Transformer Scale Gate for Semantic Segmentation

Hengcan Shi, Munawar Hayat, Jianfei Cai
Department of Data Science & AI, Monash University
Wellington Rd, Clayton VIC 3800, Melbourne, Australia
{hengcan.shi, munawar.hayat, jianfei.cai}@monash.edu

Abstract

Effectively encoding multi-scale contextual information is crucial for accurate semantic segmentation. Most of the existing transformer-based segmentation models combine features across scales without any selection, where features on sub-optimal scales may degrade segmentation outcomes. Leveraging from the inherent properties of Vision Transformers, we propose a simple yet effective module, Transformer Scale Gate (TSG), to optimally combine multi-scale features. TSG exploits cues in self and cross attentions in Vision Transformers for the scale selection. TSG is a highly flexible plug-and-play module, and can easily be incorporated with any encoder-decoder-based hierarchical vision Transformer. Extensive experiments on the Pascal Context, ADE20K and Cityscapes datasets demonstrate that the proposed feature selection strategy achieves consistent gains.

1. Introduction

Semantic segmentation aims to segment all objects including 'things' and 'stuff' in an image and determine their categories. It is a challenging task in computer vision, and serves as a foundation for many higher-level tasks, such as scene understanding [15,34], object recognition [23,29] and vision+language [30, 33]. In recent years, Vision Transformers based on encoder-decoder architectures have become a new paradigm for semantic segmentation. The encoder consists of a series of multi-head self-attention modules to capture features of image patches, while the decoder has both self- and cross-attention modules to generate segmentation masks. Earlier works [47] usually use Vision Transformers designed for image classification to tackle semantic segmentation, and only encode single-scale features. However, different from image classification where we only need to recognize one object in an image, semantic segmentation is generally expected to extract multiple objects of different sizes. It is hard to segment and recognize these varying sized objects by only single-scale features.

Some methods [20, 37] attempt to leverage multi-scale features to solve this problem. They first use hierarchical transformers such as Swin Transformer [20] and PVT [37] to extract multi-scale image features, and then combine them, *e.g.* by the pyramid pooling module (PPM) [46] or the seminal feature pyramid network (FPN) [18] borrowed from CNNs. We argue that such feature combinations cannot effectively select an appropriate scale for each image patch. Features on sub-optimal scales may impact the segmentation performance.To address this issue, CNN-based methods [3, 8, 14, 32] design learnable models to select the optimal scales. Nevertheless, these models are complex, either use complex mechanisms [3, 8, 14] and/or require scale labels [32], which decrease the network efficiency and may cause over-fitting.

In this paper, we exploit the inherent characteristics of Vision Transformers to guide the feature selection process. Specifically, our design is inspired from the following observations: **(1)** As shown in Fig. 1(a), the self-attention module in the transformer encoder learns the correlations among image patches. If an image patch is correlated to many patches, small-scale (low-resolution) features may be preferred for segmenting this patch, since small-scale features have large effective receptive fields [42] to involve as many of these patches as possible, and vice versa. **(2)** In the transformer decoder, the cross-attention module models correlations between patch-query pairs, as shown in Fig. 1(b), where the queries are object categories. If an image patch is correlated to multiple queries, it indicates that this patch may contain multiple objects and need large-scale (high-resolution) features for fine-grained segmentation. On the contrary, an image patch correlated to only a few queries may need small-scale (low-resolution) features to avoid over-segmentations. **(3)** The above observations can guide design choices for not only category-query-based decoders but also object-query-based decoders. In object-query-based decoders, each query token corresponds to an object instance. Then, the cross-attention module extracts relationships between patch-object pairs. Many high cross-attention values also indicate that the image patch may con-

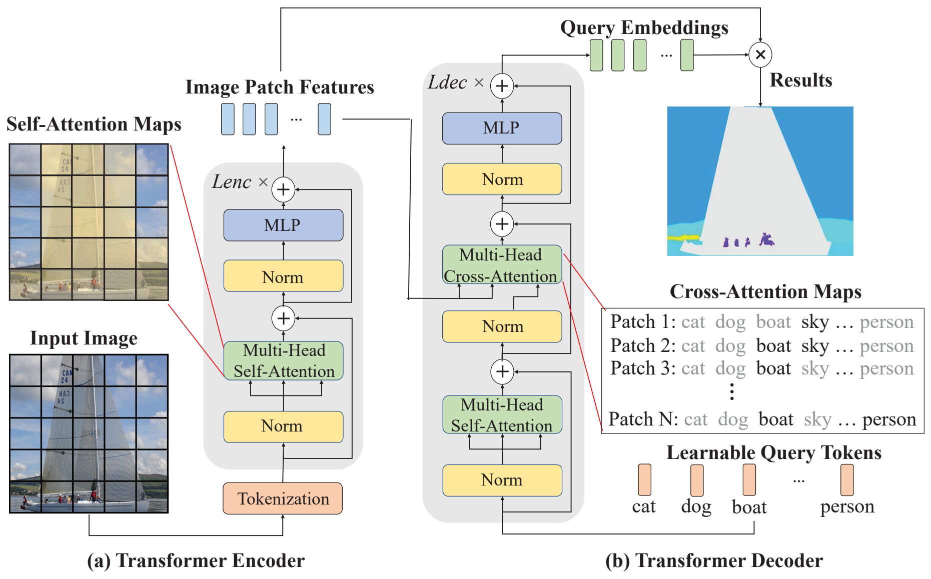


Figure 1. Illustration of Vision Transformers for semantic segmentation. (a) The image is divided into multiple patches and input into the encoder. The encoder contains L_{enc} blocks and outputs features for every image patch. (b) The decoder takes learnable query tokens as inputs, where each query is corresponding to an object category. The decoder with L_{dec} blocks outputs query embeddings. Finally, segmentation results are generated by multiplying the image patch features and the query embeddings. In this paper, we exploit inner properties of these attention maps for multi-scale feature selection.

tain multiple objects and require high-resolution features. Note that these are general observations, which do not mean that these are the only relationships between transformer correlation maps and feature scales.

