TAGNet: Learning Configurable Context Pathways for Semantic Segmentation

Di Lin, *Member, IEEE*, Dingguo Shen, Yuanfeng Ji, Siting Shen, Mingrui Xie, Wei Feng, *Member, IEEE*, and Hui Huang, *Senior Member, IEEE*

Abstract—State-of-the-art semantic segmentation methods capture the relationship between pixels to facilitate contextual information exchange. Advanced methods utilize fixed pathways for context exchange, lacking the flexibility to harness the most relevant context for each pixel. In this paper, we present Configurable Context Pathways (CCPs), a novel model for establishing pathways for augmenting contextual information. In contrast to previous pathway models, CCPs are learned, leveraging configurable regions to form information flows between pairs of pixels. We propose TAGNet to adaptively configure the regions, which span over the entire image space, driven by the relationships between the remote pixels. Subsequently, the information flows along the pathways are updated gradually by the information provided by sequences of configurable regions, forming more powerful contextual information. We extensively evaluate the traveling, adaption, and gathering (TAG) stages of our network on the public benchmarks, demonstrating that all of the stages successfully improve the segmentation accuracy and help to surpass the state-of-the-art results. The code package is available at: https://github.com/dilincv/TAGNet.

Index Terms—Semantic segmentation, convolutional neural networks

1 Introduction

Semantic segmentation is a fundamental problem that aims to infer the object label of each pixel in the RGB image. As the large-scale datasets [8], [33] and the high-performance computers with GPUs become available, the performance of semantic segmentation has been greatly advanced by the convolutional neural networks (CNNs) with the powerful network components, such as batch normalization [20], deconvolution [37] and encoder-decoder architectures [31].

The challenge of semantic segmentation mainly lies in the appearance variances and ambiguousness of object categories encoded in the pixel intensities. To tackle this problem with CNNs, it is essential to exchange context between pixels. This is done by learning the representation of each pixel[1] from its surrounding context, where the useful context may lie in the image regions with different sizes or with correlated object categories. Assorted neural network components have been proposed to aggregate the representations of pixels to form the contextual information. The representative models includes spatial pyramid pooling (SPP) models [4], [45], [47], deformable models [7], [9], [10], [21], [41] and attention models [13], [24], [48], [50]. But the structures of existing models are pre-designed and only limited to capture the context of either spatial ranges or category correlation. Therefore, it would be significant to propose a more flexible structure to consider spatial and category context simultaneously.

We unify the contextual models from a graph perspective. In these contextual models, the feature map is mapped into the graph $\mathbb{G} = \{\mathbb{V}, \mathbb{E}\}$, where the pixel of the feature map is denoted as the vertex $v \in \mathbb{V}$. A pair of pixels are connected by the edge $e \in \mathbb{E}$. The edge is associated with a weight. The vertexes and the edges can be used to construct the *information pathway*, which enables the context exchange between the pixels. According to the strategy of constructing the information pathway, the current segmentation methods can be classified into two main families. One is based on the *weighted pathways*, where the edge between the pixel pair plays as the pathway. This family includes SPP models [4], [45], [47], deformable models [7], [9], [10], [21], [41] and non-local attention models [40], [48], as illustrated in Figs. 1a, 1b, and 1c. The other is based on the *augmented pathways*, where the pathway depends on not only the edges but also the pixels within some regions, like the augmented attention model [19] illustrated in Fig. 1d.

Weighted Pathway A straightforward implementation of the weighted pathway is the CNN with the fixed convolutional kernels. Compared to the conventional CNN, the SPP model provides different sizes of convolutional kernels with regular grid connections to capture both the local and long-range context (Fig. 1a). To offer more flexibility than the regular connection patterns, the deformable model (see Fig. 1b) learns the offsets (e.g., the offset Δv_j) to find the pixels of

1. The pixel also refers to the neuron of the feature map of CNN, where the feature map may have lower-resolution than the image.

- *Di Lin and Wei Feng are with the College of Intelligence and Computing, Tianjin University, Tianjin 300072, China. E-mail: ande.lin1988@gmail.com, wfeng@ieee.org.*
- *Dingguo Shen, Yuanfeng Ji, Siting Shen, Mingrui Xie, and Hui Huang are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, Guangdong 518060, China. E-mail: {qq237942920, jyuanfeng8, sitingshen1, xie744538409, hhzhiyan}@gmail.com.*

Manuscript received 11 January 2021; revised 31 July 2021; accepted 26 March 2022. Date of publication 26 April 2022; date of current version 6 January 2023. This work was supported in part by the National Key R & D Program of China under Grant 2020YFC1522700, in part by NSFC under Grant 62072334, and in part by DEGP Key Project under Grants 2018KZDXM058 and 2020SFKC059.
(Corresponding author: Wei Feng.)
Recommended for acceptance by H. Ling.
Digital Object Identifier no. 10.1109/TPAMI.2022.3165034

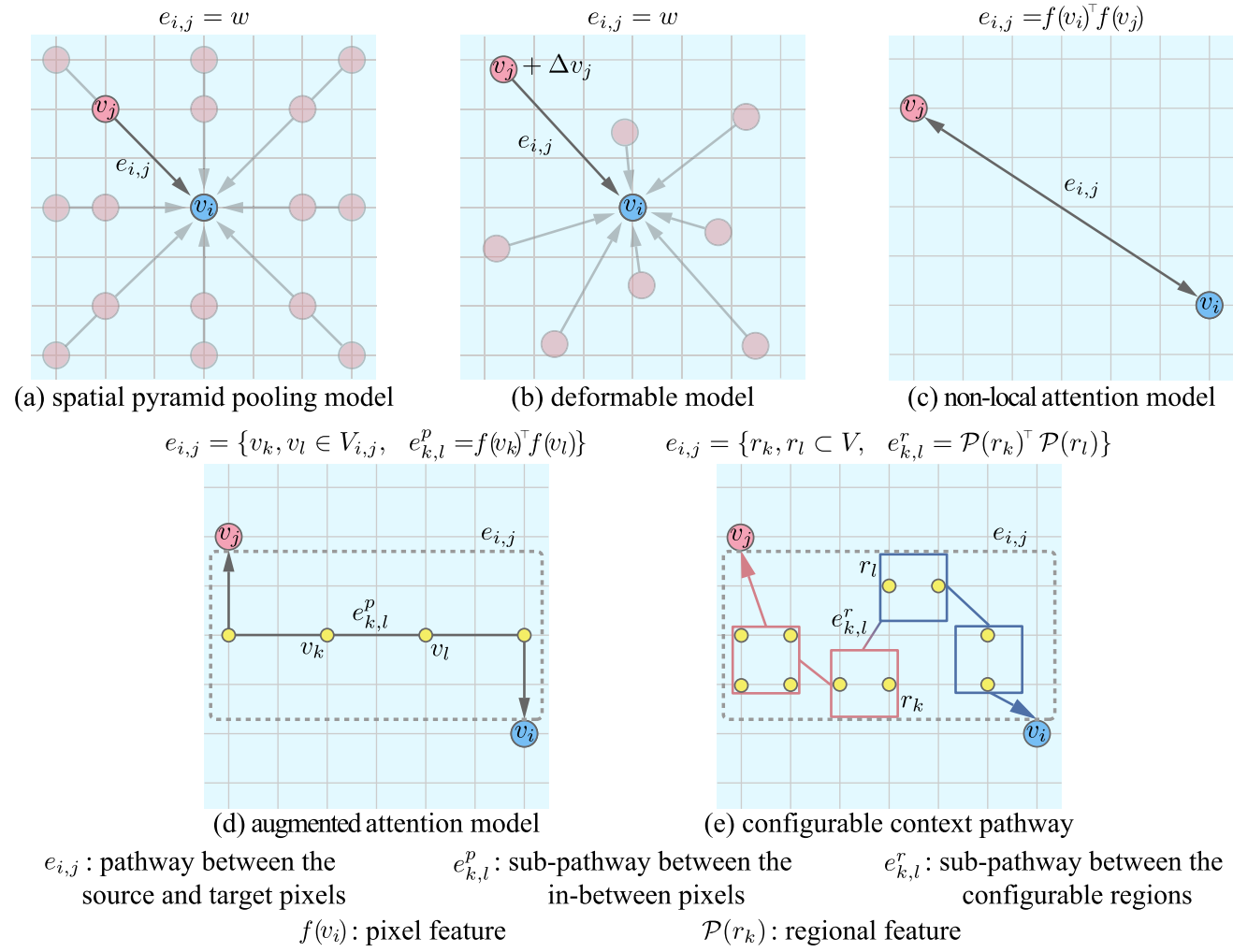


Fig. 1. Different weighted pathways (a–c) and augmented pathway (d) for context exchange between the pixels. The SPP (a) and deformable models (b) formulate the pathway as the shared weight w of the convolutional kernel. The non-local attention model (c) uses the correlation between the pixels to construct the pathway. The augmented attention model (d) uses the prescribed pathway that includes the in-between pixels (e.g., the yellow dots v_k and v_l), which are connected by the sub-pathway (e.g., $e^p_{k,l}$). The sub-pathway is associated with the correlation between the connected pixels. We learn the pathway (e) passing through a series of configurable regions (e.g., the regions r_k and r_l), which are connected by the sub-pathway (e.g., $e^r_{k,l}$) associated with the correlation between the regions.

interest (e.g., the red dot $v_j + \Delta v_j$), which distribute over the entire image. Compared to SPP and deformable models that use the shared weights of the convolutional kernels, the non-local attention model (see Fig. 1c) uses the dot-product coefficient of the pixel features. The coefficient plays as the dynamic weight of the pathway and adaptively controls the context exchange between pixels.

Augmented Pathway The latest attention models [19], [52] proposed the augmented pathway, which includes more in-between pixels (shown in Fig. 1d). Along the augmented pathway $e_{i,j}$, the information flow diffuses from v_j to v_i along the pathway, where the visual information of the in-between pixels (the yellow dots) is also encoded into the information flow. Finally, v_i is enhanced by the information from v_j with the in-between pixels along the whole pathway. But these pathways are defined based on the in-between pixels on the regular grid. They inevitably introduce irrelevant content into the information flow. Furthermore, it may cause considerable computational cost when the length of the pathway is increasing for finding the long-range context.

Configurable Context Pathway To achieve better context exchange, we propose Configurable Context Pathway (CCP), which is a general augmented pathway model with a more flexible structure to capture and propagate context. Comparing to previous models that use redundant pixels along the prescribed augmented pathway, CCP builds the bidirectional pathway depending on the pixels of both ends and multiple in-between configurable regions (i.e., the blue and red regions in Fig. 1e). The in-between regions contain multiple pixels. The adjacent regions are connected by the sub-pathway (e.g., $e^r_{k,l}$ between the regions r_k and r_l), whose weight is computed by the dot-product between the regional features (e.g., $\mathcal{P}(r_k)$ and $\mathcal{P}(r_l)$).

We propose the Travel-Adapt-Gather network (TAGNet) to construct CCPs. The network achieves better context exchange via learning the configurable regions to form the flexible information pathways, as illustrated in Fig. 2. In TAGNet, the source pixel and the target pixel are connected by the bidirectional information traffic along the pathway, which is constructed in the *traveling*, *adaption* and *gathering* stages.

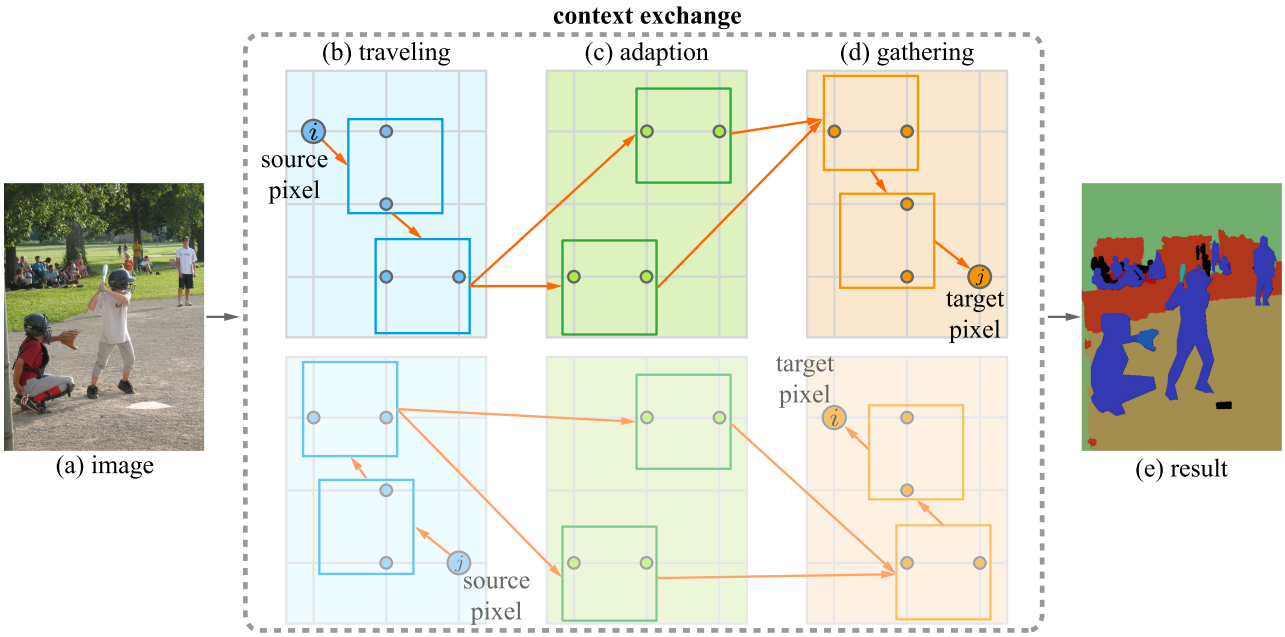


Fig. 2. The illustration of the context exchange between source and target pixels. In the traveling stage (b), each source pixel (blue dot) on the backbone feature map propagates its feature along a sequence of regions (blue rectangles), which enrich the context and yielding the intermediate enriched feature map. During the adapting stage (c), the traveling feature is passed through the fusion regions (green rectangles), where the regional content is updated. In the gathering stage (d), the information of the fusion region is gathered at the target pixel (orange dot). This stage uses the reverse sequence of regions (orange rectangles) and forms the gathering map for segmentation (e). Here, we only show a pair of source and target pixels for brevity.

