simple but effective design for the decoder based on a light weighted transformer module. *The motivation is that we find transformer enjoys several advantages which are suitable for context mining.* First, the attention mechanism is very conducive to contextual information interaction since it has a global receptive field. Second, the flexible tokenized input of the transformer makes it convenient to model cross-image information. As shown in Fig. 1, by leveraging transformer, we can conveniently integrate context from different levels.

Based on these observations, this paper proposes a pure transformer-based decoder, named SegDeformer, for semantic segmentation. SegDeformer considers using two different kinds of context modules to help model pixel-level representation. One is an internal context mining module to capture global-local context within an image, which contains only one internal attention layer but the straight design is surprisingly effective. Another is an external context mining module for cross-image context interaction. We introduce additional learnable tokens in this module which take charge of summarizing information from other images. We also decouple and impose additional constraints on these tokens to make each token category-specific and ensure that the extracted information is helpful. Compared to [17] and [27] which utilize huge memory space to store the external information, our external tokens are more flexible and can bring more hierarchical features. Benefiting from the transformer structure, internal tokens and external tokens can easily interact through several external attention layers with little computation burden.

SegDeformer can be combined with other optimization modules, such as multi-stage feature fusion modules and efficient self-attention operation, for further performance/efficiency trade off. SegDeformer is also applicable to different kinds of transformer encoders, *e.g.*, ViT [11], MiT [30] and Swin [21], and bring consistent performance gains. To demonstrate the power of SegDeformer, we conduct massive experiments on three widely used segmentation benchmarks, ADE20K, COCO-Stuff and Cityscapes, and experimental results demonstrate its superior performance.

In a nutshell, this paper makes the following contributions:

- We propose a novel transformer-based decoder for semantic context mining, which is applicable to different encoders with varied structures.
- We design simple but effective internal and external context mining modules in the decoder for different levels of feature augmentation.
- We propose optimization techniques for further expansion and analysis the effect of SegDeformer on segmentation benchmarks.

2 Related Work

Transformer-based Semantic Segmentation. Great performance breakthroughs have been achieved in semantic segmentation since the introduction of transformers [26,11,21] into computer vision tasks. Among the transformer-based methods, some works [21,10] directly transfer the transformer encoder designed for classification into semantic segmentation by fine-tuning together with the segmentation decoders [18,29]. Recent works [30,24,6,16] consider to design the overall segmentation framework for better adaptation. Among them, SegFormer [30] adopts a hierarchical encoder design for fine-to-coarse feature extraction as well as a light-weighted decoder design for efficient prediction. Segmenter [24] adds additional class-related tokens that aggregate embeddings of image patches for predicting class labels. MaskFormer [6] utilizes mask classification that predicts the class labels for binary masks related to regions or segments rather than focusing on each pixel. SeMask [16] introduces semantic attention layers into the encoder to improve the ability to encapsulate semantic information in features. Different from these methods, we pay more attention to the decoder and utilize the transformer for context interactions while ensuring our design can adapt to different encoders.

Context Scheme. It is efficient and effective to aggregate contextual information for boosting semantic segmentation. ASPP [2,3,4] and PPM [35] exploit multi-scale contexts by introducing pyramid pooling representations and parallel dilated convolutions, respectively. Recently, contextual information is further aggregated to augment the pixel representation using the spatial positions [36], channels [13] and objects [32] attention mechanisms, and these strategies are then enhanced by using the non-local operation [28] and criss-cross attention [15] to exploit the long-term dependency and criss-cross path. Furthermore, pixels are grouped in [31,33,19] to consider the relations within different object regions and augment the features with the group representation. Considering that these methods leverage the contexts from the same (current) image, MCIBI [17] improves the pixel representations with cross-image information stored in the memory bank. In this work, we consider both intra- and extra-image context mining by exploiting the input and interaction schemes of the transformer.

Discussion. Our design is inspired by MCIBI [17], which also introduces cross-image information for context mining. The main difference is that we bring the benefits of the transformer into the decoder design, while MCIBI is still limited to the CNN framework. Besides, we introduce learnable external tokens, which are proved useful in the following section to model more hierarchical cross-image information and achieve better performance than the use of memory bank in MCIBI. Segmenter [24] also introduces external class-map tokens, whereas they only use the similarity between these tokens and the internal tokens of the image for class prediction. However, we experimentally demonstrate that their structures are difficult to transfer. Our SegDeformer also goes one step further in the usage of external tokens, *i.e.*, we use external tokens serve only as key for feature augmentation and enable more specialized designs for external tokens. More details will be introduced in the following.