From these observations and analyses, we propose a novel Transformer Scale Gate (TSG) module that takes correlation maps in self- and cross-attention modules as inputs, and predicts weights of multi-scale features for each image patch. Our TSG is a simple design, with a few lightweight linear layers, and thus can be plug-and-play across diverse architectures. We further extend TSG to a TSGE module and a TSGD module, which leverage our TSG weights to optimize multi-scale features in transformer encoders and decoders, respectively. TSGE employs a pyramid structure to refine multi-scale features in the encoder by the self-attention guidance, while TSGD fuses these features in the decoder based on the cross-attention guidance. Experimental results on three datasets, Pascal Context, ADE20K, Cityscapes show that the proposed modules consistently achieve gains, up to 4.3% in terms of mIoU, compared with Swin Transformer based baseline [20].

Our main contributions can be summarized as follows: **(1)** To the best of our knowledge, this is the first work to exploit inner properties of transformer attention maps for multi-scale feature selection in semantic segmentation. We analyze the properties of Vision Transformers and design TSG for the selection. **(2)** We propose TSGE and TSGD in the encoder and decoder in transformers, respectively, which leverage our TSG to improve the semantic segmentation performance. **(3)** Our extensive experiments and ablations show that the proposed modules obtain significant improvements on three semantic segmentation datasets.

2. Related Work

CNN-based semantic segmentation methods usually use the fully convolutional network (FCN) [21], which formulates the semantic segmentation task as a pixel-wise classification problem and designs an encoder-decoder structure. The encoder extracts features of each pixel in the image, while the decoder labels every pixel. Noh *et al.* [24] design a deconvolutional decoder to gradually restore more details, which is the mirror of the CNN encoder. Although these approaches make significant progress for recognizing objects of fixed size ranges, they struggle to segment objects of diverse sizes. To segment variable sized objects, some prior works propose to use multi-scale strategies. Lin *et al.* [17] directly resize the input image into multiple scales and generate multiple predictions, which are then assembled together. Some works extract multi-scale features in the encoder by pyramid pooling module (PPM) and pyramid atrous convolutions and feature pyramid network (FPN), and then combine multi-scale features to predict segmentation results. Another methods [1,28] adopt the deconvolutional decoder [24] and incorporate multi-scale features into the decoder. All these methods only simply combine multi-scale features without any selection. While some existing works [3,8,14,31,32] propose learnable modules to select multiple scales, they either require complex mechanisms [3,8,14] and/or scale labels [32], and are therefore not ideally suited for efficient scale selection.

Vision Transformers have recently attracted increasing research interest and have become a new paradigm for semantic segmentation, thanks to their ability of modeling long-range dependencies. Ranftl *et al.* [27] and Zheng *et al.* [47] employ ViT [10] as the encoder to extract single-scale features and use CNN-based decoders for semantic segmentation. Strudel *et al.* [36] and Xie *et al.* [41] design transformer-based decoders, which take categories as queries. Cheng *et al.* [6] propose an object-query-based transformer decoder and combine it with a pixel-level decoder to predict segmentation results. These methods ignore the diversity of object sizes. Other recent works design pyramid architectures to obtain multi-scale features in the encoder and fuse them in the decoder for segmentation. Most of them [9, 20, 37, 39, 43, 44] adopt PPM [46] and/or FPN [18] for multi-scale feature fusion, while Xie *et al.* [42] and Gu *et al.* [11] use lightweight concatenations. Lee *et al.* [13] propose multi-path Transformer blocks to better integrate multi-scale features. Instead of combining multi-scale features, Bousselham *et al.* [2] leverage multiple transformer decoders to generate multi-scale segmentation results and then aggregate these results. Qin *et al.* [26] averages multi-scale results and adds a cross-attention model to capture inter-scale information. However, these combinations cannot effectively select multi-scale features for each image patch. Several works [16, 35, 38, 45, 49] select scales from transformer image or query embeddings. Different from these works, our method exploits the relationships between transformer attention maps and segmentation scales instead of image features/embeddings for scale selection. Meanwhile, our attention-map-based and previous feature/embedding-based scale selections are not mutually exclusive, but rather complementary.

3. Proposed Method

3.1. Vision Transformer for Semantic Segmentation

Vision Transformers typically contain an encoder and a decoder, as shown in Fig. 1. An image is first split into multiple patches and every patch is embedded into a token. Let $\mathbf{Z} = \{\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_N}\}$ represent the set of tokens, where N is the number of patches, $\mathbf{z_n} \in \mathbb{R}^{dz}$ $(n = 1, ..., N)$ is the token of the n-th patch, and d_Z is the dimension of tokens.

Encoder. The encoder takes these tokens as inputs, and outputs the feature vector of every image patch. The key component in the encoder is the multi-head self-attention, which learns the long-range dependencies of image patches,

$$\mathbf{Q_i} = Linear(\mathbf{Z}),\ \ \mathbf{K_i} = Linear(\mathbf{Z}),\ \ \mathbf{V_i} = Linear(\mathbf{Z}) \tag{1}$$

$$\mathbf{A_i^{self}} = Softmax(\frac{\mathbf{Q_i}\mathbf{K_i}^T}{\sqrt{d_s}}) \tag{2}$$

$$\mathbf{H_i} = \mathbf{A_i^{self}}\mathbf{V_i} \tag{3}$$

$$\mathbf{O} = Linear(Concat(\mathbf{H_1}, ..., \mathbf{H_{h_{self}}})) \tag{4}$$

where $\mathbf{Z} \in \mathbb{R}^{N\times d_Z}$ is the image token set, $i = 1, ..., h_{self}$ and h_{self} is the number of heads in the multi-head self-attention module. $Linear(\cdot)$ is the linear layer; $Softmax(\cdot)$ is the Softmax function; and $Concat(\cdot)$ represents the concatenation. $\mathbf{Q_i}, \mathbf{K_i}, \mathbf{V_i} \in \mathbb{R}^{N\times d_s}$ denote the query, key and value, respectively, where d_s is their dimension. $\mathbf{A_i^{self}} \in \mathbb{R}^{N\times N}$ is the self-attention map, which models long-range dependencies between every patch pair in the image. $\mathbf{H_i} \in \mathbb{R}^{N\times d_s}$ is the feature map generated by the i-th attention head. Feature maps in all heads are concatenated into a single map and transformed by a linear layer to output d_O-dimensional feature map $\mathbf{O} \in \mathbb{R}^{N\times d_O}$.