In the **traveling** stage (Fig. 2b), the feature of the source pixel (the bigger blue dot) is traveled along the associated sub-pathway determined by the configurable image regions in blue rectangles. This stage encodes the regional information into the feature traffic and progressively updates the contents of the configurable regions along the sub-pathway. In the *adaption* stage (Fig. 2c), the feature traffics from the source pixels are effectively merged and enhanced by a few fusion regions (the green rectangles). In contrast to the previous attention models [19], [40], [48] that connect each pair of pixels, the merge of feature traffics saves computation while attending to the important context of the image relevant to the source and target pixels. In the *gathering* stage (Fig. 2d), all of the merged feature traffics move along the sub-pathway of the target pixel (the bigger orange dot). Again, we inject the regional contents of the sub-pathway into the feature traffics and yield the augmented features of the target pixels. The pixels can propagate information along the same pathway in a reverse direction when the roles of the source and the target are exchanged, as illustrated in the dashed box of Fig. 2. The augmented features are used for computing the final segmentation result (Fig. 2d).

TAGNet can be equipped to the general segmentation frameworks (e.g., the encoder-decoder architecture in Fig. 3). We conduct the extensive evaluation on the challenging benchmarks for semantic segmentation, including Cityscapes dataset [6], PASCAL VOC 2012 dataset [12] and COCO-Stuff dataset [1]. The results demonstrate the effectiveness of CCP and TAGNet.

Our contributions are summarized as follows:

- We advocate a more general pathway, Configurable Context Pathway (CCP) that has a more flexible structure for context exchange.
- We propose TAGNet for constructing the bidirectional CCP between the pair of pixels.
- TAGNet helps to achieve the competitive results on multiple segmentation benchmarks (i.e., Cityscapes dataset [6], PASCAL VOC 2012 dataset [12] and COCO-Stuff dataset [1]).

2 Related Work

Deep CNNs [17], [23], [39] learn from large-scale image data [8], [33] and extract powerful features for semantic segmentation. Recent semantic segmentation works [4], [9], [13], [19], [21], [35], [41], [45], [47], [48] widely exploit rich context representations that focus on the relationship between pixels, achieving tremendous progress in the segmentation accuracy. In this paper, we mainly survey two aspects, i.e., *Context Propagation* and *Context Augmentation*, which have been explored by most of the existing works to compute contextual information for segmentation.

Context Propagation Context propagation refers to how pixels communicate to form a useful context representation. Many approaches have propagated contextual information between pixels across different ranges. Given the convolutional feature maps, Zhao *et al.* [47], Yuan *et al.* [45] and Peng *et al.* [38] apply spatial pyramid pooling (SPP), which utilizes different sizes of convolutional and pooling kernels, to extract the context representations. Chen *et al.* [3], [4] employ atrous spatial pyramid pooling (ASPP) to also enable different ranges of context propagation. SPP and ASPP pooling generally are applied on top of the highest-level layer, at the cost of missing visual details contained in lower-level layers. To address this problem, Lin *et al.* [31], Peng *et al.* [38] and Chen *et al.* [5] use encoder-decoder networks, which follow the top-down direction to gradually

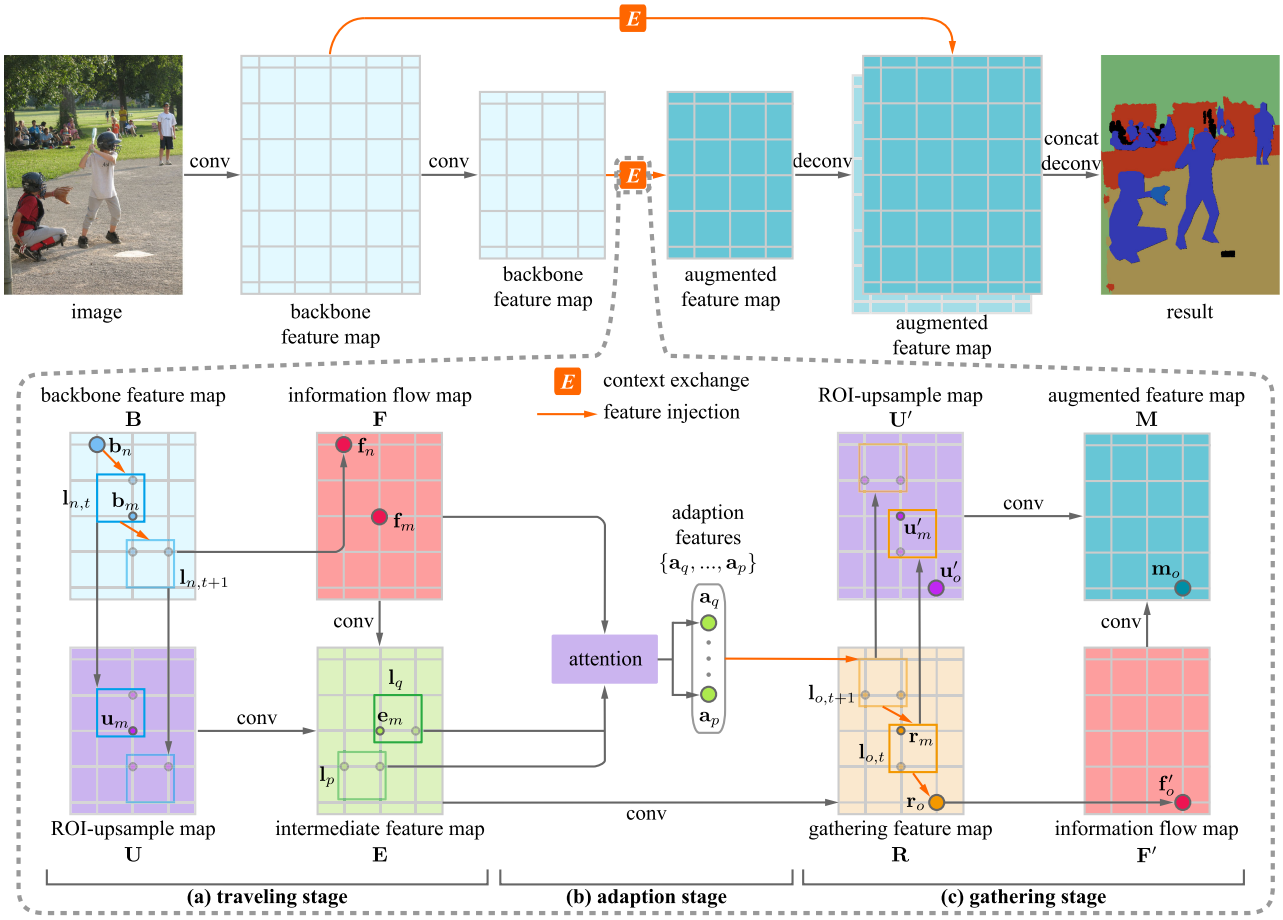


Fig. 3. The encoder-decoder architecture of our approach. We input the image into the encoder network, producing the backbone feature maps at different levels. Each backbone feature map is fed into the module of context exchange, to compute the augmented feature map. The context exchange, which consists of the context traveling, adaption, and gathering, is detailed in the dashed box at the bottom. During the context exchange, the orange arrow indicates the feature injection. Each small rectangle in blue/orange denotes an enriched feature, which is produced after feature injection.

mix the high- and low-level contextual information. All of these works propagate contextual information between pixels in fixed ranges.

In this paper, we allow pixels across the entire image to communicate. We do this by flexibly configuring pathways between pixels to exchange contextual information. Dai *et al.* [7] and Zhu *et al.* [51] propose the deformable convolution to connect pixels beyond fixed ranges, for more accurate recognition of variant object shapes. Yang *et al.* [41], Ji *et al.* [21], Deng *et al.* [9] and Ding *et al.* [10] apply the deformable convolution to enable a vast context propagation. However, the deformable convolution [7], [9], [21], [41] propagates context in a one-way manner. In contrast, our context propagation between pixels is bidirectional. Thus, it uses the relationship between pixels to better configure the information pathways.

Context Augmentation. Context augmentation takes place during context propagation between pixels. It captures higher-order relationships between pixels to enrich contextual information. Recently, many works on semantic segmentation have paid significant attention to context augmentation. This is typically done by incorporating pixels in the middle of information pathways, which are used for context propagation. Zheng *et al.* [49], Liu *et al.* [34] and Lin *et al.* [32] integrate graphical models with deep networks. Graphical models densely connect pixels with information pathways, along which pixels in the middle capture the long-range dependency for context augmentation. Because graphical models use predefined pathways, they inevitably involve irrelevant content to model inter-pixel relationships.

The latest attention models [13], [19], [24], [40], [48], [50], [52] can adjust the importance of pixels. Wang *et al.* [40] and Zhao *et al.* [48] use the non-local model to estimate the correlation between each pair of pixels, but this consumes much GPU memory. Several works [44], [50], [52] have focused on reducing the number of connections between pixels to save computation. Zhu *et al.* [52] build the connection between the query pixel and the representative key pixel. Zhong *et al.* [50] and Yuan *et al.* [44] model the relationship between image regions rather than pixels. The important pixels in the local range may be unhelpful when modeling the long-range dependencies. They sometimes lead to overly long pathways that distract the useful contextual information.

Unlike previous methods [13], [19], [32], [34], [48], [49], our information pathways function as bidirectional shortcuts to propagate contextual information between pixels. Compared with predefined shortcuts investigated in existing works [29], [30], [48], we configure our pathways to select effective content at different ranges. In contrast to predefined pathways, image-adapted pathways determined by superpixels [16], [26], [28], [29] have been used for context augmentation. In these works, multiple streams of contextual

information are directly summed at a pixel. Information streams with different properties, however, may contain inconsistent content, leading to a negative effect on context augmentation. In our approach, we use an adaptive path fusion to effectively combine different information streams, providing more enhanced contextual information for each pixel.

3 Overview

We propose TAGNet, which adaptively configures CCP between the neurons (also referred to as "pixels" in the following) to exchange context. The network consists of the *Travel-Adapt-Gather* (TAG) stages to enhance the traveling information by aggregating information from image regions, as shown in Fig. 2. First, the source pixel *travel* its deep feature along a sequence of regions and produces an information flow. Second, we *adapt* different information flows, which are passed through the configurable fusion regions and merged into fewer but more representative information flows. Finally, the representative information flows are *gathered* at the target pixel, along the associated region sequences. It yields the information-augmented feature map.

We equip TAGNet to the encoder-decoder architecture as shown in Fig. 3, where the TAG stages for context exchange is in the dashed box. The network uses the convolutional layers to encode the image content into multiple layers of backbone feature maps. We use the TAG stages to process each backbone map and compute the augmented feature map. The low-resolution augmented feature map is deconvolved and concatenated with the higher-resolution augmented feature map for the final segmentation. More details of TAGNet are presented in Section 4.

3.1 Notations

For better clarification, we use the capital bold letter $\mathbf{X}$ (e.g., $\mathbf{B}$ and $\mathbf{F}'$) to denote the feature map at the traveling/adaption/gathering stage, where the feature map[2] $\mathbf{X} \in \mathbb{R}^{(H\times W)\times C}$. We use the lowercase bold letter $\mathbf{x}$ (e.g., $\mathbf{b}_n$ and $\mathbf{f}'_n$) to denote the feature vector, the location of region or the attention weight. More detailed descriptions of the important notations in this paper are listed in Table 1.