3 Methods

3.1 Overall Architecture

An overview of our architecture is shown in Fig. 2. Given an $H \times W \times 3$ input image, we first divide it into patches and pass these patches to the vision transformer encoder to obtain multi-stage feature map $F_i \in \mathbb{R}^{C_i \times H_i \times W_i}$, where $i \in \{1, 2, 3, 4\}$. Vision transformer encoder can benefit from our architecture regardless of their kinds, so it can be both flat structures like ViT, where the feature maps pass through each stage are all at $\frac{1}{16}$ of the original image resolution, and deep-narrow structures like MiT and Swin, where the feature maps are at $\{\frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}\}$ of the original image resolution. Similarly, the channel dimension of the feature maps increases with the deepening of the deep-narrow structures but keeps the same in the flat structures.

After obtaining these multi-stage features, we then pass them to our proposed SegDeformer to augment the representations and predict the final segmentation mask M of size $H \times W \times N_{cls}$, where N_{cls} is the number of categories. The

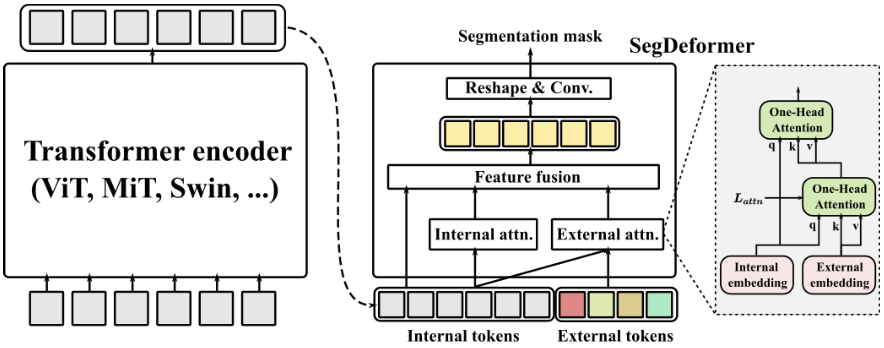

Fig. 2. The overall architecture. SegDeformer contains two key components: internal attention and external attention, for internal and external context mining. It is a general decoder that is applicable to different kinds of transformer encoders.

proposed SegDeformer consists of the following main steps. First, following SegFormer [30], basic MLP layers and optional up-sampling operations are used to unify the channel dimension and feature scale of multi-stage features:

$$\begin{aligned}
\hat{F}_i &= \text{Linear}\left(C_i, C\right)\left(F_i\right), \forall i, \\
\hat{F}_i &= \text{Upsample}\left(H_1 \times W_1\right)\left(\hat{F}_i\right), \forall i, \\
F &= \text{Linear}(4C, C)\left(\text{Concat}\left(\hat{F}_i\right)\right), \forall i,
\end{aligned} \tag{1}$$

where $\text{Linear}(C_{in}, C_{out})(\cdot)$ denotes a linear layer and C_{in} and C_{out} are input and output vector dimensions respectively and F denotes the fused feature. Then, we flatten F back to the internal token sequence X_{inter} with size $N \times C$, where $N = H_1 \times W_1$ denotes the length of X_{inter}, and conduct internal and external context mining as follows:

$$\begin{aligned}
Y_{inter} &= \mathcal{M}_{inter}(X_{inter}), \\
Y_{exter} &= \mathcal{M}_{exter}(X_{inter}, X_{exter}),
\end{aligned} \tag{2}$$

where $\mathcal{M}_{inter}$ and $\mathcal{M}_{exter}$ denotes the internal context mining module and external context mining module, respectively, and Y_{inter} and Y_{exter} are their outputs. $X_{exter} \in \mathbb{R}^{N_{cls} \times C}$ is a learnable external token sequence which will be introduced in the following. Next, the augmented feature $F_{aug} \in \mathbb{R}^{H_1 \times W_1 \times C}$ is obtained by a feature fusion operation $\mathcal{F}$ and a reshape operation:

$$F_{aug} = \text{Reshape}(\mathcal{F}(X_{inter}, Y_{inter}, Y_{exter})), \tag{3}$$

where $\mathcal{F}$ is simply set to an element-wise adding operation in our architecture. Finally, the predicted segmentation mask M is obtained by:

$$M = \text{Upsample}(H \times W)(\mathcal{H}(F_{aug})), \tag{4}$$

where $\mathcal{H}$ denotes the classification head.