An encoder block composes of a multi-head self-attention module, a multilayer perceptron (MLP) and normalizations, as shown in Fig. 1 (a). By cascading L_{enc} encoder blocks, we can obtain the encoder-output image patch feature map $\mathbf{F} \in \mathbb{R}^{N\times d_F}$, and $\mathbf{F} = \{\mathbf{f_1}, \mathbf{f_2}, ..., \mathbf{f_N}\}$, where $\mathbf{f_n} \in \mathbb{R}^{d_F}$ is the encoder-output feature vector for the n-th image patch, and d_F is the feature dimension.

Our baseline decoder. The inputs of the transformer decoder are a series of query tokens. Inspired by [36], we use C query tokens $\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_C}\}$, where C is the number of object classes and each token $\mathbf{x_c} \in \mathbb{R}^{d_F}$ $(c = 1, ..., C)$ corresponds to a class. The dimension of query tokens is the same as image patch features. Different from [36], which also uses image patch features $\mathbf{F}$ as query tokens in the decoder, we take image patch features $\mathbf{F}$ as keys and values to reduce computational costs.

As shown in Fig. 1 (b), the decoder includes multi-head self-attention modules and cross-attention modules. Similar to the encoder, self-attention modules take query tokens as inputs and learn their relationships. In contrast, cross-attention modules aim to capture relationships between each image patch and query token:

$$\mathbf{Q_j} = Linear(\mathbf{X}), \mathbf{K_j} = Linear(\mathbf{F}), \mathbf{V_j} = Linear(\mathbf{F}) \tag{5}$$

$$\mathbf{A_j^{cross}} = Softmax(\frac{\mathbf{Q_j}\mathbf{K_j}^T}{\sqrt{d_c}}) \tag{6}$$

$$\mathbf{H_j} = \mathbf{A_j^{cross}}\mathbf{V_j} \tag{7}$$

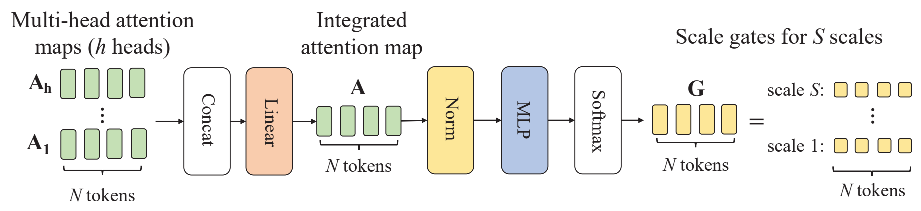


Figure 2. Illustration of our Transformer Scale Gate (TSG). Consider transformer multi-head attention maps $\{\mathbf{A_1}, ..., \mathbf{A_h}\}$ (h heads), each of which contains N image patches (tokens), with each patch represented as a 512-d feature vector. Our TSG first integrates them into one map $\mathbf{A}$, and then generates scale gates $\mathbf{G}$ for each patch. By these scale gates, our model selects suitable segmentation scales from $S = 4$ scales for each patch. The detailed usages of $\mathbf{G}$ are shown in Fig. 3&4.

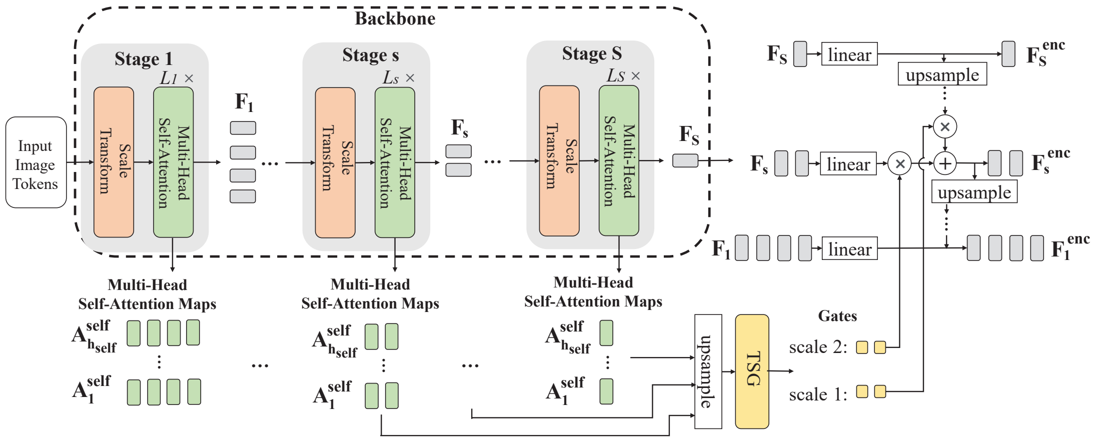


Figure 3. Transformer Scale Gate in Encoder (TSGE). For simplicity, we only depict the self-attention module in each block. In the encoder, our TSG generates scale gates from self-attention maps, and we leverage these gates to select and fuse multi-scale feature maps.

where $\mathbf{F} \in \mathbb{R}^{N\times d_F}$ represents the image patch feature matrix, $\mathbf{X} \in \mathbb{R}^{C\times d_F}$ is the query token matrix, $j = 1, ..., h_{cross}$ and h_{cross} denotes the number of heads in the cross-attention module. Similar to self-attention modules, $\mathbf{Q_j}, \mathbf{K_j}, \mathbf{V_j} \in \mathbb{R}^{N\times d_c}$ are the d_c-dimension query, key and value, respectively. $\mathbf{A_j^{cross}} \in \mathbb{R}^{C\times N}$ captures relationships between every patch-class pair. $\mathbf{H_j} \in \mathbb{R}^{C\times d_c}$ represents the feature map in the j-th head, and features in all heads can be combined by Eq. (4).

Through L_{dec} decoder blocks, our decoder outputs query embeddings $\mathbf{Y} = \{\mathbf{y_1}, \mathbf{y_2}, ..., \mathbf{y_C}\}$, where $\mathbf{Y} \in \mathbb{R}^{C\times d_F}$ and $\mathbf{y_c} \in \mathbb{R}^{d_F}$ is the embedding vector of the c-th class. The segmentation result can be predicted by the matrix product of patch feature matrix $\mathbf{F}$ and class query embedding matrix $\mathbf{Y}$:

$$\mathbf{P} = Softmax(\frac{\mathbf{F}\mathbf{Y}^T}{\sqrt{d_F}}) \tag{8}$$

where $\mathbf{P} \in \mathbb{R}^{N\times C}$ contains the classification scores for every image patch. The segmentation result is composed of these patch-wise classification results.