TABLE 1
Important Notations Used in This Paper

$\mathbf{X} \in \mathbb{R}^{(H\times W)\times C}$: **feature map at different stages**	
$\mathbf{B}, \mathbf{F}, \mathbf{U}, \mathbf{E}$	feature map at the traveling stage
$\mathbf{R}, \mathbf{F}', \mathbf{U}', \mathbf{M}$	feature map at the gathering stage
$\mathbf{x}_n \in \mathbb{R}^C$: **feature vector at different stages**	
$\mathbf{b}_n, \mathbf{f}_n, \mathbf{u}_m, \mathbf{e}_m$	feature vector at the traveling stage
$\mathbf{a}_q$	feature vector at the adaption stage
$\mathbf{r}_m, \mathbf{f}'_o, \mathbf{u}'_m, \mathbf{m}_o$	feature vector at the gathering stage
$\mathbf{f}_{n,t}, \mathbf{f}'_{o,t}$	enriched feature of the t^{th} region of the n^{th} (o^{th}) pixel on the feature map $\mathbf{F}$ ($\mathbf{F}'$)
others	
$\mathbf{l}_{n,t}, \mathbf{l}_{o,t} \in \mathbb{R}^2$	the location of the t^{th} region of the n^{th} (o^{th}) pixel on the feature map $\mathbf{B}$ ($\mathbf{R}$)
$\mathbf{l}_q \in \mathbb{R}^2$	the location of the q^{th} fusion region on the feature map $\mathbf{E}$
$\mathbf{W}_q \in \mathbb{R}^{(H\times W)}$	the attention weight map of the q^{th} fusion region
$w_{q,n} \in (0, 1)$	the attention weight of the n^{th} pixel on the attention weight map $\mathbf{w}_q$

3.2 Overview of TAG Stages

*Traveling Stage*We denote the traveling stage as $\mathbb{T}$ (see Fig. 3a). Let $\mathbf{B}, \mathbf{F}, \mathbf{U}, \mathbf{E} \in \mathbb{R}^{(H\times W)\times C}$ be the backbone feature map, information flow map, ROI-upsample map and intermediate feature map, respectively; $\mathbf{b}_n, \mathbf{f}_n, \mathbf{u}_m, \mathbf{e}_m \in \mathbb{R}^C$ be the feature vectors of the n^{th} (m^{th}) pixels on the maps $\mathbf{B}, \mathbf{F}, \mathbf{U}, \mathbf{E}$. We denote $\mathbf{f}_{n,t} \in \mathbb{R}^C$ as the enriched feature of t^{th} region of the n^{th} pixel located on the map $\mathbf{B}$. $\mathbf{\Phi}_m$ denotes a set of configurable regions, which commonly contain the m^{th} pixel located on $\mathbf{B}$. These configurable regions may belong to different region sequences.

We learn the region sequence $\{\mathbf{l}_{n,1}, \ldots, \mathbf{l}_{n,T}\}$ from the backbone feature vector $\mathbf{b}_n$. This sequence plays as the sub-pathway of the n^{th} pixel on the backbone map $\mathbf{B}$. We travel the feature $\mathbf{b}_n$ along the sub-pathway, computing the $\mathbf{F}, \mathbf{U}, \mathbf{E}$ by the operations $\mathbb{T}_1, \mathbb{T}_2, \mathbb{T}_3$ as:

$$\mathbf{f}_n = \mathbb{T}_1(\mathbf{b}_n,\ \{\mathcal{P}(\mathbf{l}_{n,1}, \mathbf{B}), \ldots, \mathcal{P}(\mathbf{l}_{n,T}, \mathbf{B})\}), \tag{1}$$

$$\mathbf{u}_m = \mathbb{T}_2(\{\mathbf{f}_{n,t} \mid \mathbf{l}_{n,t} \in \mathbf{\Phi}_m\}), \tag{2}$$

$$\mathbf{e}_m = \mathbb{T}_3(\mathbf{f}_m,\ \mathbf{u}_m), \tag{3}$$

where

- $\mathbb{T}_1$ uses the Precise ROI Pooling[3] $\mathcal{P}$ [22] to extract regional features $\{\mathcal{P}(\mathbf{l}_{n,1}, \mathbf{B}), \ldots, \mathcal{P}(\mathbf{l}_{n,T}, \mathbf{B})\}$, where $\mathcal{P}(\mathbf{l}_{n,t}, \mathbf{B}) \in \mathbb{R}^C$, from the $1^{st} \sim T^{th}$ regions of the n^{th} pixel on $\mathbf{B}$. After injecting these regional features into the information flow, we achieve the feature $\mathbf{f}_n \in \mathbf{F}$.
- $\mathbb{T}_2$ uses the enriched feature $\mathbf{f}_{n,t}$, which is associated with the region centered at $\mathbf{l}_{n,t} \in \mathbf{\Phi}_m$ on $\mathbf{B}$, to compute the feature $\mathbf{u}_m \in \mathbf{U}$ that accounts for the relationship between the regional contents.
- $\mathbb{T}_3$ combines $\mathbf{f}_m \in \mathbf{F}$ and $\mathbf{u}_m \in \mathbf{U}$ to yield the intermediate feature $\mathbf{e}_m \in \mathbf{E}$.

Adaption Stage We denote the adaption stage as $\mathbb{A}$ (see Fig. 3b). On the intermediate feature map $\mathbf{E}$, each pixel comprehensively accounts for the impact of different image

2. $H \times W$ represents the spatial dimensions and C is the number of feature channels. We squeeze the spatial dimensions by using the notation $(H \times W)$, allowing the single index (e.g., n, m and o) to indicate the pixel.

3. For a concise presentation in the paper, we abbreviate the term "Precise ROI Pooling" as "ROIpooling" in the text, or "ROI-pool" in the figures.

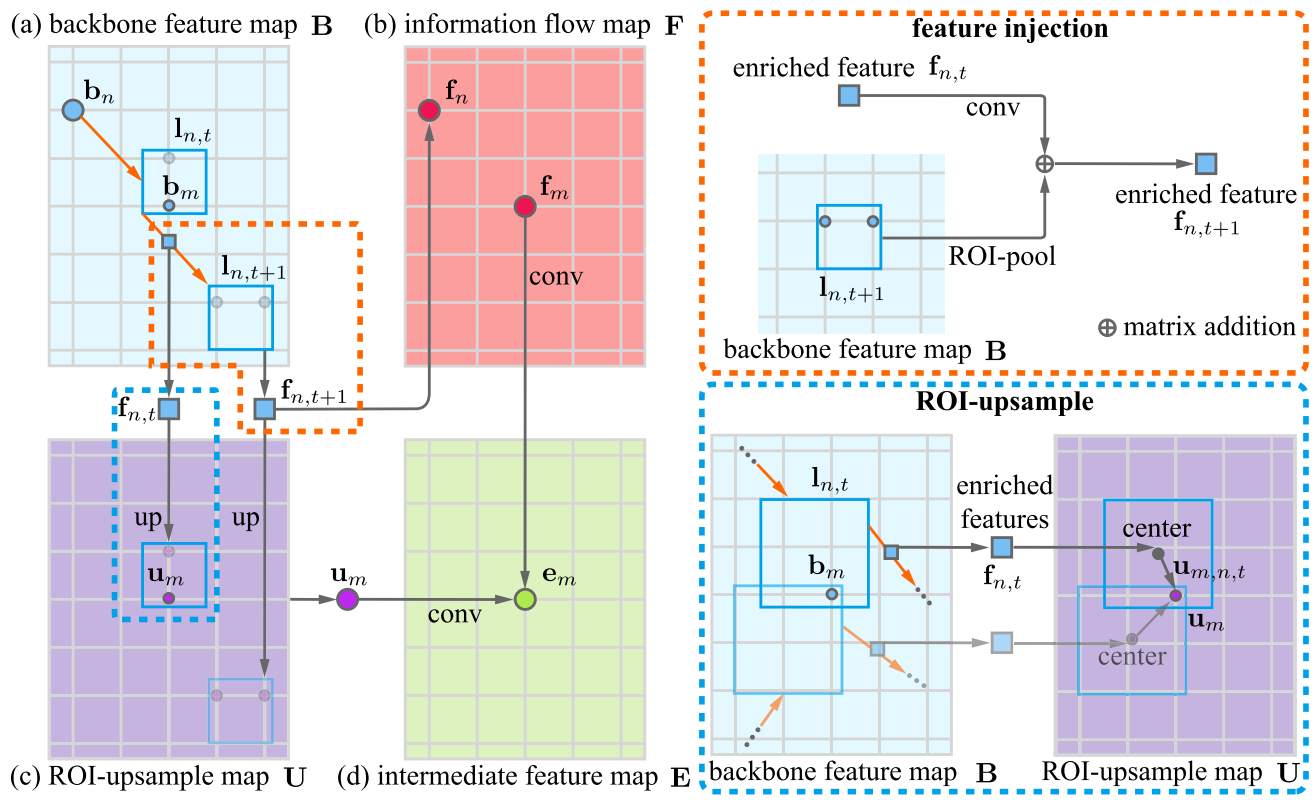


Fig. 4. Given the backbone feature map (a), we conduct information traveling to produce the information flow map (b) and the ROI-upsample map (c). We combine (b) and (c) to achieve the intermediate feature map (d). We provide the details of the feature injection and ROI-upsample in the orange and blue boxes, respectively. For brevity, we only show two sequential regions for the source pixel $\mathbf{b}_n$ on the backbone feature map **B**, where the location $\mathbf{l}_{n,t}$ (or $\mathbf{l}_{n,t+1}$) indicates the center of the first (or the last) configurable region for $\mathbf{b}_n$. The context traveling is done from the first to the last configurable regions.

regions, which provides rich context for achieving the representative contents of the image. Based on **E**, we learn a set of fusion regions, whose centers are $\{\mathbf{l}_1, \ldots, \mathbf{l}_Q\}$ $(Q < H \times W)$.

At the adaption stage, we use **E** along with **F** to achieve a set of adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$, where:

$$\mathbf{a}_q = \mathbb{A}(\mathcal{P}(\mathbf{l}_q, \mathbf{E}),\ \mathbf{F}). \tag{4}$$

Here, we employ ROI-pooling to extract the regional feature $\mathcal{P}(\mathbf{l}_q, \mathbf{E}) \in \mathbb{R}^C$ from the q^{th} fusion region on **E**. Then, we combine the information of the regional feature $\mathcal{P}(\mathbf{l}_q, \mathbf{E})$ and the information flow map **F**, achieving the adaption feature $\mathbf{a}_q$.

Gathering Stage. We denote the gathering stage as $\mathbb{G}$ (see Fig. 3c). Let $\mathbf{R}, \mathbf{M} \in \mathbb{R}^{(H\times W)\times C}$ be the map of gathering features, augmented features; $\mathbf{F}', \mathbf{U}' \in \mathbb{R}^{(H\times W)\times C}$ be the information flow map and the ROI-upsample map at the gathering stage; $\mathbf{r}_m, \mathbf{f}'_o, \mathbf{u}'_m, \mathbf{m}_o \in \mathbb{R}^C$ be the feature vectors of the m^{th} (o^{th}) pixel on the maps $\mathbf{R}, \mathbf{F}', \mathbf{U}', \mathbf{M}$. $\mathbf{f}_{o,t} \in \mathbb{R}^C$ is the enriched feature of t^{th} region of the o^{th} pixel located on $\mathbf{F}'$.

We pass **E** to the convolutional layer for computing **R**, where the target pixels are located. We use the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$ and **R** to compute $\mathbf{F}', \mathbf{U}'$ and **M** by the operations $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3$ as:

$$\mathbf{f}'_o = \mathbb{G}_1(\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}, \{\mathcal{P}(\mathbf{l}_{o,T}, \mathbf{R}), \ldots, \mathcal{P}(\mathbf{l}_{o,1}, \mathbf{R})\}), \tag{5}$$

$$\mathbf{u}'_m = \mathbb{G}_2(\{\mathbf{f}'_{o,t} \mid \mathbf{l}_{o,t} \in \mathbf{\Phi}_m\}), \tag{6}$$

$$\mathbf{m}_o = \mathbb{G}_3(\mathbf{f}'_o,\ \mathbf{u}'_o). \tag{7}$$

where

- $\mathbb{G}_1$ propagates the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$ long the sub-pathway defined by the regions $\{\mathbf{l}_{o,T}, \ldots, \mathbf{l}_{o,1}\}$. These regions contribute the regional features $\{\mathcal{P}(\mathbf{l}_{o,T}, \mathbf{R}), \ldots, \mathcal{P}(\mathbf{l}_{o,1}, \mathbf{R})\}$, for enhancing the o^{th} pixel on $\mathbf{F}'$ and yielding the feature $\mathbf{f}'_o \in \mathbf{F}'$.
- $\mathbb{G}_2$ computes $\mathbf{U}'$ by using the operation similar to **U** in Eq. (2), where $\mathbf{u}'_m$ gathers the information from the sub-pathways belonging to different target pixels.
- $\mathbb{G}_3$ combines $\mathbf{F}'$ and $\mathbf{U}'$ to produce **M** for regressing the pixel-wise categories on the image.

4 Architecture of TAGNet

In this section, we elaborate on the architecture of TAGNet, where we employ the traveling, adaption, and gathering stages to construct CCPs for context exchange between pixels.

4.1 Traveling Stage

The traveling stage is illustrated in Fig. 4, which aims to compute the intermediate feature map **E** by combining the information flow map **F** and the ROI-upsample map **U**. At this stage, we use the operations $\mathbb{T}_1$, $\mathbb{T}_2$ and $\mathbb{T}_3$ to learn the maps **F**, **U** and **E** as below.