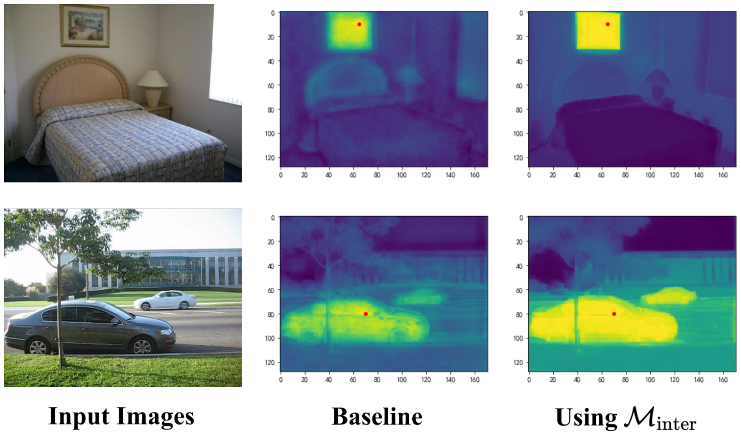

Fig. 3. Feature consistency visualization. The feature consistency of a given pixel (red dots in images) is calculated by the similarity between its feature and the features of other pixels.

3.2 Internal Context Mining

Although the highest layers of transformer architecture are proved to have global respective field [30], we observe that the following multi-stage feature unification operation (Eq. 1) still brings feature confusion and leads to noncontinuous and incorrect mask prediction. We think this is because attention at different stages usually focuses on different contents, some aggregated information may not be suitable for segmentation and requires re-integration. Besides, for transformer encoders with deep-narrow designs, features are organized in a fine-to-coarse way, and regions represented by tokens at different stages have scale differences.

The internal context mining module $\mathcal{M}_{inter}$ with global aggregation capabilities is therefore designed to *reintegrate information from other related pixels. Such that confused pixels can aggregate relevant high-quality information from other pixels*, and improve the representation. We do not adopt complicated designs for $\mathcal{M}_{inter}$ because we empirically find only a one-head self-attention operation is enough to solve the feature confusion problem, as shown in Fig. 3. For computing self-attention in $\mathcal{M}_{inter}$, X_{inter} is first transformed into $Q_{inter}, K_{inter}, V_{inter}$ with the same dimensions $N \times C$ by projecting. Then Y_{inter} is computed by:

$$Y_{inter} = \text{SoftMax}\left(\frac{Q_{inter}K_{inter}^{T}}{\sqrt{C}}\right)V_{inter}. \tag{5}$$

3.3 External Context Mining

Besides the internal information, the contextual information from other images can also enrich features and benefit globally consistent representation across images. This section elaborates the details of our external context mining module

$\mathcal{M}_{exter}$, which uses cross-image information to augment features and is complimentary to $\mathcal{M}_{inter}$. *We hope to convey that the tokenized input used in the transformer is flexible and easily expandable to carry cross-image information.*

Adding external tokens to the decoder. As shown in Fig. 2, in addition to the internal token sequence X_{inter}, which represents the intrinsic information of the current image, we further introduce several external tokens, which are responsible for bring cross-image information to the current image. The external token sequence contains N_{cls} tokens, and each token aggregates information with different meanings. $\mathcal{M}_{exter}$ is composed of two one-head cross-attention operations, which is performed between X_{inter} and X_{exter}. Specifically, we first transform X_{inter} into Q_{mid} and transform X_{exter} into K_{mid} and V_{mid}, then the mid-level feature X_{mid} is obtained by:

$$\begin{aligned} Attn_{mid} &= \frac{Q_{mid}K_{mid}^T}{\sqrt{C}}, \\ X_{mid} &= \text{SoftMax}(Attn_{mid})V_{mid}. \end{aligned} \tag{6}$$

Then we project X_{inter} again to Q_{exter} and project X_{mid} to K_{exter} and V_{exter}, and the augmented feature Y_{exter} is finally obtained by:

$$Y_{exter} = \text{SoftMax}\left(\frac{Q_{exter}K_{exter}^T}{\sqrt{C}}\right)V_{exter}. \tag{7}$$

Adding constraint to external tokens. As part of the network parameters, X_{exter} can continuously update itself along with the learning process. However, we find that adding additional constraints to X_{exter} to make each token category-specific can bring better performance. The category specialization can be achieved by applying additional cross-entropy loss on the mid output M^{mid} arising from $Attn_{mid} \in \mathbb{R}^{N \times N_{cls}}$:

$$\begin{aligned} M^{mid} &= \text{Upsample}(H \times W)(\text{Reshape}(Attn_{mid})), \\ \mathcal{L}_{attn} &= \frac{1}{H \times W}\sum_{i,j}\mathcal{L}_{ce}\left(M^{mid}_{[i,j,*]}, \S\left(\mathcal{GT}_{[ij]}\right)\right), \end{aligned} \tag{8}$$