3.2. Transformer Scale Gate (TSG)

From Eq. (2)&(6), we observe that the self-attention map $\mathbf{A_i^{self}}$ reflects the correlation between an image patch and other patches, while the cross-attention map $\mathbf{A_j^{cross}}$ reflects the correlation between every image patch and object category. In $\mathbf{A_i^{self}}$, if an image patch is highly related to a large number of patches, it may require small-scale features (large effective receptive fields), and vice versa. In $\mathbf{A_j^{cross}}$, if an image patch is correlated to many object classes, this patch may contain multiple objects and require large-scale (high-resolution) features.

Considering the above observations, our TSG takes attention maps as inputs and generates gates for every scale. As shown in Fig. 2, we first integrate multi-head attention maps into a single map $\mathbf{A} \in \mathbb{R}^{N\times d_A}$, where d_A is its dimension. For self-attention modules in the encoder, attention maps are integrated as

$$\mathbf{A} = Linear(Concat(\mathbf{A_1^{self}}, ..., \mathbf{A_{h_{self}}^{self}})). \tag{9}$$

We concatenate attention maps in all heads and use a linear layer to project them to d_A dimensions.

For cross-attention modules in the decoder, we first

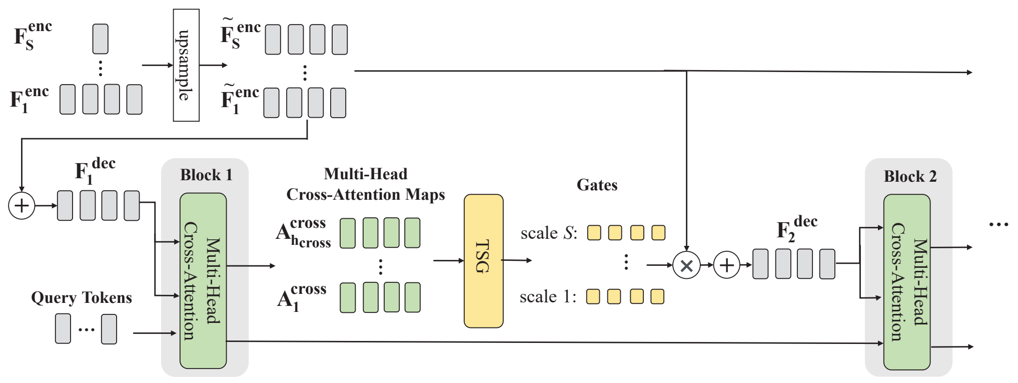


Figure 4. Transformer Scale Gate in Decoder (TSGD). We only show the cross-attention module in each block for simplicity. $\oplus$ and $\otimes$ mean the sum and product of all inputs, respectively. In the decoder, TSG predicts scale gates from cross-attention maps to better select multi-scale feature maps for decoding.

change the softmax in $\mathbf{A_j^{cross}}$. The original softmax is applied on the patch dimension (i.e., over N image patches), which reflects the importance of patches for each object class. We change this softmax to the class dimension. The changed softmax reveals the importance of classes for each image patch, which is more suitable for our TSG. Let $\widetilde{\mathbf{A}}_\mathbf{j}^\mathbf{cross}$ represent the cross-attention map with the modified softmax. We concatenate the transposes of $\widetilde{\mathbf{A}}_\mathbf{j}^\mathbf{cross}$ in all heads and use a linear layer to transform dimensions,

$$\mathbf{A} = Linear(Concat((\widetilde{\mathbf{A}}_\mathbf{1}^\mathbf{cross})^T, ..., (\widetilde{\mathbf{A}}_\mathbf{h_{cross}}^\mathbf{cross})^T)). \quad (10)$$

After the integration, our TSG generates multi-scale feature gates as

$$\widetilde{\mathbf{G}} = MLP(Norm(\mathbf{A})), \quad (11)$$

$$\mathbf{G} = Softmax(\widetilde{\mathbf{G}}). \quad (12)$$

We employ a layer normalization to normalize $\mathbf{A}$, and use an MLP to predict the scale gates $\widetilde{\mathbf{G}} \in \mathbb{R}^{N \times S}$, where S is the number of scales. Here, we use a two-layer MLP with GELU activation function. $\widetilde{\mathbf{G}}$ is normalized into $\mathbf{G}$ by a softmax function on the scale dimension. In matrix $\mathbf{G}$, value $g_{n,s}$ indicates the gate of the s-th scale for the n-th image patch. We next introduce how to use our scale gates to select features in the encoder and decoder.

3.3. Transformer Scale Gate in Encoder (TSGE)

The multi-scale transformer backbone contains S stages. In each stage s, the backbone generates a feature map $\mathbf{F_s} \in \mathbb{R}^{N_s \times d_{F,s}}$, where $d_{F,s}$ is its dimension, and N_s is the number of patches in this feature map. Therefore, we have S multi-scale features $\{\mathbf{F_1}, \mathbf{F_2}, ..., \mathbf{F_S}\}$ from the backbone model. We propose a TSGE module, which leverages TSG to generate scale gates $\mathbf{G}$ to refine these features, where we denote $\{\mathbf{F_1^{enc}}, \mathbf{F_2^{enc}}, ..., \mathbf{F_S^{enc}}\}$ as the refined features, $\mathbf{F_s^{enc}} \in \mathbb{R}^{N_s \times d_F}$ $(s = 1, ..., S)$, and all features are refined into the same dimension d_F.