Formulation of $\mathbb{T}_1$We propose $\mathbb{T}_1$ to compute the information flow feature $\mathbf{f}_n \in \mathbf{F}$ from the backbone feature $\mathbf{b}_n \in \mathbf{B}$. First, we compute a set of centers $\{\mathbf{l}_{n,1}, \ldots, \mathbf{l}_{n,T}\}$ for the configurable regions (see the blue rectangles in Fig. 4a) as:

$$\begin{cases} \mathbf{l}_{n,1} = \mathbf{l}_n, \\ \{\mathbf{l}_{n,2}, \ldots, \mathbf{l}_{n,T}\} = \mathbf{W}^l * \mathbf{b}_n, . \end{cases} \tag{8}$$

where $\mathbf{W}^l$ denotes the parameters of the convolutional layer for regressing the regional centers. We denote $\mathbf{l}_n \in \mathbb{R}^2$ as the location of the n^{th} pixel. These regions constitute the sub-pathway of the n^{th} pixel on $\mathbf{B}$.

Second, for the configurable regions, whose centers are in the set $\{\mathbf{l}_{n,1}, \ldots, \mathbf{l}_{n,T}\}$, we compute a set of enriched features $\{\mathbf{f}_{n,1}, \ldots, \mathbf{f}_{n,T}\}$. As illustrated in the orange box ("feature injection"), we compute the enriched feature $\mathbf{f}_{n,t}$ by injecting the regional feature of the region centered at $\mathbf{l}_{n,t}$, into the information flow along the sub-pathway of the n^{th} pixel on $\mathbf{B}$. This feature injection is formulated as:

$$\begin{cases} \mathbf{f}_{n,1} = \mathbf{b}_n, \\ \mathbf{f}_{n,t+1} = \mathbf{W}_t^f * \mathbf{f}_{n,t} + \mathcal{P}(\mathbf{l}_{n,t+1}, \mathbf{B}), . \end{cases} \tag{9}$$

where $\mathbf{W}_t^f$ denotes the parameters of the convolutional kernel. $\mathcal{P}(\mathbf{f}_{n,t}, \mathbf{B})$ is the regional feature extracted from the region $\mathbf{l}_{n,t}$ on $\mathbf{B}$. $\mathcal{P}$ denotes the ROI-pooling operation, which allows the back-propagation gradients to update $\mathbf{f}_{n,t}$.

Finally, we use Eq. (9) to inject the $1^{st} \sim T^{th}$ regional features into the information flow, achieving the enriched feature $\mathbf{f}_{n,T}$. We set $\mathbf{f}_n = \mathbf{f}_{n,T}$, which represents the n^{th} pixel on the map $\mathbf{F}$ (see Fig. 4b).

Formulation of $\mathbb{T}_2$As illustrated in the blue box ("ROI-upsample") of Fig. 4, we use the operation $\mathbb{T}_2$ to compute the feature $\mathbf{u}_m$ for the m^{th} pixel on the ROI-upsample map $\mathbf{U}$. We formulate $\mathbb{T}_2$ as:

$$\mathbf{u}_m = \sum_{n,t} \mathbf{u}_{m,n,t}, \quad s.t.\ \mathbf{l}_{n,t} \in \mathbf{\Phi}_m. \tag{10}$$

In Eq. (10), we denote $\mathbf{\Phi}_m$ as a set of centers, which belong to the configurable regions (the blue rectangles) covering the m^{th} source pixel on the backbone feature map $\mathbf{B}$. Specifically, to determine the centers in the set $\mathbf{\Phi}_m$, we enumerate the configurable regions, which are predicted by Eq. (8) for all of the source pixels on the backbone feature map $\mathbf{B}$. For a region (e.g., the blue rectangle with the center $\mathbf{l}_{n,t}$ in "ROI-upsample," Fig. 4) that contains the m^{th} source pixel, we put its center into $\mathbf{\Phi}_m$. The enumeration of all of the configurable regions helps to determine the center set for every source pixel. Note that the size of the center set $\mathbf{\Phi}_m$ is equal to the number of configurable regions, which contain the m^{th} source pixel. Different source pixels may have various sizes of center sets.

Given the center $\mathbf{l}_{n,t} \in \mathbf{\Phi}_m$, we use the enriched feature $\mathbf{f}_{n,t}$, which is associate with the corresponding configurable region on $\mathbf{B}$, to represent the center $\mathbf{l}_{n,t}$ on $\mathbf{U}$. We use ROI-upsample [25] to propagate $\mathbf{f}_{n,t}$, from the center to the m^{th} pixel on $\mathbf{U}$. The ROI-upsample respects the distance between the center and the m^{th} pixel, where a smaller/larger distance indicates more/less propagated information of $\mathbf{f}_{n,t}$. This propagation yields the feature $\mathbf{u}_{m,n,t} \in \mathbb{R}^C$, which are summed (i.e., $\{\mathbf{u}_{m,n,t} \mid \mathbf{l}_{n,t} \in \mathbf{\Phi}_m\}$) to produce $\mathbf{u}_m \in \mathbf{U}$.

Formulation of $\mathbb{T}_3$We use the operation $\mathbb{T}_3$ to compute the intermediate feature $\mathbf{f}_m \in \mathbf{E}$. $\mathbb{T}_3$ uses the convolutional layers to reduce the feature channels of $\mathbf{f}_m \in \mathbf{F}$ and $\mathbf{u}_m \in \mathbf{U}$, by a half. Then the output features to are concatenated to form $\mathbf{e}_m \in \mathbf{E}$.

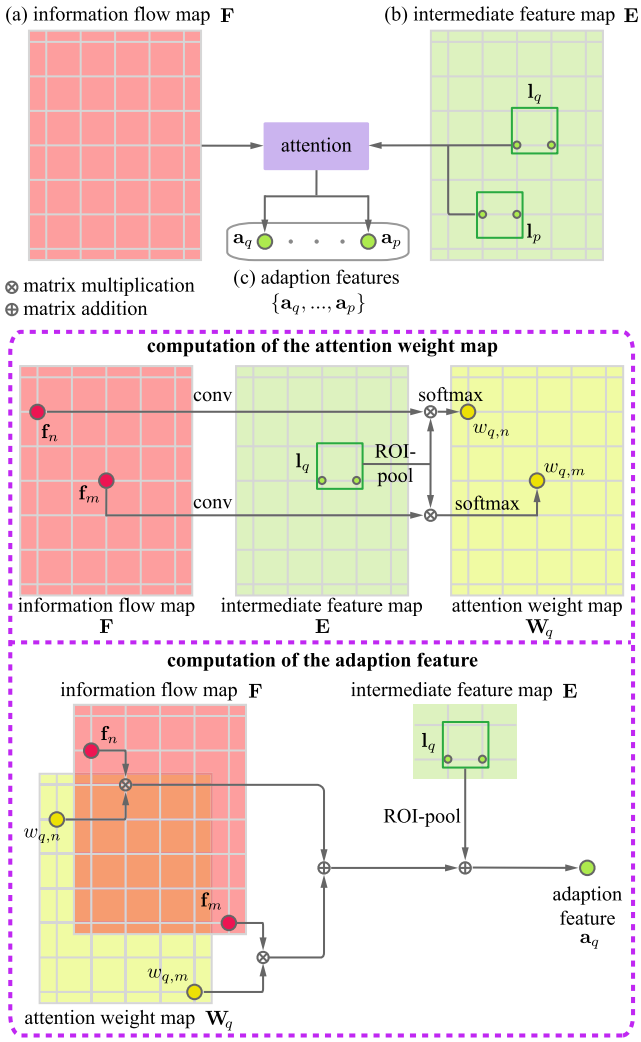


Fig. 5. We feed the information flow map (a) and the intermediate feature map (b) to the attention component that yields the adaption features (c). We provide the details of the attention component in the purple box.

4.2 Adaption Stage

The adaption stage is illustrated in Fig. 5. At this stage, we use the operation $\mathbb{A}$ to compute the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$, based on the information flow map $\mathbf{F}$ and the intermediate feature map $\mathbf{E}$.

At first, we pass $\mathbf{E}$ through the fully connected layer, and compute the centers of Q fusion regions (the green boxes) as:

$$\{\mathbf{l}_1, \ldots, \mathbf{l}_Q\} = \mathbf{W}^e * \mathbf{E}, \tag{11}$$

where $\mathbf{W}^e$ denotes the fully connected layer. $\mathbf{l}_q$ is the center of the q^{th} fusion region. The feature $\mathcal{P}(\mathbf{l}_q, \mathbf{E})$ is extracted from the q^{th} fusion region on $\mathbf{E}$. Then, $\mathbf{F}$, along with all of the regional features of the fusion regions, are put into the attention component, for computing the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$.

In Fig. 5, we detail the attention component in the purple box. Given $\mathbf{F}$ and $\mathcal{P}(\mathbf{l}_q, \mathbf{E})$ (see "computation of the attention weight map"), we compute an attention weight map $\mathbf{W}_q$ for the q^{th} fusion region, where the attention weight $w_{q,n} \in (0, 1)$ is formulated as:

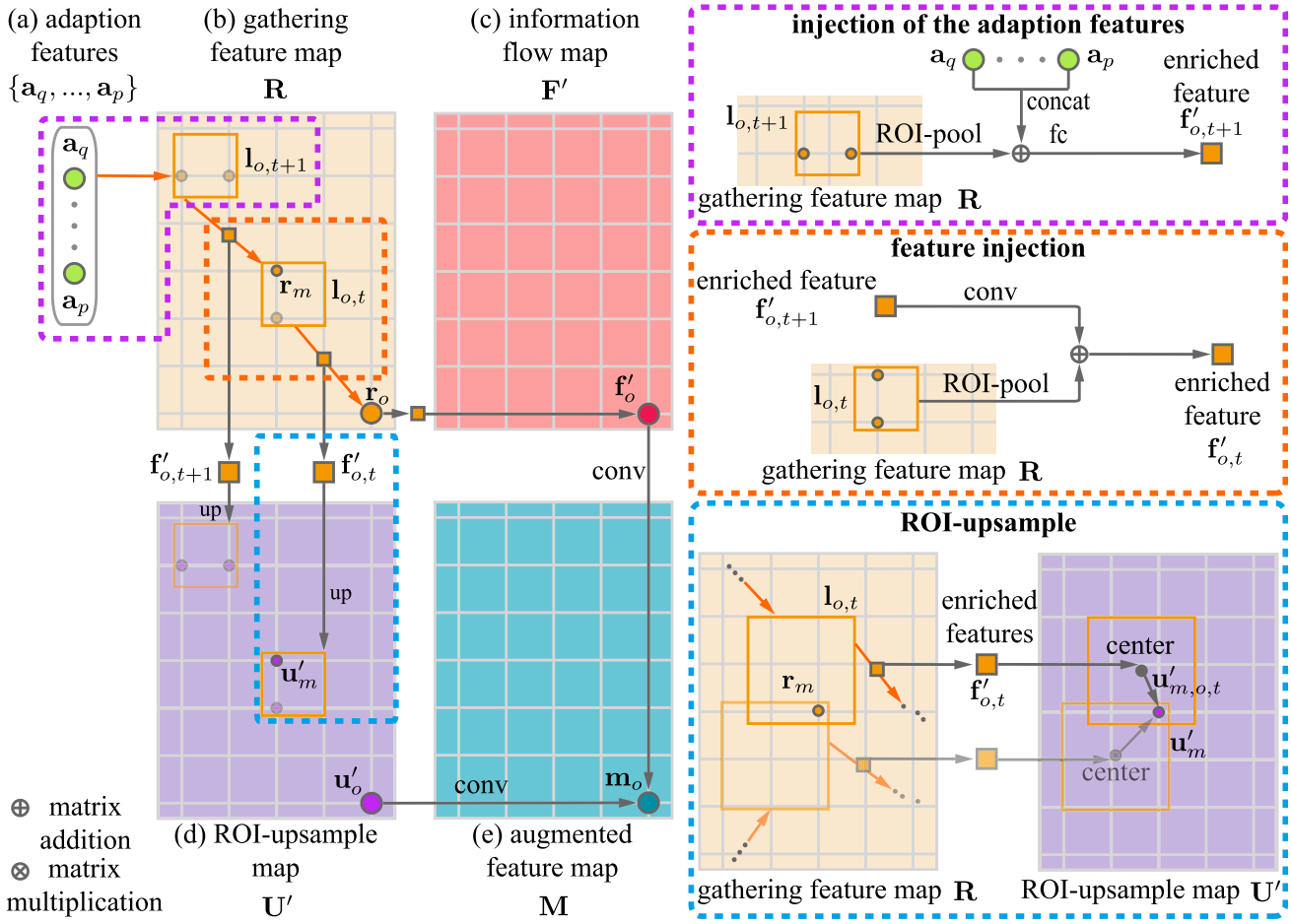


Fig. 6. Given the adaption features (a) and gathering feature map (b), we gather the useful context to compute the information flow map (c) and ROI-upsample map (d). We combine (c) and (d) to form the target pixels on the augmented feature map (e). For brevity, we only show two sequential regions for the target pixel $\mathbf{r}_o$ on the gathering feature map $\mathbf{R}$, where the location $\mathbf{l}_{n,t}$ (or $\mathbf{l}_{n,t+1}$) indicates the center of the first (or the last) configurable region for $\mathbf{r}_o$. The context gathering is done from the last to the first configurable regions.

$$w_{q,n} = \frac{\exp(\mathbf{f}_n^\top \mathcal{P}(\mathbf{l}_q, \mathbf{E}))}{\sum_{m=1}^{H\times W} \exp(\mathbf{f}_m^\top \mathcal{P}(\mathbf{l}_q, \mathbf{E}))}, \quad s.t.\ \mathbf{f}_n, \mathbf{f}_m \in \mathbf{F}. \tag{12}$$

Second, we use $\mathbf{W}_q$ to re-weight all of the features on $\mathbf{F}$ (see "computation of the adaption feature"). The re-weighted features are added to $\mathcal{P}(\mathbf{l}_q, \mathbf{E})$ of the q^{th} fusion region. It forms the adaption feature $\mathbf{a}_q$ as:

$$\mathbf{a}_q = \mathcal{P}(\mathbf{l}_q, \mathbf{E}) + \sum_{n=1}^{H\times W} w_{q,n} \cdot \mathbf{f}_n. \tag{13}$$

With Eq. (13), we compute a set of adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$ for Q fusion regions.