Here, § denotes for converting the ground truth class label stored in $\mathcal{GT}$ into a one-hot format, $\sum_{i,j}$ denotes that the summation is carried out over all the pixels of the $\mathcal{GT}$, and $\mathcal{L}_{ce}$ is the cross-entropy loss. Similarly, we can also get the final segmentation loss $\mathcal{L}_{seg}$ using the mask prediction M:

$$\begin{aligned} \mathcal{L}_{seg} &= \frac{1}{H \times W}\sum_{i,j}\mathcal{L}_{ce}\left(M_{[i,j,*]}, \S\left(\mathcal{GT}_{[ij]}\right)\right), \\ \mathcal{L}_{total} &= \mathcal{L}_{seg} + \alpha\mathcal{L}_{attn}, \end{aligned} \tag{9}$$

where α is the hyper-parameters to balance the losses. We empirically set $\alpha = 0.4$ by default.

Decoupling external tokens. In the above basic framework, X_{exter} is responsible for both information interaction and category prediction, which increases the difficulty of its ability to compress information. We find that decoupling X_{exter} into two parts that take the two responsibilities separately can further enhance expressiveness. So in practice, we first enlarge the dimension of X_{exter} to $\hat{X}_{exter}$ with size $N \times 2C$. Then, for one external token $\hat{x}_{exter} \in \hat{X}_{exter}$ with size $2C$, we decouple it into two parts:

$$\hat{x}_{exter} = [\hat{x}_{info}, \hat{x}_{class}], \tag{10}$$

where $\hat{x}_{info} \in \mathbb{R}^C$ takes charge of exchanging information and $\hat{x}_{class} \in \mathbb{R}^C$ is used for mid-level prediction. The decoupling of responsibilities can be easily achieved by substituting K_{mid} and V_{mid} in Eq. 6 with the projection of $\hat{X}_{class}$ and $\hat{X}_{info}$, respectively.

3.4 Optimization Modules

SegDeformer can be seamlessly integrated with other modules for further adaptation and expansion. We list part of the optimization techniques we used in this section, *i.e.*, multi-stage feature fusion and efficient self-attention.

Multi-stage feature fusion. SegDeformer mainly utilizes the attention mechanism to augment features while only using basic MLP (Eq. 1) for multi-stage feature fusion. Some other multi-level feature aggregation techniques, *e.g.*, Semantic-FPN [18], UperNet [29], and FAPN [14], can be used in conjunction with our method and bring further performance gains. Following Swin [21], We additionally introduce UperNet in some cases to pursue better feature representation.

Efficient self-attention. Despite SegDeformer can make good use of the characteristics of the transformer, the computational cost of the attention operations, mainly coming from Eq. 5 and 7, makes it unable to adapt to some real-time structures. In this case, we can introduce efficient self-attention to reduce the calculation amount. Denote the reduction ratio as R, we can use a $R \times R$ convolution with stride R to reduce the scale of K and V in Eq. 5 and 7. As a result, the complexity of the self-attention is reduced from $O\left(N^2\right)$ to $O\left(\frac{N^2}{R}\right)$.

4 Experiments

4.1 Experimental Settings

Datasets. We conduct experiments on three wildly used datasets, namely:

- **ADE20K** [9] is a scene parsing dataset. It contains around 25K images spanning 150 semantic categories, of which 20K for training, 2K for validation, and another 3K for testing.

Table 1. Benchmark results on ADE20K (val). † means using UperNet. ‡ means using ImageNet-22K for pretraining. SeMask-L in the encoder is short for SeMask Swin-L.

Method	Encoder	Crop Size	Params↓	mIoU↑	MS mIoU↑
FCN [22]	ResNet101	512 × 512	68.59	39.91	41.40
EncNet [34]	ResNet101	512 × 512	55.05	42.61	44.01
PSPNet [35]	ResNet101	512 × 512	68.07	44.39	45.35
CCNet [15]	ResNet101	512 × 512	68.92	43.71	45.04
DeeplabV3+ [5]	ResNet101	512 × 512	62.68	45.47	46.35
Deit-B† [25]	Deit-B	512 × 512	120.57	45.36	47.16
DPT [23]	ViT-B	512 × 512	109.71	46.97	48.34
SETR-PUP [37]	ViT-L	512 × 512	317.29	48.24	49.99
Twins† [7]	SVT-L	512 × 512	132.78	49.65	50.63
SegFormer-B1 [30]	MiT-B1	512 × 512	13.72	40.97	42.54
SegFormer-B5 [30]	MiT-B5	512 × 512	82.01	49.13	50.22
Swin-L†‡ [21]	Swin-L	512 × 512	233.96	51.61	52.98
Swin-L‡ [21]	Swin-L	640 × 640	203.65	50.85	52.95
SeMask-L‡[16]	SeMask-L	640 × 640	211.79	51.89	53.52
Swin-L†‡ [21]	Swin-L	640 × 640	233.96	52.09	53.47
Deit-B† + SegDeformer	Deit-B	512 × 512	122.96	46.07	47.94
SegFormer-B1 + SegDeformer	MiT-B1	512 × 512	14.35	44.05	45.98
SegFormer-B5 + SegDeformer	MiT-B5	512 × 512	82.65	50.32	51.29
Swin-L†‡ + SegDeformer	Swin-L	512 × 512	236.35	52.77	53.90
Swin-L†‡ + SegDeformer	Swin-L	640 × 640	236.35	**53.12**	**54.13**