Inspired by feature fusion methods in CNN [8, 18], our TSGE gradually fuses small-scale features into large-scale features, as shown in Fig. 3. Specifically, we fuse two feature maps, the smaller-scale refined feature map $\mathbf{F_{s+1}^{enc}}$ and the larger-scale feature map $\mathbf{F_s}$, at each step. $\mathbf{F_{s+1}^{enc}}$ is first upsampled to fit the size of $\mathbf{F_s}$. Then, we use a linear layer to transform the feature dimension of the larger-scale feature map $\mathbf{F_s}$. Finally, the unsampled smaller-scale feature map and the transformed larger-scale feature map are weighted by our scale gates and summed as

$$\mathbf{f_{n,s}^{enc}} = g_{n,1}\widetilde{\mathbf{f}}_\mathbf{n,s+1}^\mathbf{enc} + g_{n,2}\widetilde{\mathbf{f}}_\mathbf{n,s} \quad (13)$$

where $\widetilde{\mathbf{f}}_\mathbf{n,s+1}^\mathbf{enc} \in \mathbb{R}^{d_F}$ and $\widetilde{\mathbf{f}}_\mathbf{n,s} \in \mathbb{R}^{d_F}$ are the feature vectors of the n-th image patch in the unsampled smaller-scale feature map and the transformed larger-scale feature map, respectively, and $\mathbf{f_{n,s}^{enc}} \in \mathbb{R}^{d_F}$ is the weighted sum. Weights $g_{n,1}$ and $g_{n,2}$ are generated from our TSG. The inputs of our TSG at each step are the self-attention maps corresponding to features maps we used. For example, when we fuse feature maps $\mathbf{F_s}$ and $\mathbf{F_{s+1}^{enc}}$, the inputs are self-attention maps in the last blocks from the s-th stage to the S-th stage, because $\mathbf{F_{s+1}^{enc}}$ has already included features from $\mathbf{F_{s+1}}$ to $\mathbf{F_S}$. Since these attention maps are in different sizes, we upsample them into the size of $\mathbf{F_s}$, before inputting them to TSG. For each image patch n, our TSG outputs two gates $g_{n,1}$ and $g_{n,2}$ for the two feature maps, respectively.

3.4. Transformer Scale Gate in Decoder (TSGD)

The refined features $\{\mathbf{F_1^{enc}}, \mathbf{F_2^{enc}}, ..., \mathbf{F_S^{enc}}\}$ are then input to our decoder, and we propose a TSGD module to integrate them. In the l-th decoder block, to integrate multi-scale features, we first upsample them into the same size. $\{\widetilde{\mathbf{F}}_\mathbf{1}^\mathbf{enc}, \widetilde{\mathbf{F}}_\mathbf{2}^\mathbf{enc}, ..., \widetilde{\mathbf{F}}_\mathbf{S}^\mathbf{enc}\}$ are the upsampled feature maps, where $\widetilde{\mathbf{F}}_\mathbf{s}^\mathbf{enc} \in \mathbb{R}^{N \times d_F}$ $(s = 1, ..., S)$ and $N = N_1$. Then, we leverage our TSG to predict scale gates, which takes the cross-attention maps from the previous block as inputs, as shown in Fig. 4. Our TSG outputs a matrix $\mathbf{G} \in \mathbb{R}^{N \times S}$, which contains the gates of all scales for every image patch. Finally, we use these gates to weight the upsampled feature maps and sum them as follows:

$$\mathbf{f}_{l,\mathbf{n}}^{\mathbf{dec}} = \sum_{s=1}^{S} g_{n,s} \widetilde{\mathbf{f}}_{\mathbf{n,s}}^{\mathbf{enc}} \tag{14}$$

where $\widetilde{\mathbf{f}}_{\mathbf{n,s}}^{\mathbf{enc}} \in \mathbb{R}^{d_F}$ is the feature vector of the n-th image patch in the feature map $\widetilde{\mathbf{F}}_{\mathbf{s}}^{\mathbf{enc}}$, $g_{n,s}$ in $\mathbf{G}$ represents the gate of the s-th scale for this patch, and $\mathbf{f}_{l,\mathbf{n}}^{\mathbf{dec}} \in \mathbb{R}^{d_F}$ is the weighted sum feature vector of this patch. The feature map $\mathbf{F}_l^{dec}$ consists of $\{\mathbf{f}_{l,\mathbf{1}}^{\mathbf{dec}}, \mathbf{f}_{l,\mathbf{2}}^{\mathbf{dec}}, ..., \mathbf{f}_{l,\mathbf{N}}^{\mathbf{dec}}\}$, which is used to generate keys and values in the current decoder block. For the first block, since there is no previous block, we only sum the upsampled multi-scale features:

$$\mathbf{f}_{\mathbf{1,n}}^{\mathbf{dec}} = \sum_{s=1}^{S} \widetilde{\mathbf{f}}_{\mathbf{n,s}}^{\mathbf{enc}}. \tag{15}$$

The final segmentation result is generated by Eq. (8), where we use the integrated feature map $\mathbf{F}_{\mathbf{L_{dec}}}^{\mathbf{dec}}$ in the last decoder block to generate the result:

$$\mathbf{P} = Softmax(\frac{\mathbf{F}_{\mathbf{L_{dec}}}^{\mathbf{dec}} \mathbf{Y}^T}{\sqrt{d_F}}). \tag{16}$$

4. Experiments

4.1. Experimental Settings

Datasets. We evaluate our method on three datasets, Pascal Context, ADE20K and Cityscapes. Pascal Context [22] contains 10103 images, 4998 for training and 5105 for validation. There are 60 category labels in this dataset, including 59 object classes and one background class. ADE20K [48] includes 150 object categories, 20210 training images, 2000 validation images and 3352 testing images. Cityscapes [7] has 19 object classes, 2975, 500, and 1525 images for training, validation and testing, respectively. For all these datasets, similar to previous works, we train our method on training images and report the results on validation images.

Metrics. We adopt the common segmentation metric, 'mIoU', for evaluation, which is the average of the 'IoU' values of all object classes. We report our results of a single model, without multi-scale and horizontal flip ensembles.

Implementation Details. Our TSG can be used for any hierarchical Vision Transformer. Here, we use Swin Transformer [20] as a running example. There are four-scale feature maps in Swin Transformer [20], i.e., $S = 4$. We only use local attention maps in each window in Swin Transformer to generate scale gates. We set the dimension d_F of refined features to 512, use eight heads for both self- and cross-attention modules, and use three blocks in the decoder. We also set d_A to 512, and employ a 512-dimension hidden layer in the MLP in our TSG. Our model is built on the Pytorch [25] platform. Following common settings [2, 20], we leverage weights pretrained on ImageNet-1K and ImageNet-22K to initialize Swin-T and Swin-L, respectively. Query tokens in the decoder are initialized to zero. Other parts are randomly initialized. We adopt cross-entropy loss, 'AdamW' optimizer and the 'poly' learning rate decay scheduling during training, with an initial learning rate of 6×10^{-5} and a weight decay of 10^{-2}.