4.3 Gathering Stage

The gathering stage is illustrated in Fig. 6. We propagate the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$ to the target pixels on the gathering feature map $\mathbf{R}$, and compute the information flow map $\mathbf{F}'$ and the ROI-upsample map $\mathbf{U}'$. We use $\mathbf{F}'$ and $\mathbf{U}'$ to form the augmented map $\mathbf{M}$. Below, we introduce the operations $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$ for computing $\mathbf{F}'$, $\mathbf{U}'$ and $\mathbf{M}$.

Formulation of $\mathbb{G}_1$As illustrated in Figs. 6a and 6b, $\mathbb{G}_1$ is used for propagating the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$ to the o^{th} pixel on $\mathbf{R}$, along the sub-pathway defined by the configurable regions centered at $\{\mathbf{l}_{o,T}, \ldots, \mathbf{l}_{o,1}\}$. Note that these centers has been used at the traveling stage for forming a reverse sub-pathway for the o^{th} pixel on $\mathbf{B}$.

We feed $\mathbf{E}$ to the convolutional layer to compute $\mathbf{R}$. In the purple box ("injection of the adaption features"), we concatenate the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$, and feed them to the fully connected layer. The output feature is injected into the regional feature $\mathcal{P}(\mathbf{l}_{o,T}, \mathbf{R})$, which is extracted from the region centered at $\mathbf{l}_{o,T}$ on $\mathbf{R}$. Note that the location $\mathbf{l}_{o,T}$ indicates the region centered at $\mathbf{l}_{o,t+1}$, in "injection of the adaption features," Fig. 6. Given the adaption features $\{\mathbf{a}_1, \ldots, \mathbf{a}_Q\}$ and the regional feature $\mathcal{P}(\mathbf{l}_{o,T}, \mathbf{R})$, we compute the enriched feature $\mathbf{f}'_{o,T}$ as:

$$\mathbf{f}'_{o,T} = \mathbf{W}_o^g * [\mathbf{a}_1, \ldots, \mathbf{a}_Q] + \mathcal{P}(\mathbf{l}_{o,T}, R), \tag{14}$$

where $\mathbf{W}_o^g$ denotes the parameters of the fully connected layer for the o^{th} pixel on $\mathbf{R}$; $[\cdot]$ denotes the operation of feature concatenation.

Next, we propagate $\mathbf{f}'_{o,T}$ along the sub-pathway defined by the centers $\{\mathbf{l}_{o,T-1}, \ldots, \mathbf{l}_{o,1}\}$. As detailed in the orange box ("feature injection"), we compute the set of enriched features $\{\mathbf{f}'_{o,T-1}, \ldots, \mathbf{f}'_{o,1}\}$, where:

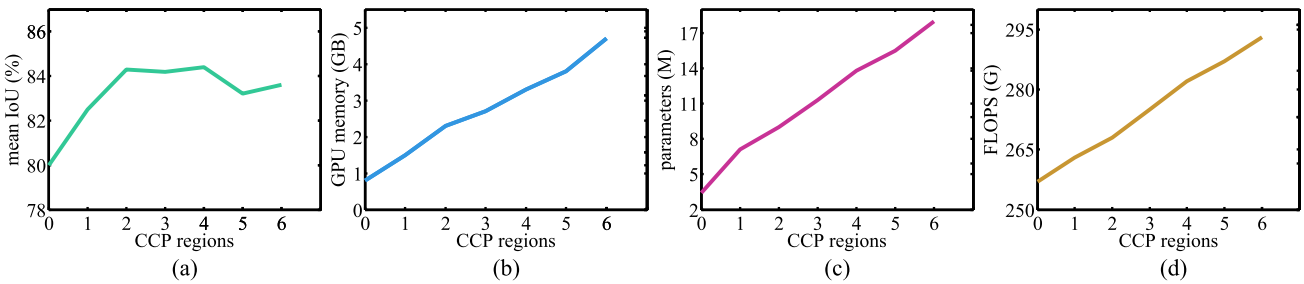


Fig. 7. Sensitivities of the mean IoU (a), the increase in the GPU memory (b), the increase in the number of parameters (c), and the increase in the FLOPS (d) to the number of regions along the CCP. All of the performances are evaluated on the Cityscapes validation set. These computational overheads (b–d) are calculated independently, apart from the computation of the backbone network. The size of the input image is 769×769.

$$\begin{cases} \mathcal{P}(\mathbf{l}_{o,1}, \mathbf{R}) = \mathbf{r}_o, \\ \mathbf{f}'_{o,t} = \mathbf{W}^f_{t+1} * \mathbf{f}'_{o,t+1} + \mathcal{P}(\mathbf{l}_{o,t}, \mathbf{R}), . \end{cases} \tag{15}$$

where $\mathbf{W}^f_{t+1}$ denotes the parameters of the convolutional kernel. The set of parameters, i.e., $\{\mathbf{W}^f_1, \ldots, \mathbf{W}^f_T\}$, has been used at the traveling stage in a reverse order. We reuse these parameters for gathering context, thus decreasing the model complexity.

We inject the regional features of the T^{st} to 1^{th} regions into the information flow along the sub-pathway, yielding the enriched feature $\mathbf{f}'_{o,1}$. We set $\mathbf{f}'_o = \mathbf{f}'_{o,1}$, for representing the o^{th} pixel on $\mathbf{F}'$ (see Fig. 6c).

Formulation of $\mathbb{G}_2$We use $\mathbb{G}_2$ to construct the ROI-upsample map $\mathbf{U}'$ at the gathering stage (see Fig. 6d). We follow the construction of $\mathbf{U}$ at the traveling stage, and propagate the set of enriched features $\{\mathbf{f}'_{o,T}, \ldots, \mathbf{f}'_{o,1}\}$ to $\mathbf{U}'$. As illustrated in the blue box ("ROI-upsample"), the m^{th} pixel is commonly covered by the set $\mathbf{\Phi}_m$ of regions (the orange boxes) on $\mathbf{R}$. We use the enriched feature $\mathbf{f}'_{o,t}$ to represent the regional center $\mathbf{l}_{o,t} \in \mathbf{\Phi}_m$ on $\mathbf{U}'$. We use ROI-upsample to propagate $\mathbf{f}'_{o,t}$, from the center $\mathbf{l}_{o,t}$ to the m^{th} pixel. This propagation yields the feature $\mathbf{u}'_{m,o,t}$. We sum the features $\{\mathbf{u}'_{m,o,t} \mid \mathbf{l}_{o,t} \in \mathbf{\Phi}_m\}$ to produce $\mathbf{u}'_m \in \mathbf{U}'$ as:

$$\mathbf{u}'_m = \sum_{o,t} \mathbf{u}'_{m,o,t}, \quad s.t.\ \mathbf{l}_{o,t} \in \mathbf{\Phi}_m. \tag{16}$$

In contrast to $\mathbf{U}$ at the traveling stage, $\mathbf{U}'$ contains the representative contents that are gathered from the fusion regions on $\mathbf{E}$.

Formulation of $\mathbb{G}_3$$\mathbb{G}_3$ is used for computing the augmented feature map $\mathbf{M}$. Specifically, $\mathbb{G}_3$ reduces the feature channels of $\mathbf{F}'$ and $\mathbf{U}'$ by a half using two convolutional layers. The output feature maps are concatenated to form $\mathbf{M}$ (see Fig. 6e).

5 Implementation Details

The TAGNet in this paper is implemented on the PyTorch platform.[4] The backbone of TAGNet is based on the 8-stride ResNet-101 [17]. We apply the TAG stages on the *res4* and *res5* layers, respectively, producing two augmented feature maps. Different augmented feature maps are aggregated by the encoder-decoder architecture. The number of successive regions is set as $T = 3$ via cross-validation on the training data. Each sub-pathway consists of the end pixel and two regions. For an input image, the number of fusion regions Q is 10‰ of the total number of pixels in the backbone feature map ($Q = 100$) and the size of all regions is 7×7.

The CNN-based backbone is pre-trained on the ImageNet dataset [8]. The parameters of the TAGNet are initialized randomly and then optimized using a standard SGD solver. The training images are resized to 769×769 and augmented with flipping, cropping, scaling, and rotating. During network training, the learning rate is 0.01 and is decayed linearly every 50 K mini-batches where each mini-batch contains 8 images. We applied multi-scale training and testing on each image. For each input image, we use a set of scaling factors (i.e., $\{0.5, 0.75, 1.0, 1.25, 1.5\}$) to achieve the resized versions for training. Given the trained network, we enter the resized images to compute the corresponding predictions, which are averaged to achieve the final segmentation result.

6 Experiment and Analysis

6.1 Datasets for Approach Evaluation

Cityscapes [6] The Cityscapes dataset contains 5 K images (2,975 training, 500 validation, and 1,525 testing images) that are finely annotated with 30 object categories in urban streets. We follow the convention of the Cityscapes dataset and focus on the 19 categories with a mixture of complex objects (e.g., small cars, crowded persons, and irregular-shape roads). We use the Cityscapes [6] dataset for the major evaluation of our method.

PASCAL VOC 2012 [12] We also evaluate our method on the PASCAL VOC 2012 [12] dataset, which contains annotations of 21 object categories. PASCAL VOC 2012 provides 10,582, 1,449 and 1,456 images for training, validation and testing.

COCO-Stuff [1] We provide more evaluation results on the COCO-Stuff-10 K and -164 K datasets. The COCO-Stuff-10 K dataset contains 9 K/1 K images for training/validation, while the COCO-Stuff-164 K dataset contains more images, where about 120 K/5 K images are used for training/validation.

6.2 Sensitive Analysis of TAGNet

6.2.1 Sensitivity to the Regions of CCP

We conduct the sensitive analysis of how the properties of CCP regions influence the performance of TAGNet, in terms of mean Intersection-over-Union (mean IoU).