- **COCO-Stuff** [1] is a large scale dataset, which includes 118K training images and 5K validation images from COCO 2017 [20], which contains annotations for 80 object classes and 91 stuff classes.
- **Cityscapes** [8] is an urban scene dataset which contains 5,000 finely annotated images, with 2,975 for training, 500 for validation and 1,524 for testing, respectively. It contains 19 categories, such as person, sky and car *etc.*

Implementation details. We train our model with 8 Tesla V100 using the *mmsegmentation*[3] codebase. Unless specified, the encoder is pretrained on Imagenet-1K dataset and the decoder is randomly initialized. Most training and evaluation settings follow [30]. Specifically, during training, we randomly crop the training images to 512 × 512 for ADE20K and COCO-Stuff and to 1024 × 1024 for Cityscapes. Other data augmentation strategies follow [30]. We train the models for 160K iterations and use a batch size of 16 for ADE20K and COCO-Stuff and 8 for Cityscapes. We use AdamW optimizer and the learning rate is set to an initial value of $6e-5$ with "poly" LR schedule. During the evaluation, we keep the

[3] https://github.com/open-mmlab/mmsegmentation

Table 2. Benchmark results on COCO-Stuff. † means using UperNet. ‡ means using ImageNet-22K for pretraining.

Method	Encoder	mIoU↑
Deit-B† [25]	Deit-B	45.68
SegFormer-B5 [30]	MiT-B5	46.71
Swin-L†‡ [21]	Swin-L	49.05
Deit-B† + SegDeformer	Deit-B	46.02
SegFormer-B5 + SegDeformer	MiT-B5	47.51
Swin-L†‡ + SegDeformer	Swin-L	**49.51**

Table 3. Benchmark results on Cityscapes (val). † means using UperNet. ‡ means using ImageNet-22K for pretraining. ⋆ means using efficient self-attention.

Method	Encoder	mIoU↑
Deit-B† [25]	Deit-B	79.09
SegFormer-B5 [30]	MiT-B5	82.07
Swin-L†‡ [21]	Swin-L	82.80
Deit-B† + SegDeformer	Deit-B	80.10
SegFormer-B5⋆ + SegDeformer	MiT-B5	82.46
Swin-L†‡⋆ + SegDeformer	Swin-L	**83.52**

aspect ratio and rescale the short side of the image to training cropped size. For COCO-Stuff and Cityscapes, we additionally conduct inference using the sliding window test. Following the standard, we use mean Intersection-over-Union (mIoU) averaged over all classes for evaluation. *It should be noted that for fair comparisons, the results reported following are all based on mmsegmentation.*

4.2 Main Results

This section compares our results with other methoss on ADE20K, COCO-Stuff and Cityscapes, as well as the qualitative results on Cityscapes.

Results on ADE20K. Tab. 1 summarizes our results for ADE20K. The top part of the table reports some CNN-based methods and the middle includes some transformer-based methods, which achieve higher performance and are served as our primary research objective. At the bottom, we report the results of our SegDeformer with different encoders. As shown, Segdocoder achieves 43.88%, 46.07%, 50.32% and 52.77% mIoU based on MiT-B1, Deit-B, MiT-B5 and Swin-L, respectively, which is 2.91%, 0.71%, 1.19% and 1.16% better than corresponding baselines with only a small increase in the number of parameters. We also conducted multi-scale inference following standard practice [21], and the performance gains are consistent, which are 3.44% for MiT-B1, 0.78% for Deit-B,

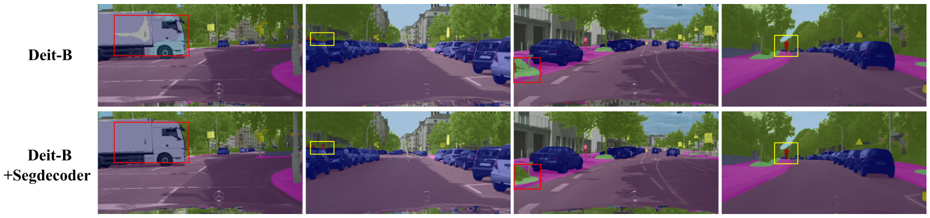

Fig. 4. Visual comparisons between Deit-B and Deit-B + SegDeformer on Cityscapes.