4.2. Results and Comparisons

Table 1 shows the results of existing state-of-the-art methods and our method. On Pascal Context, compared with our baselines, Swin Transformer [20] Tiny and Large, our method achieves 4.3% and 3.0% gains, respectively. Compared with SenFormer [2], which also uses category-query-based decoders and multi-scale features, our proposed method yields improvements of 1.3% and 0.9% when using Swin-T and Swin-L backbones, respectively. However, the number of parameters of our method is significantly lower than that of SenFormer. SenFormer uses a heavyweight architecture with multiple transformer decoders to generate multi-scale predictions. In contrast, our method only leverages one decoder and lightweight scale gates, while achieving better performance. Results on ADE20K and Cityscapes also show our superior accuracy.

Our method can also be used for object-query-based decoders. Take Mask2Former [5] as an example. We add our TSGE to its encoder and TSGD to its transformer decoder to select optimal feature scales based on patch-object correlations. We also keep the masked attention in the decoder, as the same as in Mask2Former [5]. On Pascal Context, compared with the original Mask2Former [5], our method achieves 1.4% and 1.1% gains with Swin-T and Swin-L, respectively. Compared with other state-of-the-art methods, our method also shows the superior performance. These results demonstrate not only the effectiveness of our proposed method, but also its versatility in supporting both category-query-based decoders and object-query-based decoders.

Fig. 5(a) depicts qualitative results on the Pascal Context dataset. Swin Transformer [20] and SenFormer [2] only simply combine multi-scale features or multi-scale segmentation results without any selection, and thus fail to segment many small objects, such as the left 'person' in the second image in Fig. 5(a). Our TSG selects suitable scales for image patches. When segmenting patches including small objects, our approach selects high-resolution features based on transformer attention cues. Therefore, our method success-

Method	Backbone	Params	Pascal Context mIoU (%)	ADE20K mIoU (%)	Cityscapes mIoU (%)
Swin (Upernet decoder) [20]	Swin-T	60M	50.2	44.4	71.8
Swin (Upernet decoder) [20]	Swin-L	234M	60.3	52.1	79.9
TSG (Ours)	Swin-T	72M	54.5 (+4.3)	47.5 (+3.1)	75.8 (4.0)
TSG (Ours)	Swin-L	250M	**63.3** (+3.0)	**54.2** (+2.1)	**83.1** (+3.2)
Mask2Former [5]	Swin-T	42M	54.6	47.7	79.7
Mask2Former [5]	Swin-L	216M	63.8	56.1	83.3
TSG (Ours) + Mask2Former [5]	Swin-T	51M	56.0 (+1.4)	49.0 (+1.3)	80.5 (+0.8)
TSG (Ours) + Mask2Former [5]	Swin-L	232M	**64.9** (+1.1)	**56.9** (+0.8)	**83.6** (+0.3)
Other state-of-the-art methods					
PSPNet [46]	ResNet101	60M	47.0	42.0	78.4
DeepLabV3+ [4]	ResNet101	63M	47.4	45.5	80.9
SETR [47]	ViT-L	311M	54.9	48.6	79.3
Segformer [42]	MiT-B5	85M	54.8	51.0	82.4
Segmenter [36]	ViT-L	333M	58.1	51.8	80.4
MaskFormer [6]	Swin-T	42M	53.3	46.7	-
MaskFormer [6]	Swin-L	212M	62.6	54.1	-
HRViT [12]	HRViT-b3	28.6M	-	50.2	83.2
PCAA [19]	ResNet101	-	55.6	46.7	82.3
SenFormer [2]	Swin-T	144M	53.2	46.0	-
SenFormer [2]	Swin-L	314M	62.4	53.1	82.8
PFT [26] + Mask2Former [5]	Swin-L	232M	-	56.1	-
PFT [26] + Mask2Former [5] + MSDA [49]	Swin-L	232M	-	56.3	-

Table 1. Results of semantic segmentation on Pascal Context, ADE20k and Cityscapes validation. All methods use single model, without multi-scale-model and horizontal-flip ensembles. Our proposed TSG achieves consistent gains with different backbones.

Model	Encoder	Decoder	mIoU(%)
1	Swin-T	Upernet	50.2
2	Swin-T + TSGE	Upernet	51.6 (+1.4)
3	Swin-T	BD	51.3
4	Swin-T + TSG	BD	52.4 (+1.1)
5	Swin-T	BD + TSGD	52.7 (+1.4)
6	Swin-T + FPN	BD	52.9
7	Swin-T + TSGE	BD	53.9 (+1.0)
8	Swin-T + FPN	BD + TSGD	54.1 (+1.2)
9	Swin-T + TSGE	BD + TSGD	54.5 (+4.3)

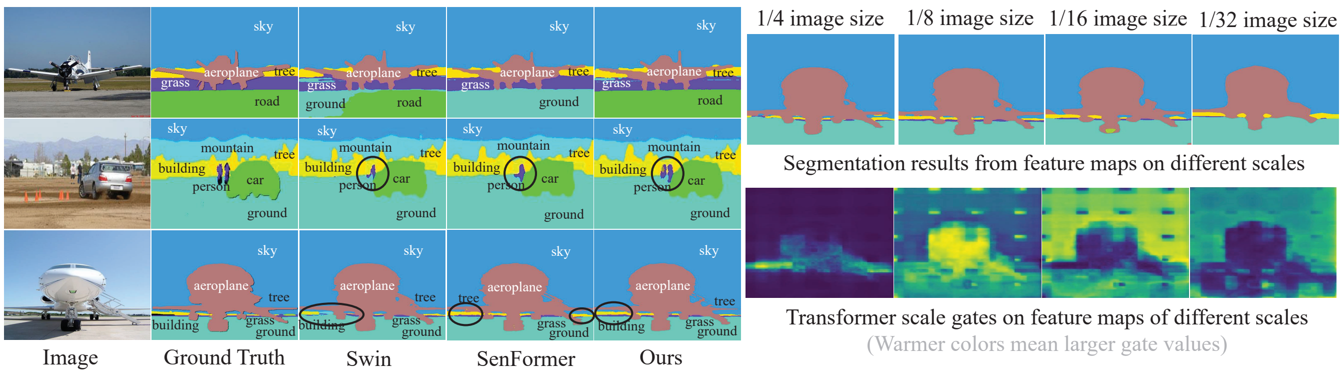
Table 2. The effects of main components in our method on Pascal Context validation. 'BD' means our baseline decoder.