Length of CCP The length of CCP is determined by the number of regions along the pathway, i.e., the parameter T in traveling and gathering stages. In Fig. 7a, We evaluated

4. https://pytorch.org/

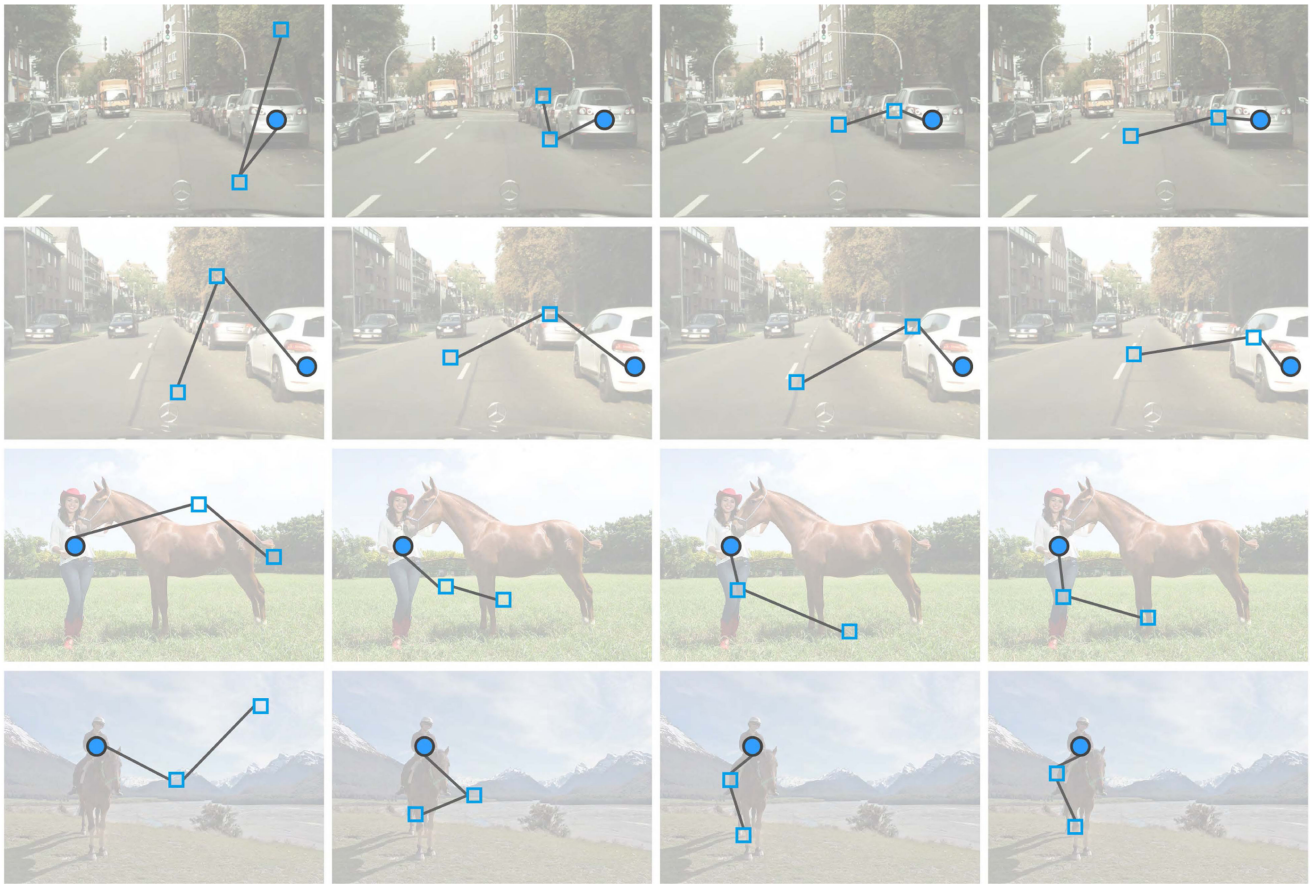


Fig. 8. Visualization of CCPs at different stages of network learning. We take the images from the Cityscapes validation set (the first two rows) and the COCO-Stuff-10 K validation set (the last two rows). For simplicity, only a single pixel and the associated CCP are shown in each image. Each CCP has two localized regions as default.

the segmentation accuracy on the Cityscapes validation set using different number of regions (i.e., $T \in \{1, 2, 3, 4, 5, 6, 7\}$) with default setting of other region hyper-parameters. Here, $T = 1$ means that there is no configurable region along the pathway. In this case, the source pixel, on the backbone feature map, is directly connected to the fusion region on the intermediate feature map.

With $T = 1$, only 80.0% mean IoU is achieved without context augmentation provided by CCP. As more regions are added to each CCP, i.e., increasing T, a considerable improvement on the segmentation performance is achieved, e.g., 84.3% mean IoU when $T = 3$. It demonstrates the effectiveness of CCP. But we find that more regions, e.g., $T = 5, 6$, unnecessarily improve the performance. This is because more regions (large T) increase the model complexity, leading to more difficult training of TAGNet on the limited data. Note that more regions also require more computational overheads, such as the GPU memory, the number of model parameters, and the floating-point operations per second (FLOPS) shown in Figs. 7b, 7c, and 7d. It suggests that we should make a tradeoff for setting the length of CCP. In this paper, $T = 3$ leads to the reasonable performance of TAGNet.

*Visualization of CCP*In Fig. 8, we visualize the learning of CCPs ($T = 3$) in the images. In the first column (Fig. 8a), we select a pixel from each image and show its randomly initialized CCP. From the second to the last column (Figs. 8b, 8c, and 8d), we use three networks trained on 10 K, 30 K, and 50 K mini-batches, respectively, to compute CCPs of the selected pixels. Along each CCP, the first/second regions respectively cover local-range and relatively long-range content. It demonstrates that CCP captures the context in different ranges. In the first/last two rows of Fig. 8, we compare two images that have similar scenes. At the early stages of the network training, the locations of the first and second regions distribute over the entire image but show little correlation. As the network training becomes stable, the first and second regions in similar scenes reside at the objects (e.g., car and road in the first two rows, human and horse in the last two rows), which have semantic/spatial relationship and form consistent context.

Size of the Configurable Regions. The size of the configurable regions along CCP also affects the performance. In Fig. 9a, we conducted the experiment of segmentation using different region sizes, including 1×1, 3×3, 5×5, 7×7, 9×9, 11×11 and 13×13. The results indicate that large regions with richer context information may lead to higher mean IoUs. However, if the region size becomes too large, the configurable regions along CCPs would be easily overlapped and capture similar image content, which yields little improvement on the segmentation accuracy but more computational costs (see the increase of FLOPS in Fig. 9b). In our experiments, the number of network parameters is insensitive to the size of configurable regions, because the

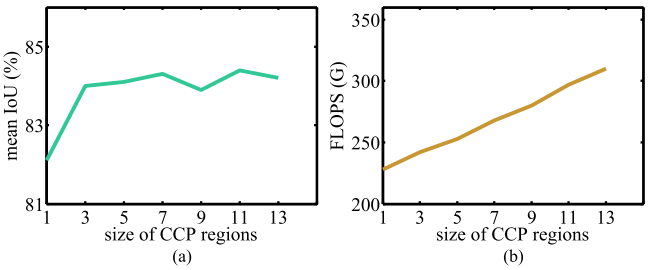


Fig. 9. Sensitivities of the mean IoU (a) and the increase in the FLOPS (b) to the size of regions. We evaluate the results on the Cityscapes validation set.

differentiable operations of ROI-pooling/upsample are non-parametric. Regardless of the size of configurable regions, the ROI-pooling and -upsample operations yield the same dimension of feature maps, thus making little difference to the requirement of GPU memory.

6.2.2 Sensitivity to the Fusion Regions

We examine how the fusion regions affect the model complexity, the computational properties, and the segmentation performance of TAGNet.

Discussion on the Model Complexity. In traditional attention models, the model complexity is largely influenced by the number of links between pixels. For example, non-local attention models [40], [48] is based on the links between each pair of pixels, where an $H \times W$ map produce $\mathcal{O}((H \times W)^2)$ pixel-wise links. Recent attention models [19], [52] disentangle the direct communication between pixels, which reduces the number of pixel-wise links to $\mathcal{O}((H + W - 1) \times (H \times W))$ [19] or $\mathcal{O}(Q \times H \times W)$ [52] ($Q << H \times W$ is the number of representative pixels).

Compared to previous models, the total complexity of constructing CCPs is $\mathcal{O}(2Q \times H \times W)$, where Q is the number of fusion regions. In TAGNet, the adaption stage produces $\mathcal{O}(Q \times H \times W)$ links for all pairs of source pixels and fusion regions, while the gathering stage produces $\mathcal{O}(Q \times H \times W)$ links for all pairs of fusion regions and target pixels. Since the computation of pixel-wise links is implemented by the matrix product, limiting the value of Q is essential to achieve reasonable complexity of the whole model.

Quantitative and Qualitative Analysis. We conduct the quantitative experiment to evaluate how the number of fusion regions Q influences the segmentation performance (mean IoU) and GPU memory, parameters, FLOPS, as shown in Figs. 10a, 10b, 10c, and 10d. The results indicate that as Q increases, mean IoU, GPU memory, parameters and FLOPS also increase. We also made qualitative visualization of fusion regions in Fig. 11. The results show that more fusion regions can better capture the representative image contents for adaption, but too many fusion regions drastically increase FLOPS (Fig. 10d) and slow the network training and testing. Therefore, we should set a proper value of Q to achieve the tradeoff between the segmentation performance and the computational cost. We find that $Q = 100$ or 10‰ of the total number of pixels in the backbone feature maps is appropriate for TAGNet.

In Table 2, we compare the accuracy and computational cost of TAGNet with some latest attentional models, including PSANet [48], CCNet [19] and ANLNet [52]. TAGNet using default hyperparameters achieves the best mean IoU with reasonable GPU memory and FLOPS. Since the parameters of CCPs can be shared at both the traveling and gathering stages, it allows TAGNet achieve a better performance than other methods with fewer parameters. To evaluate the influence of reusing CCPs, we build a TAGNet whose traveling and gathering stages employ separated sets of CCPs without parameter reusing. The result is shown in "TAGNet, separate CCPs," the fifth row of Table 2. TAGNet with separate CCPs requires more GPU memory. More network parameters increase the complexity of network optimization and lead to slight performance degradation, compared to the TAGNet with CCPs reusing.

We make a comparison between TAGNet and the segmentation network that consists of two sets of deformable convolutional blocks [7] with an asymmetric non-local attention [52] in-between, as reported in the fourth row "DCN-ANL-DCN" of Table 2. The "DCN-ANL-DCN" network achieves the configurable pathways to some extent, where a pixel of each pathway obtains the information of its neighbor pixels. Although the "DCN-ANL-DCN" network shares some similarity with our TAGNet, it yields a lower accuracy and more computational cost. The reasons is threefold, which further indicates the effectiveness of the TAG stages.

First, apart from propagating the context of surrounding regions to the source pixel, the traveling stage of TAGNet uses each configurable pathway for distributing the information of the source pixel to the image regions along the pathway. It yields the ROI-upsample map, which is combined with the information flow map to comprehensively capture the relationship between the source pixel and more relevant image regions. Second, compared to DCN, the pathway can be reused at the gathering stage, thus requiring fewer parameters. Third, compared to the regular pooling regions used in

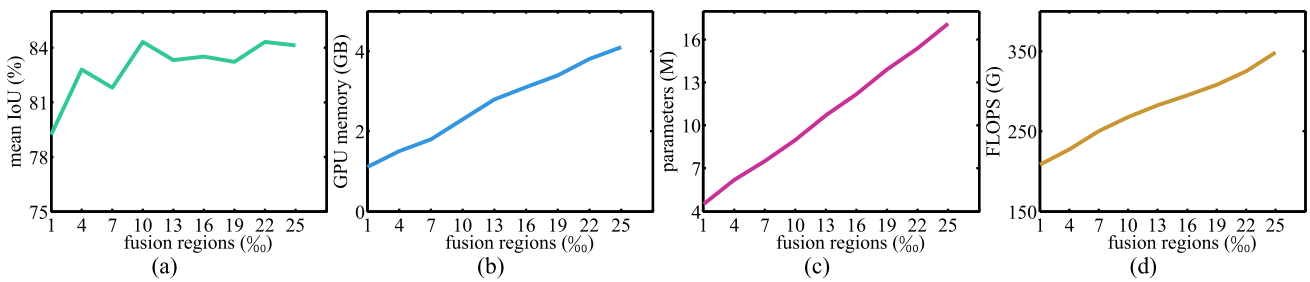


Fig. 10. Sensitivities of the mean IoU (a), the increase in the GPU memory (b), the increase in the number of parameters (c) and the increase in the FLOPS (d) to the number of fusion regions. All performances are evaluated on the Cityscapes validation set. These computational overheads (b–d) are calculated independently, apart from the computation of the backbone network.

Fig. 11. Visualization of learning fusion regions. The images are taken from the Cityscapes validation set (the first two rows) and the COCO-Stuff-10 K validation set (the last two rows). In (b–d), we show 100 fusion regions in each image.

the ANL [52], the adaption stage adaptively configures the fusion regions on the intermediate feature map, which finds more representative image contents to guide the segmentation procedure.

6.2.3 Sensitivity to the Multi-Scale Context

The multi-scale training/testing strategy and different backbone layers can influence the multi-scale context, which is capture by TAGNet.

Multi-Scale Training and Testing. We examined the impact of multiple scales of the input images on the network training and testing, as shown in Table 3. We use the single scale of 1 (i.e., keeping the size of input images) as the baseline (i.e., 83.9% mean IoU). With different multi-scale settings, i.e., $0.75 \sim 1.25$, $0.5 \sim 1.5$ and $0.25 \sim 1.75$, for resizing the input images, we achieve $0.1\% \sim 0.4\%$ mean IoU improvement on the Cityscapes validation set.

Different Backbone Layers. We apply TAG stages on top of different layers (i.e., *res3*∼*res5*) of backbone feature maps, and report the segmentation performances in Table 4. Compared with the single-layer *res5*, the layers *res4* and *res5* with TAG stages improve 0.6% segmentation accuracy. However, adding TAG stages to *res3*∼*res5* yields 0.1% drop, compared to using *res4* and *res5*. This is because *res3*, at the shallower layer, captures insufficient semantic information. Since the high resolution of *res3* layer requires substantial GPU memory, we use the layers *res4* and *res5* in TAGNet.