1.07% for MiT-B5 and 0.92% for Swin-L. Besides, SegDeformer can improve the result when using larger input for Swin-L, from 52.09% to 53.12%. A recent work, SeMask [16], achieves close performance growth when using Swin-L with FPN as the baseline. Its design requires adding attention modules to every encoder layer, while our decoder design only introduces a small number of external tokens with three attention operations, which is simpler and more effective.

Results on COCO-Stuff. We then evaluate SegDeformer on the COCO-Stuff dataset. Since *mmsegmentation* does not provide results on COCO-Stuff, we reproduce the Deit-B, Mit-B5 and Swin-L baseline for fair comparisons. The results shown in Tab. 2 are still positive. We achieve 0.34% gains for Deit-B, 0.80% gains for MiT-B5 and 0.46% gains for Swin-L.

Results on Cityscapes. We also report benchmark results on Cityscapes. To meet the larger input size, we replace self-attention with efficient self-attention when using MiT-B5 and Swin-L. As shown in Tab. 3, we achieve 1.01% gains for Deit-B, 0.39% gains for MiT-B5 and 0.72% gains for Swin-L. Fig. 4 shows the qualitative results, where SegDeformer provides smoother predictions and better details than the baseline.

4.3 Ablation Study

This section ablates different variants of our SegDeformer framework on ADE20K, including the components of $\mathcal{M}_{inter}$ and $\mathcal{M}_{exter}$, and the use of $\mathcal{M}_{inter}$ and $\mathcal{M}_{exter}$. Some other discussions and comparisons are also included.

Designs of $\boldsymbol{X_{inter}}$. This section studies the structure design of $\mathcal{M}_{inter}$. As shown in Tab. 4, Equipping $\mathcal{M}_{inter}$ with only a one-head self-attention can already boost the performance by a large margin, from 40.97% to 42.65%. The experimental results are robust and no additional performance gains are brought when using more complex designs, so we choose to keep the simplest design.

Table 4. Studies on the structure of internal context mining modules based on SegFormer-B1, including the number of heads, depths, and the use of MLP after the internal attention.

Heads	Depths	MLP	mIoU
0	0		40.97
1	1		42.65 (1.68 ↑)
1	2		42.53 (1.56 ↑)
2	1		42.62 (1.65 ↑)
1	1	✓	**42.81** (1.84 ↑)

Table 5. Studies on the design of external context mining modules based on SegFormer-B1, including the type of X_{exter}, the use of $\mathcal{L}_{attn}$ and Y_{exter}, and the decoupling (*De.*) operation.

Type of X_{exter}		$\mathcal{L}_{attn}$	Y_{exter}	*De.*	mIoU↑
Mom.	Learn.				
					40.97
✓			✓		41.44 (0.47 ↑)
✓		✓	✓		41.74 (0.77 ↑)
	✓		✓		41.57 (0.60 ↑)
	✓	✓			41.86 (0.89 ↑)
	✓	✓	✓		42.37 (1.40 ↑)
	✓	✓	✓	✓	**42.84** (1.87↑)

Type of X_{exter}. We first replace the learnable external tokens with momentum update tokens to analyze the influence of the type of X_{exter}. The momentum update schedule is the same as [17]. The results are shown in Tab. 5. Compared to baseline, it achieves minor improvement (0.47%) and adds mid-level supervision merely bringing 0.30% additional gains, which is still 0.63% lower than our learnable external tokens under the same setting. We believe this is because the momentum updated features are relatively similar to the internal features, so their augmentation effect is limited, while our method can bring more hierarchical features. More analysis is included in the appendix.

Influence of $\mathcal{L}_{attn}$, Y_{exter} and the decoupling operation. Then we study $\mathcal{L}_{attn}$, Y_{exter} and the decoupling operation. As shown in Tab. 5, although learning X_{exter} freely can also bring performance improvements, it is not as effective as adding additional regularization to each token (41.57% vs 42.37%). Removing Y_{exter} and directly using X_{mid} for feature fusion means merely using one attention layer in $\mathcal{M}_{exter}$, which causes 0.51% performance loss compared to our final module. Decoupling X_{exter} can bring further performance gain, from 42.37% to 42.84%, which indicates that fully mining external information has more potential benefits.