fully segments these small objects. Previous approaches are also prone to over-segmentation and mis-recognitions. For example, in the first image in Fig. 5, Swin Transformer [20] over-segments the 'road' object as two objects, and SenFormer [2] mis-recognizes the 'road' object as 'ground'. Our method avoids these, by selecting suitable scales to segment and recognize objects of diverse scales.

4.3. Ablation Analysis

TSG in encoder. We conduct multiple ablations to verify the contributions of our TSG module for the Transformer encoder in Table 2. The vanilla Swin Transformer [20] (Model 1) takes UperNet [40] as its decoder, which already includes FPN and PPM. In Model 2, we use our TSGE to replace FPN and PPM in UperNet. From the first and second lines in Table 2, it can be observed that our method obtains an improvement of 1.4% in this setting. We next test models using our baseline decoder. Our baseline decoder requires to first integrate multi-scale feature maps into one feature map. Thus, we first add a linear layer to every feature map to convert their dimensions into the same. Then, the converted multi-scale feature maps are upsampled into the same size, and summed as the input of our baseline decoder. In Model 4, our TSG generates scale gates from self-attention modules in the encoder, and these gates are used to weight the multi-scale feature maps before summing them. Through our TSG, the mIoU can be improved by 1.1%. In Models 6&7, we use FPN [18] and our TSG to refine multi-scale feature maps, respectively, and sum the refined feature maps in our decoder. Compared with FPN, our TSGE achieves a gain of 1.0%.

TSG in decoder. From Models 3&5 in Table 2, we observe that our TSGD outperforms the baseline decoder by 1.4% when using the 'Swin-T' encoder. With the 'Swin-T + FPN' encoder (Models 6&8), our TSGD improves the performance by 1.2%. Our final model (Model 9) uses both TSGE and TSGD, which achieves an improvement of 4.3%, compared with Swin Transformer [20]. These results suggest that our proposed TSG is effective in fusing cues across


sky
aeroplane
tree
grass
road
ground
mountain
building
person
car
Image
Ground Truth
Swin
SenFormer
Ours
1/4 image size
1/8 image size
1/16 image size
1/32 image size
Segmentation results from feature maps on different scales
Transformer scale gates on feature maps of different scales
(Warmer colors mean larger gate values)

(a) Segmentation results **(b) Visualization of our scale gates**

Figure 5. (a) Qualitative results on Pascal Context samples, with Swin-L backbone. Our method generates better results than prior works, especially for large and small objects. (b) Visualization of our transformer scale gates. For small objects ('tree', 'building', etc), TSG generates large gate values on large-scale feature maps (1/4 image size), indicating that our model successfully chooses high-resolution maps for small objects, and vice versa.

Model	F_1 (1/4)	F_2 (1/8)	F_3 (1/16)	F_4 (1/32)	mIoU(%)
Swin-T + FPN + BD				✓	47.2
Swin-T + FPN + BD			✓		50.1
Swin-T + FPN + BD		✓			51.6
Swin-T + FPN + BD	✓				52.5
Swin-T + FPN + BD	✓	✓	✓	✓	52.9
Swin-T + TSGE + BD + TSGD	✓	✓	✓	✓	54.5

Table 3. The effects of features on different scales on Pascal Context validation. 'BD' is our baseline decoder. '1/4', '1/8', '1/16' and '1/32' mean that the resolutions of feature maps are '1/4', '1/8', '1/16' and '1/32' of the image size.

Design	mIoU(%)
TSG with multi-head average	54.1
TSG with multi-head concatenation	54.5 (+0.4)
TSGs with shared weights	53.8
TSGs with independent weights	54.5 (+0.7)

Table 4. The impact of different design choices in our TSG module evulated on Pascal Context validation set.

multiple spatial resolutions.

Results on different scales. We compare the results from multi-scale features in Table 3. Our method significantly outperforms all singe-scale models and the approaches that simply combine the multi-scale features, benefiting from our transformer-based scale selection.

Dissecting TSG. Table 4 shows the results of our TSG with different settings. In Sec. 3.2, we concatenate multi-head attention maps, which improves the mIoU by 0.4%, compared with averaging multi-head attention maps. This is because some information may be lost in the average, while the concatenation retains all information in attention maps. We use multiple TSG with independent weights in different steps in the encoder and different blocks in the decoder. Compared with sharing weights among different TSGs, independent weights achieve 0.7% improvements, because every TSG has different inputs and independent weights show better ability for different inputs. Nonetheless, our TSG with shared weights can also improve the performance by 3.6%, compared with Swin Transformer [20] baseline.

TSG visualization. We visualize our scale gates in Fig. 5(b). It can be observed that our TSG highlights small objects ('tree' and 'grass') in large-scale feature maps for more accurate segmentation, while selecting smaller-scale feature maps for larger objects ('aeroplane', 'sky' and 'ground') to reduce over-segmentations.

5. Conclusion

In this paper, we have presented a Transformer Scale Gate (TSG) module, which exploits inherent properties in Vision Transformers to effectively select multi-scale features for semantic segmentation. Our TSG is a simple and lightweight transformer-based module, which can be used in transformer segmentation networks in a plug-and-play manner. We have also proposed TSGE and TSGD, which leverage our TSG to further improve the segmentation accuracy in the transformer encoder and decoder, respectively. TSGE refines multi-scale features in the encoder by the self-attention guidance, while TSGD integrates multi-scale features in the decoder based on cross-attention maps. Extensive experiments on three semantic segmentation datasets demonstrate the effectiveness of our proposed method.