Discussion on the Multi-Scale Strategies. Using the traditional multi-scale training and testing with multiple layers of backbone feature maps, we can achieve $0.1\% \sim 0.7\%$

TABLE 2
Comparisons With Attention Models

method	▲memory	▲param	▲FLOPS	time	mIoU
PSANet [48]	3.1	51	478	1.1	78.6
CCNet [19]	**1.2**	20	239	0.6	80.2
ANLNet [52]	1.5	14	**163**	**0.4**	80.1
DCN-ANL-DCN [7], [52]	2.8	32	352	1.0	82.0
TAGNet, separate CCPs	2.6	13	271	0.8	84.1
TAGNet	2.3	**9**	268	0.7	**84.3**

▲ *Indicates the Increase in GPU memory/the Number of parameters/FLOPS. We Also Provide the Average Testing Time in Terms of second(s) Per Image. We Evaluate All Performances on the Cityscapes Validation Set*

TABLE 3
Different Strategies of Using Multiple Scales in Network Training and Testing. We Use the Interval of 0.25 in Each Set of Scales. We Report the Segmentation Accuracy on the Cityscapes Validation Set, in Terms of Mean IoU (%)

scale	1	0.75∼1.25	0.5∼1.5	0.25∼1.75
mean IoU	83.9	84.0	**84.3**	84.1

TABLE 4
Different Strategies of Using Multiple Layers of Backbone Feature Maps. We Report the Segmentation Accuracy on the Cityscapes Validation Set, in Terms of Mean IoU (%)

layer	*res5*	*res4∼res5*	*res3∼res5*
mean IoU	83.7	84.3	**84.4**

TABLE 5
Ablation Study of TAG Stages on the Cityscapes Validation Set With Mean IoU (%)

single stage			
traveling	**adaption**	**gathering**	**mean IoU**
			78.2
✓			79.6
	✓		80.4
		✓	79.0
multiple stages			
traveling	**adaption**	**gathering**	**mean IoU**
	✓	✓	81.3
✓		✓	80.7
✓	✓		82.1
✓	✓	✓	**84.3**

TABLE 6
Different Strategies of Using the Enriched Features. We Report the Segmentation Accuracy on the Cityscapes Validation Set, in Terms of Mean IoU (%)

information flow map	ROI-upsample map	mean IoU
		80.4
✓		82.6
	✓	83.5
✓	✓	**84.3**

TABLE 7
Comparisons With State-of-The-Art Methods on the Cityscapes Dataset, in Terms of Mean IoU (%)

validation set		**test set**	
method	**mIoU**	**method**	**mIoU**
Zhao *et al.* [48]	78.6	Ding *et al.* [10]	81.0
Chen *et al.* [4]	79.1	Huang *et al.* [19]	81.4
Chen *et al.* [5]	79.3	Fu *et al.* [13]	81.5
Yuan *et al.* [44]	79.6	Zhu *et al.* [52]	81.8
Chen *et al.* [2]	80.8	Hou *et al.* [18]	82.0
Huang *et al.* [19]	81.3	Chen *et al.* [2]	82.7
Fu *et al.* [13]	81.5	Zhuang *et al.* [53]	82.8
ours	**84.3**	ours	**84.0**

accuracy improvement, as listed in Tables 3 and 4. It should be noted that the flexible configuration of CCP regions is more effective to capture multi-scale information, and more CCP regions produce better segmentation improvement (up to 4.4% mean IoU), as demonstrated in Fig. 7a.

6.3 Analysis on the Architecture of TAGNet

6.3.1 Ablation Study of the TAG Stages

We evaluate the segmentation performance of the TAGNet with a single or multiple combinations of the traveling, adaption, and gathering stages, whose results are shown in Table 5. The encoder-decoder network without all of the three TAG stages is used as the baseline, which achieves 78.2% mean IoU.

Evaluation with Single Stage Given the traveling stage solely, the intermediate feature map is obtained directly for segmentation. Given the adaption stage solely, the backbone network is used to obtain two independent feature maps, which are regarded as the information flow map and intermediate feature map, respectively; a set of adaption features are used as the keys of the non-local attention layer [52] for segmentation. Given the gathering stage only, the adaption features are regarded as the features of all of the source pixels, without selecting any representative image content. The results show that by adding the traveling, adaption, or gathering stage independently to the baseline network, there would be $0.8\% \sim 2.2\%$ mean IoU improvement on the segmentation accuracy. It verifies the effectiveness of each stage.

Evaluation with Multiple Stages Furthermore, ablation studies are conducted to evaluate the effectiveness of different combinations of multiple TAG stages for semantic segmentation, whose results are listed in Table 5 "multiple stages". When all of the TAG stages are used, it achieves 84.3% mean IoU, over 6% improvement to the baseline 78.2% mean IoU. It can be also observed that TAGNet with two or three stages is superior to that with a single stage only.

We disable the single traveling, adaption, or gathering stage to investigate its effect on the segmentation performance. When the traveling stage is disabled, the information of the source pixel passes to the fusion region directly, and the regions containing highly-correlated contents along CCP are missed for enhancing the information flow traveling between the source pixel and the fusion region. It causes 3% drops compared to the full TAGNet. Similarly, when the gathering stage is disabled, there is about 2% performance drop compared to the full TAGNet.

When only the adaption stage is disabled, each pair of source and target pixels are connected without the in-between fusion region, similar to the non-local attention models [40], [48]. Since the TAGNet without adaption stage fails to select the representative image contents to fuse the information flows from different source pixels, it yields a 3.6% performance drop.

6.3.2 Ablation Study of the Enriched Features

We study the enriched features, which are used for computing the information flow maps and the ROI-upsample maps at the traveling and gathering stages.

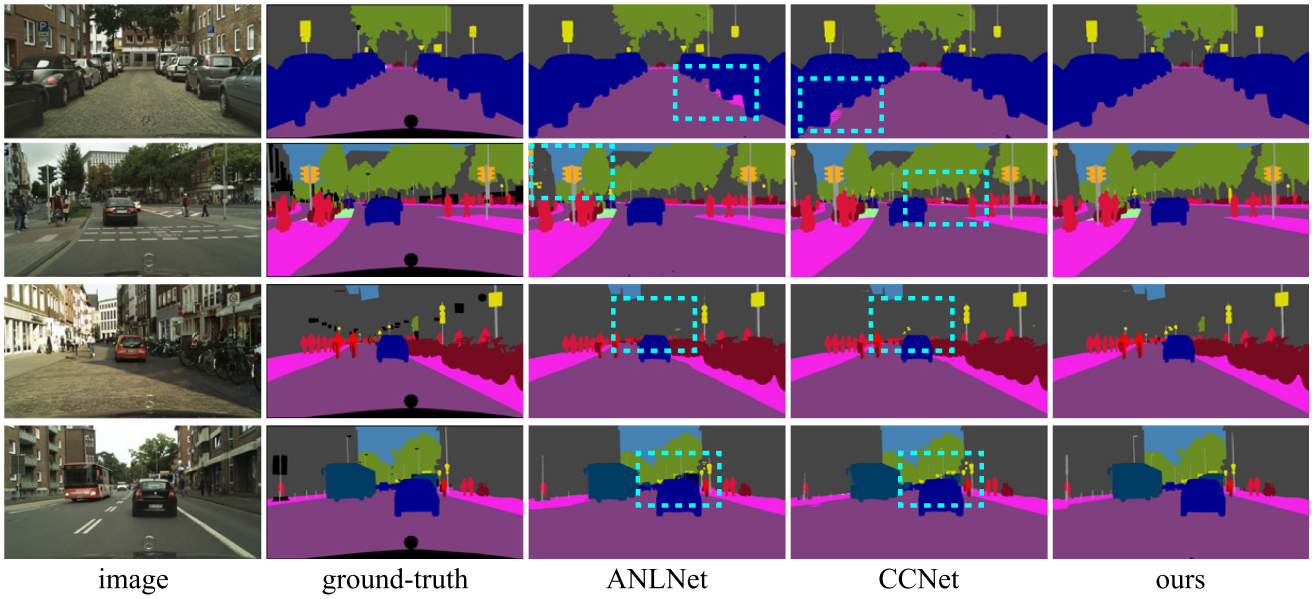


Fig. 12. Segmentation results on the Cityscapes validation set. We compare our approach with ANLNet [52] and CCNet [19].

In Table 6, we report the segmentation results, with different strategies of using the enriched features. Without using the enriched features to compute the information flow or ROI-upsample map, we degrade the network to the baseline encoder-decoder architecture with the adaption stage only, which achieves 80.4% mean IoU (see the first row). This result demonstrates that both the information flow map and ROI-upsample map are essential for segmentation.

Next, we disable the information flow map or the ROI-upsample map to analyze their contribution to the segmentation performance. When only information flow maps are used, each pixel on the intermediate feature map or the augmented feature map is enhanced by the information flow along a single sub-pathway. Compared to the full network, the performance drops over 1.7% mean IoU (see 82.6% mean IoU in the second row).

When only the ROI-upsample maps are used, the regional contents are propagated along different sub-pathways, injected into the pixels on the intermediate map and the augmented map. It helps to capture a more global relationship between the regions, leading to a higher accuracy than the network with the information flow maps alone. However, without the information flow map, the network loses the information enhanced by the complete set of regions along the sub-pathway. It leads to a drop of 0.8% mean IoU (see 83.5% mean IoU in the third row), compared to the full model (see the fourth row) that combines the information flow and ROI-upsample maps.

6.4 Comparisons With State-of-The-Art Methods

We extensively compare our TAGNet with the state-of-the-art methods on the Cityscapes validation set and test set.[5] For a fair comparison, we train our network without using extra coarse annotations provided in the Cityscapes training set. We make comparison with the SPP/ASPP, deformable and attention models [4], [10], [47], [48], [52], [53], where the quantitative results are listed in Table 7 and some representative visualized results are shown in Fig. 12. TAGNet outperforms the competitive methods.

TABLE 8
Comparisons With State-of-The-Art Methods on the PASCAL VOC 2012 Dataset, in Terms of Mean IoU (%)

validation set		test set	
method	**mIoU**	**method**	**mIoU**
Fu *et al.* [13]	80.4	Zhao *et al.* [48]	85.7
Lin *et al.* [31]	82.7	Zhong *et al.* [50]	86.1
Chen *et al.* [4]	82.7	Yu *et al.* [43]	86.2
Chen *et al.* [5]	84.6	Fu *et al.* [15]	86.6
Fu *et al.* [15]	84.8	Luo *et al.* [36]	86.8
Lin *et al.* [29]	85.1	Chen *et al.* [4]	86.9
Zhang *et al.* [46]	85.8	Li *et al.* [24]	87.7
ours	**86.8**	ours	**87.9**

TABLE 9
Comparisons With State-of-The-Art Methods on the COCO-Stuff Dataset, in Terms of Mean IoU (%)

10K validation set		164K validation set	
method	**mIoU**	**method**	**mIoU**
Dai *et al.* [7]	33.2	Zhao *et al.* [47]	31.2
Ji *et al.* [21]	35.0	Yu *et al.* [42]	31.3
Ding *et al.* [11]	35.7	Wang *et al.* [40]	35.8
Zhu *et al.* [52]	37.2	Zhu *et al.* [52]	38.1
Liang *et al.* [27]	38.9	Chen *et al.* [3]	39.1
Fu *et al.* [13]	39.7	Chen *et al.* [4]	40.0
Yuan *et al.* [44]	40.5	Fu *et al.* [14]	40.1
ours	**41.0**	ours	**41.5**

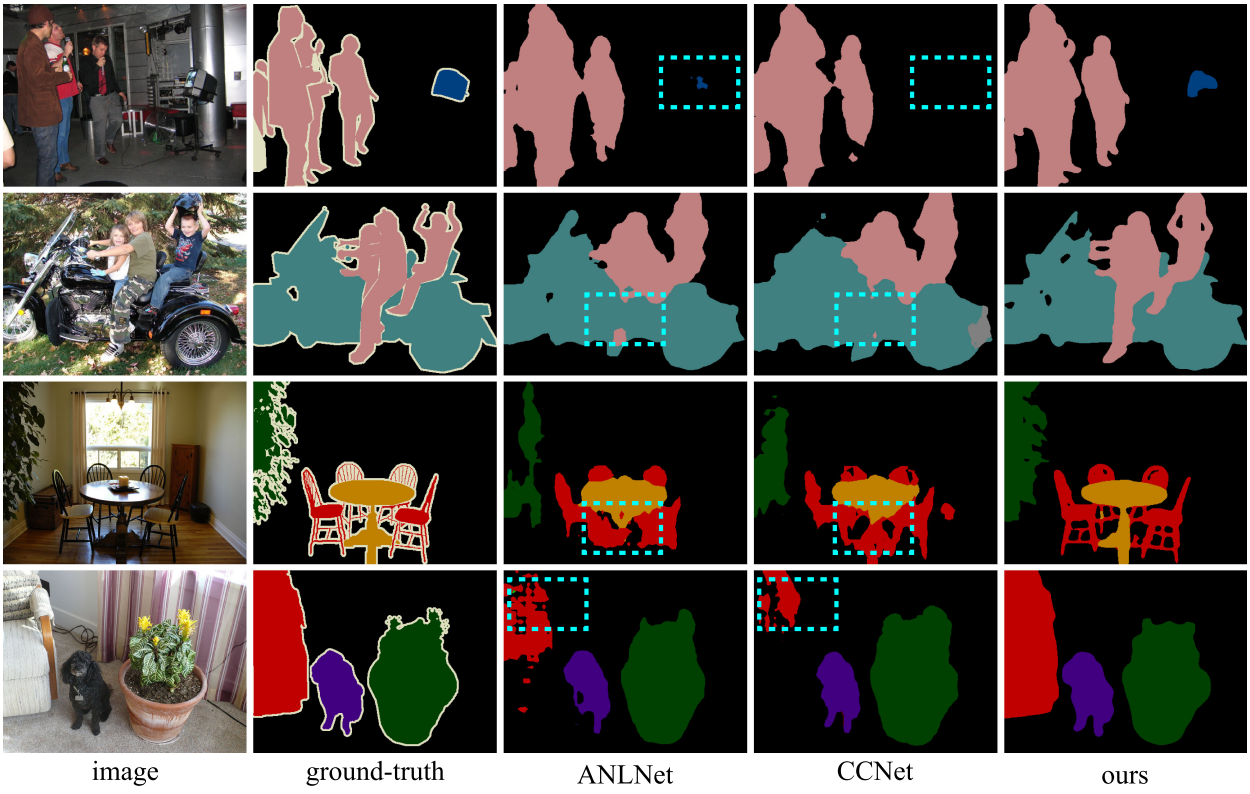


Fig. 13. Segmentation results on the PASCAL VOC 2012 validation set. We compare our approach with ANLNet [52] and CCNet [19].