Table 6. Effectiveness of internal and external context mining based on SegFormer-B1.

Mining Type		mIoU↑
Internal	External	
		40.97
✓		42.65 (1.68 ↑)
	✓	42.84 (1.87 ↑)
✓	✓	**44.05** (3.08 ↑)

Table 7. Studies on the applicability of SegDeformer and optimization modules. † means using UperNet. ⋆ means using efficient self-attention.

Method	Params↓	fps↑	mIoU↑
Deit-S†	52.09	30.30	42.87
Deit-S + SegDeformer	27.21	**33.65**	43.93
Deit-S† + SegDeformer	54.48	28.52	**44.10**
Swin-S†	81.26	**15.89**	47.72
Swin-S + SegDeformer	54.02	4.83	48.16
Swin-S† + SegDeformer	83.65	4.34	**48.94**
Swin-S†* + SegDeformer	92.43	11.44	48.42
SegFormer-B2	24.91	**33.47**	45.58
SegFormer-B2 + SegDeformer	25.40	7.80	**47.49**
SegFormer-B2* + SegDeformer	27.59	24.63	47.09

Effectiveness of $\mathcal{M}_{inter}$ and $\mathcal{M}_{exter}$. Tab. 6 inspects the influence of internal and external mining. It reveals that both kinds of mining bring performance gains (1.68% for $\mathcal{M}_{inter}$ and 1.87% for $\mathcal{M}_{exter}$), and combining them brings larger improvement (3.08%). The results reflect the effectiveness of mining in both global-local and cross-image contexts, and their effects are complementary.

Adaptability of SegDeformer. In Tab. 7, we investigate the adaptability of our SegDeformer. As shown, our method can bring performance improvement to all the encoders. For Deit-B and Swin-L, replacing the original UperNet decoder with SegDeformer can respectively bring 1.06% and 0.44% gains, and integrating SegDeformer with UperNet decoder can lead to another 0.27% and 0.78% improvement. For SegFormer-B2, our SegDeformer can also boost the performance from 45.58% to 47.49%, which demonstrates its adaptability.

Complexity analysis. Tab. 7 also reports the parameters and latency for a comprehensive comparison. For flat structures like DeiT-S, SegDeformer enjoys fewer parameters and less latency (27.21M and 33.65fps) compared to the UperNet decoder (52.09M and 30.30fps). For deep-narrow structures like MiT-B2 and Swin-S, although the amount of parameters is still small (54.02M and 25.40M),

Table 8. Ablation studies on SegFormer-B2 with different depths and widths at base decoders, denoted as (*depth,width*). $\star$ means using efficient self-attention.

Method	Params↓	mIoU↑
SegFormer-B2 (2,256)	**24.91**	45.58
SegFormer-B2 (8,256)	25.46	45.68
SegFormer-B2 + SegDeformer (-,256)	25.40	**47.49**
Method	fps↑	mIoU↑
SegFormer-B2 (2,256)	**33.47**	45.58
SegFormer-B2 (8,512)	24.33	46.08
SegFormer-B2$\star$ + SegDeformer (-,256)	24.63	**47.09**

the latency of SegDeformer becomes hard to tolerate ($4.83fps$ and $7.80fps$). This is because the final output size of the deep-narrow network is larger, which increases the computational burden of self-attention, and we can resort to efficient self-attention for better performance and efficiency trade off. The latency can be great optimized when using efficient self-attention, *i.e.*, $4.83fps \to 11.44fps$ for MiT-B2 and $7.80fps \to 24.63fps$ for Swin-S, with a tiny performance penalty.

Comparison with decoders with varying depths and widhts. We deliberately design decoders with varying depths and widths for fair comparisons with our SegDeformer under roughly the same parameters and fps. As shown in Tab. 8, merely adding depths or widths brings marginal gains, and SegDeformer enjoys better performance, which validates that the advantage comes from the architecture rather than the capacity. Note that for fps, we provide an efficient self-attention optimization to fit some deep-narrow encoders, and SegDeformer with efficient self-attention achieves better performance under close fps. We also provide comparisons with some other representative decoders in the appendix.

5 Conclusion

This paper proposed a pure transformer-based decoder termed SegDeformer for semantic segmentation, which can effectively model the intra- and inter-image context for better feature representation. The main contributions are two folds. First, we propose an internal context mining module equipped with an internal attention layer to capture global-local context within an image. Second, we model cross-image context via introducing learnable external tokens and designing an external context mining module for cross-image feature interaction. We also provide several optimization modules for scalable deployment. SegDeformer can be integrated with different encoders and experiments demonstrate its effectiveness.