Acknowledgement. This research is supported by Monash FIT Start-up Grant and ARC DECRA Fellowship DE200101100.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017. 2

[2] Walid Bousselham, Guillaume Thibault, Lucas Pagano, Archana Machireddy, Joe Gray, Young Hwan Chang, and Xubo Song. Efficient self-ensemble framework for semantic segmentation. *arXiv preprint arXiv:2111.13280*, 2021. 3, 6, 7

[3] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3640–3649, 2016. 1, 3

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision*, pages 801–818, 2018. 7

[5] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1290–1299, 2022. 6, 7

[6] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in Neural Information Processing Systems*, 34, 2021. 3, 7

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016. 6

[8] Henghui Ding, Xudong Jiang, Bing Shuai, Ai Qun Liu, and Gang Wang. Context contrasted feature and gated multi-scale aggregation for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2393–2402, 2018. 1, 3, 5

[9] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. *arXiv preprint arXiv:2107.00652*, 2021. 3

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*, 2021. 3

[11] Jiaqi Gu, Hyoukjun Kwon, Dilin Wang, Wei Ye, Meng Li, Yu-Hsin Chen, Liangzhen Lai, Vikas Chandra, and David Z Pan. Multi-scale high-resolution vision transformer for semantic segmentation. *arXiv preprint arXiv:2111.01236*, 2021. 3

[12] Jiaqi Gu, Hyoukjun Kwon, Dilin Wang, Wei Ye, Meng Li, Yu-Hsin Chen, Liangzhen Lai, Vikas Chandra, and David Z Pan. Multi-scale high-resolution vision transformer for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12094–12103, 2022. 7

[13] Youngwan Lee, Jonghee Kim, Jeffrey Willette, and Sung Ju Hwang. Mpvit: Multi-path vision transformer for dense prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7287–7296, 2022. 3

[14] Xiangtai Li, Houlong Zhao, Lei Han, and Yunhai Tong. Gated fully fusion for semantic segmentation. In *AAAI*, 2020. 1, 3

[15] Xiaodan Liang, Xiaohui Shen, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with graph lstm. In *Proceedings of the European Conference on Computer Vision*, pages 125–143. Springer, 2016. 1

[16] Fangjian Lin, Sitong Wu, Yizhe Ma, and Shengwei Tian. Full-scale selective transformer for semantic segmentation. In *Proceedings of the Asian Conference on Computer Vision*, pages 2663–2679, 2022. 3

[17] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1925–1934, 2017. 2

[18] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 1, 3, 5, 7

[19] Sun-Ao Liu, Hongtao Xie, Hai Xu, Yongdong Zhang, and Qi Tian. Partial class activation attention for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16836–16845, 2022. 7

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 1, 2, 3, 6, 7, 8

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 2

[22] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 891–898, 2014. 6

[23] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *Proceedings of the European Conference on Computer Vision*, pages 300–317, 2018. 1

[24] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528, 2015. 2

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019. 6

[26] Zipeng Qin, Jianbo Liu, Xiaolin Zhang, Maoqing Tian, Aojun Zhou, Shuai Yi, and Hongsheng Li. Pyramid fusion transformer for semantic segmentation. *arXiv preprint arXiv:2201.04019*, 2022. 3, 7

[27] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188, 2021. 3

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[29] Hengcan Shi, Munawar Hayat, Yicheng Wu, and Jianfei Cai. Proposalclip: unsupervised open-category object proposal generation via exploiting clip cues. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9611–9620, 2022. 1

[30] Hengcan Shi, Hongliang Li, Fanman Meng, and Qingbo Wu. Key-word-aware network for referring expression image segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 38–54, 2018. 1

[31] Hengcan Shi, Hongliang Li, Fanman Meng, Qingbo Wu, Linfeng Xu, and King Ngi Ngan. Hierarchical parsing net: Semantic scene parsing from global scene to objects. *IEEE Transactions on Multimedia*, 20(10):2670–2682, 2018. 3

[32] Hengcan Shi, Hongliang Li, Qingbo Wu, Fanman Meng, and King N Ngan. Boosting scene parsing performance via reliable scale prediction. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 492–500, 2018. 1, 3

[33] Hengcan Shi, Hongliang Li, Qingbo Wu, and King N Ngan. Query reconstruction network for referring expression image segmentation. *IEEE Transactions on Multimedia*, 2020. 1

[34] Hengcan Shi, Hongliang Li, Qingbo Wu, and Zichen Song. Scene parsing via integrated classification model and variance-based regularization. In *IEEE conference on computer vision and pattern recognition*, 2019. 1

[35] Lin Song, Songyang Zhang, Songtao Liu, Zeming Li, Xuming He, Hongbin Sun, Jian Sun, and Nanning Zheng. Dynamic grained encoder for vision transformers. *Advances in Neural Information Processing Systems*, 34:5770–5783, 2021. 3

[36] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7262–7272, 2021. 3, 7

[37] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 568–578, 2021. 1, 3

[38] Yulin Wang, Rui Huang, Shiji Song, Zeyi Huang, and Gao Huang. Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition. *Advances in Neural Information Processing Systems*, 34:11960–11973, 2021. 3

[39] Sitong Wu, Tianyi Wu, Fangjian Lin, Shengwei Tian, and Guodong Guo. Fully transformer networks for semantic image segmentation. *arXiv preprint arXiv:2106.04108*, 2021. 3

[40] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 418–434, 2018. 7

[41] E Xie, WJ Wang, WH Wang, P Sun, H Xu, D Liang, and P Luo. Segmenting transparent objects in the wild with transformer. In *International Joint Conference on Artificial Intelligence. Proceedings*, 2021. 3

[42] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 3, 7

[43] Haotian Yan, Chuang Zhang, and Ming Wu. Lawin transformer: Improving semantic segmentation transformer with multi-scale representations via large window attention. *arXiv preprint arXiv:2201.01615*, 2022. 3

[44] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *Advances in Neural Information Processing Systems*, 2021. 3

[45] Dafeng Zhang and Xiaobing Wang. Dynamic multi-scale network for dual-pixel images defocus deblurring with transformer. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2022. 3

[46] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2881–2890, 2017. 1, 3, 7

[47] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021. 1, 3, 7

[48] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 6

[49] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2021. 3, 7