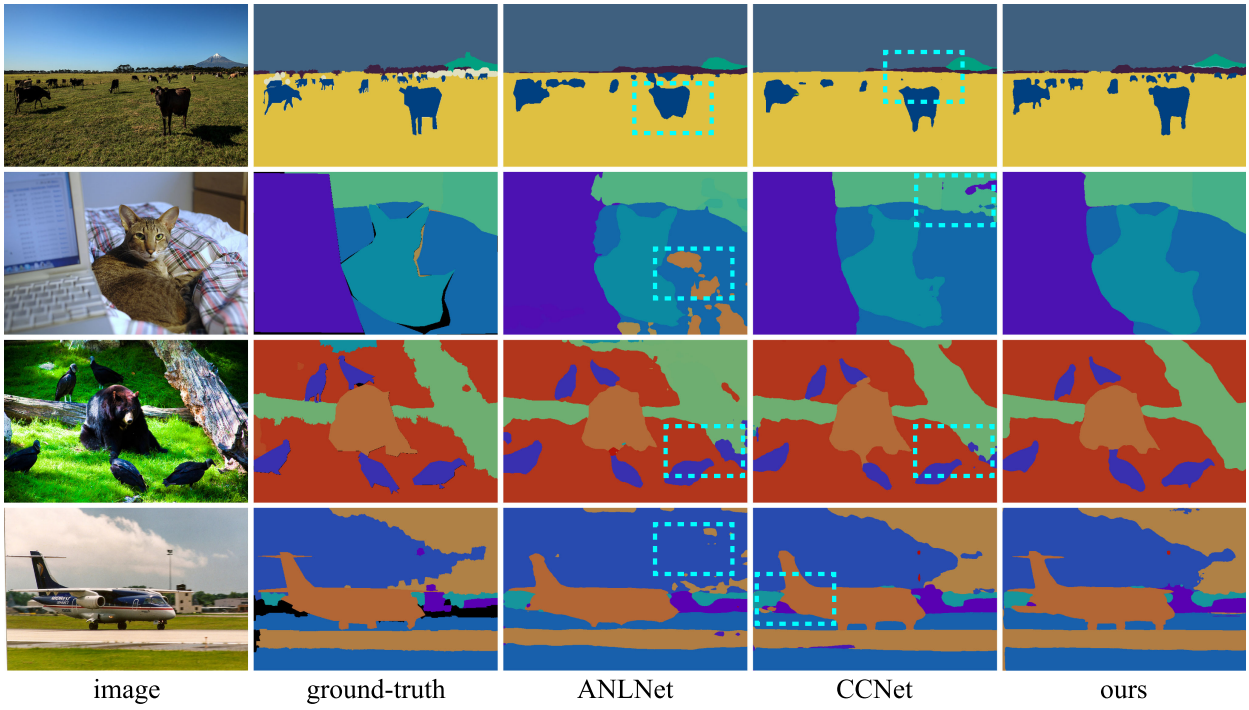


Fig. 14. Segmentation results on the COCO-Stuff-10 K validation set (the first two rows) and -164 K validation set (the last two rows). We compare our approach with ANLNet [52] and CCNet [19].

On PASCAL VOC 2012[6] [12] and COCO-Stuff-10 K/-164K [1], we also make comparison with the SPP/ASPP models [4], [46], deformable models [7], [21] and attention models [24], [50], where the quantitative results are listed in Tables 8 and 9. Compared to Cityscapes with outdoor urban images only, PASCAL VOC 2012 and COCO-Stuff are much challenging, since they contain images the indoor and outdoor images with more complex scenes, as shown in Figs. 13 and 14. Again, TAGNet achieve better performance

6. http://host.robots.ox.ac.uk:8080/anonymous/Y7FP7P.html

than other methods. It demonstrates the flexibility and robustness of CCP model and TAGNet for more challenging segmentation scenarios.

7 Conclusion

Recent advances in semantic image segmentation can be attributed largely to the exploitation of context information. In this paper, we have introduced TAGNet to establish CCP for exchanging context between pixels. CCPs are bidirectional and learnable, driven by the accuracy improvement of the segmentation task. CCP has configurable regions that are located with respect to the image content. These regions contain pixels to form context information, which is encoded into the information flowing along the CCP. By adaptively fusing multiple streams of context information at each pixel, we achieve state-of-the-art results on the public benchmarks. Our method can be also applied to improve performance in related recognition tasks, which will be investigated in our future work.

References

[1] H. Caesar, J. Uijlings, and V. Ferrari, "Coco-stuff: Thing and stuff classes in context," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1209–1218.

[2] L.-C. Chen *et al.*, "Searching for efficient multi-scale architectures for dense image prediction," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 8713–8724.

[3] 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.

[4] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv: 1706.05587*.

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

[6] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3213–3223.

[7] J. Dai *et al.*, "Deformable convolutional networks," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 764–773.

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

[9] L. Deng, M. Yang, H. Li, T. Li, B. Hu, and C. Wang, "Restricted deformable convolution based road scene semantic segmentation using surround view cameras," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4350–4362, Oct. 2020.

[10] H. Ding, X. Jiang, B. Shuai, A. Q. Liu, and G. Wang, "Semantic correlation promoted shape-variant context for segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8885–8894.

[11] H. Ding, X. Jiang, B. Shuai, A. Qun Liu, and G. Wang, "Context contrasted feature and gated multi-scale aggregation for scene segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2393–2402.

[12] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, pp. 303–338, 2010.

[13] J. Fu, J. Liu, H. Tian, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3146–3154.

[14] J. Fu *et al.*, "Adaptive context network for scene parsing," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 6748–6757.

[15] J. Fu, J. Liu, Y. Wang, J. Zhou, C. Wang, and H. Lu, "Stacked deconvolutional network for semantic segmentation," *IEEE Trans. Image Process.*, early access, Jan. 25, 2019, doi: 10.1109/TIP.2019.2895460.

[16] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel convolutional networks using bilateral inceptions," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 597–613.

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

[18] Q. Hou, L. Zhang, M.-M. Cheng, and J. Feng, "Strip pooling: Rethinking spatial pooling for scene parsing," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4003–4012.

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

[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[21] W. Ji *et al.*, "Semantic locality-aware deformable network for clothing segmentation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 764–770.

[22] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 784–799.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 84–90.

[24] X. Li, Z. Zhong, J. Wu, Y. Yang, Z. Lin, and H. Liu, "Expectation-maximization attention networks for semantic segmentation," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 9167–9176.

[25] Y. Li *et al.*, "Attention-guided unified network for panoptic segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7026–7035.

[26] X. Liang, X. Shen, J. Feng, L. Lin, and S. Yan, "Semantic object parsing with graph LSTM," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 125–143.

[27] X. Liang, H. Zhou, and E. Xing, "Dynamic-structured semantic propagation network," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 752–761.

[28] D. Lin, G. Chen, D. Cohen-Or, P.-A. Heng, and H. Huang, "Cascaded feature network for semantic segmentation of RGB-D images," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 1311–1319.

[29] D. Lin, Y. Ji, D. Lischinski, D. Cohen-Or, and H. Huang, "Multi-scale context intertwining for semantic segmentation", in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 603–619.

[30] D. Lin *et al.*, "ZigzagNet: Fusing top-down and bottom-up context for object segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7490–7499.

[31] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks with identity mappings for high-resolution semantic segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1925–1934.

[32] G. Lin, C. Shen, A. van den Hengel, and I. Reid, "Efficient piecewise training of deep structured models for semantic segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3194–3203.

[33] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[34] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, "Semantic image segmentation via deep parsing network," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1377–1385.

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

[36] P. Luo, G. Wang, L. Lin, and X. Wang, "Deep dual learning for semantic image segmentation," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2718–2726.

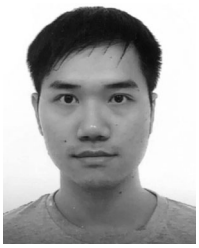
[37] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1520–1528.

[38] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, "Large kernel matters-improve semantic segmentation by global convolutional network," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4353–4361.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–14.

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

[41] T.-J. Yang *et al.*, "DeeperLab: Single-shot image parser," 2019, *arXiv:1902.05093*.
[42] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiseNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 325–341.
[43] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Learning a discriminative feature network for semantic segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1857–1866.
[44] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 173–190.
[45] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, "OCNet: Object context network for scene parsing," *Int. J. Comput. Vis.*, 2021.
[46] Z. Zhang, X. Zhang, C. Peng, X. Xue, and J. Sun, "ExFuse: Enhancing feature fusion for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 269–284.
[47] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2881–2890.
[48] H. Zhao *et al.*, "PsaNet: Point-wise spatial attention network for scene parsing," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 267–283.
[49] S. Zheng *et al.*, "Conditional random fields as recurrent neural networks," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1529–1537.
[50] Z. Zhong *et al.*, "Squeeze-and-attention networks for semantic segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13065–13074.
[51] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convNets V2: More deformable, better results," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9308–9316.
[52] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 593–602.
[53] Y. Zhuang *et al.*, "Dense relation network: Learning consistent and context-aware representation for semantic image segmentation," in *Proc. Int. Conf. Image Process.*, 2018, pp. 3698–3702.

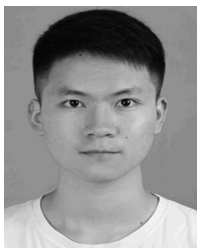


Di Lin (Member, IEEE) received the bachelor's degree in software engineering from Sun Yat-sen University in 2012, and the PhD degree from the Chinese University of Hong Kong in 2016. He is an associate Professor with the College of Intelligence and Computing, Tianjin University, China. His research interests are computer vision and machine learning.

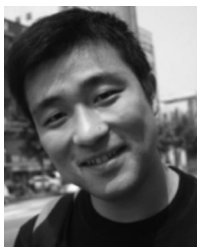
Dingguo Shen received the bachelor's degree in electronic commerce from the Guangdong University of Foreign Studies in 2018. He is currently working toward the MSc degree with Visual Computing Research Center, College of Computer Science and Software Engineering, Shenzhen University. His research interest is computer vision.

Yuanfeng Ji received the bachelor's degree in electronic information from Shenzhen University in 2018. He is currently a research assistant with Visual Computing Research Center, College of Computer Science and Software Engineering, Shenzhen University. His research interests is computer vision.

Siting Shen received the bachelor's degree from Guangzhou Medical University in 2018. She is currently working toward the MSc degree with Visual Computing Research Center, the College of Computer Science and Software Engineering, Shenzhen University. Her research interest is computer vision.



Mingrui Xie received the bachelor's degree from Shenzhen University in 2018. He is currently working toward the MSc degree with Visual Computing Research Center, the College of Computer Science and Software Engineering, Shenzhen University. His research interests is computer vision.



Wei Feng (Member, IEEE) received the BS and MPhil degrees in computer science from Northwestern Polytechnical University, China, in 2000 and 2003, respectively, and the PhD degree in computer science from the City University of Hong Kong in 2008. He is a full professor with the School of Computer Science and Technology, Tianjin University, China. His major research interests are active robotic vision and visual intelligence. He is the associate editor for *Neurocomputing and Journal of Ambient Intelligence and Humanized Computing*.

Hui Huang (Senior Member, IEEE) received the PhD degree in applied math from The University of British Columbia in 2008 and another PhD in computational math from Wuhan University in 2006. She is a distinguished professor with Shenzhen University, where she directs the Visual Computing Research Center at the College of Computer Science and Software Engineering. Her research interests span on computer graphics and computer vision. She is currently a senior member of ACM/CSIG, a distinguished member of CCF, an associate editor-in-chief of The Visual Computer and is on the editorial board of *ACM Transactions on Graphics and Computers & Graphics*.