Acknowledgement. This work was supported in part by the National Natural Science Foundation of China under Grant 61932022, Grant 61931023, Grant 61971285, Grant 62120106007, and in part by the Program of Shanghai Science and Technology Innovation Project under Grant 20511100100.

References

1. Caesar, H., Uijlings, J., Ferrari, V.: Coco-stuff: Thing and stuff classes in context. In: CVPR (2018)
2. Chen, L., Papandreou, G., Kokkinos, I., et. al: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR (2015)
3. Chen, L., Papandreou, G., Kokkinos, I., et. al: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. In: TPAMI (2017)
4. Chen, L., Papandreou, G., Schroff, F., et. al: Rethinking atrous convolution for semantic image segmentation. arXiv (2017)
5. Chen, L., Zhu, Y., Papandreou, G., et. al: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)
6. Cheng, B., Schwing, A., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. In: NeurIPS (2021)
7. Chu, X., Tian, Z., Wang, Y., et. al: Twins: Revisiting the design of spatial attention in vision transformers. In: NeurIPS (2021)
8. Cordts, M., Omran, M., Ramos, S., et. al: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
9. Cordts, M., Omran, M., Ramos, S., et. al: Semantic understanding of scenes through the ade20k dataset. In: CVPR (2017)
10. Dong, X., Bao, J., Chen, D., et.al: Cswin transformer: A general vision transformer backbone with cross-shaped windows. arXiv (2021)
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., et. al: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)
12. Fang, J., Xie, L., Wang, X., et. al: Msg-transformer: Exchanging local spatial information by manipulating messenger tokens. arXiv (2021)
13. Fu, J., Liu, J., Tian, H., et. al: Dual attention network for scene segmentation. In: CVPR (2019)
14. Huang, S., Lu, Z., Cheng, R., et. al: FaPN: Feature-aligned pyramid network for dense image prediction. In: ICCV (2021)
15. Huang, Z., Wang, X., Wei, Y., et. al: Ccnet: Criss-cross attention for semantic segmentation. In: TPAMI (2020)
16. Jain, J., Singh, A., Orlov, N., et al: Semask: Semantically masking transformer backbones for effective semantic segmentation. arXiv (2021)
17. Jin, Z., Gong, T., Yu, D., et.al: Mining contextual information beyond image for semantic segmentation. In: ICCV (2021)
18. Kirillov, A., Girshick, R., He, K., et. al: Panoptic feature pyramid networks. In: CVPR (2019)
19. Li, X., Zhong, Z., Wu, J., et. al: Expectation-maximization attention networks for semantic segmentation. In: ICCV (2019)
20. Lin, T., Maire, M., Belongie, S., et. al: Microsoft coco: Common objects in context. In: ECCV (2014)
21. Liu, Z., Lin, Y., Cao, Y., et.al: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV (2021)
22. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
23. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. arXiv (2021)

24. Strudel, R., Garcia, R., Laptev, I., et. al: Segmenter: Transformer for semantic segmentation. In: ICCV (2021)
25. Touvron, H., Cord, M., Douze, M., et. al: Training data-efficient image transformers & distillation through attention. arXiv (2020)
26. Vaswani, A., Shazeer, N., Parmar, N., et.al: Attention is all you need. In: NeurIPS (2017)
27. Wang, W., Zhou, T., Yu, F., et. al: Exploring cross-image pixel contrast for semantic segmentation. In: ICCV (2021)
28. Wang, X., Girshick, R., Gupta, A., et. al: Non-local neural networks. In: CVPR (2018)
29. Xiao, T., Liu, Y., Zhou, B., et. al: Unified perceptual parsing for scene understanding. In: ECCV (2018)
30. Xie, E., Wang, W., Yu, Z., et.al: Segformer: Simple and efficient design for semantic segmentation with transformers. In: NeurIPS (2021)
31. Yuan, Y., Chen, X., Wang, J.: Object-contextual representations for semantic segmentation. arXiv (2019)
32. Yuan, Y., Wang, J.: Ocnet: Object context network for scene parsing. arXiv (2018)
33. Zhang, F., C, Y., Li, Z., et. al: Acfnet: Attentional class feature network for semantic segmentation. In: ICCV (2019)
34. Zhang, H., Dana, K., Shi, J., et. al: Context encoding for semantic segmentation. arXiv (2018)
35. Zhao, H., Shi, J., Qi, X., et. al: Pyramid scene parsing network. In: CVPR (2017)
36. Zhao, H., Zhang, Y., Liu, S., et. al: PSANet: Point-wise spatial attention network for scene parsing. In: ECCV (2018)
37. Zheng, S., Lu, J., Zhao, H., et. al: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: CVPR (2021)