Small-Object Sensitive Segmentation Using Across Feature Map Attention

Shengtian Sang, Yuyin Zhou, Md Tauhidul Islam, *Student Member, IEEE*, and Lei Xing

Abstract—Semantic segmentation is an important step in understanding the scene for many practical applications such as autonomous driving. Although Deep Convolutional Neural Networks-based methods have significantly improved segmentation accuracy, small/thin objects remain challenging to segment due to convolutional and pooling operations that result in information loss, especially for small objects. This article presents a novel attention-based method called Across Feature Map Attention (AFMA) to address this challenge. It quantifies the inner-relationship between small and large objects belonging to the same category by utilizing the different feature levels of the original image. The AFMA could compensate for the loss of high-level feature information of small objects and improve the small/thin object segmentation. Our method can be used as an efficient plug-in for a wide range of existing architectures and produces much more interpretable feature representation than former studies. Extensive experiments on eight widely used segmentation methods and other existing small-object segmentation models on CamVid and Cityscapes demonstrate that our method substantially and consistently improves the segmentation of small/thin objects.

Index Terms—Small-object semantic segmentation, across feature map attention

1 Introduction

Semantic segmentation is an important processing step in natural or medical image analysis for the detection of distinct types of objects in images [1]. In this process, a semantic label is assigned to each pixel of a given image. The breakthrough of semantic segmentation methods came when fully convolutional neural networks (FCN) were first used by [2] to perform end-to-end segmentation of images. While semantic segmentation has achieved significant improvement based on the conception of fully convolutional networks, small and thin items in the scene remain difficult to segment because the information of small objects is lost throughout the convolutional and pooling processes [3], [4], [5], [6]. For example, Fig. 1a is an image of size 800 by 1200 pixels, which contains two cars: the larger car is 160 by 220 pixels (Fig. 1b), and the smaller one is 30 by 40 (Fig. 1c). After a convolution operation with a convolution kernel of 10×10, the length and width of the image are compressed to one-tenth of the original size (as shown in Fig. 1d). Accordingly, the dimensions of the large and small cars become 16 by 22 and 3 by 4 pixels, respectively. As seen from the example, we can still see the car's features from Fig. 1e (feature map of the large car), but we can hardly see the features of the small car from the 12-pixel size Fig. 1c (feature map of the small car). This is because the high-level representation from convolutional and pooling operations generated along lowers the resolution, which often leads to the loss of the detailed information of small/thin objects [3] — as a result, recovering the car information from the coarse feature maps is difficult for segmentation models [7]. However, accurately segmenting small objects is critical in many applications, such as autonomous driving, where the segmentation and recognition of small-sized cars and pedestrians in the distance is critical [8], [9], [10], [11].

Some methods for small object segmentation have been proposed [6], [9], [10], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25]. A common strategy is to scale up the input images to improve the resolution of small objects or generate high-resolution feature maps [6], [9], [10], [12], [13]. The strategy relying on data augmentation or feature dimension increment generally results in significant time consumption for training and testing. Another promising strategy is to develop network variants, such as skip connections [14], hypercolumns [15], [16], feature pyramids [17], [18], [19], dilated convolution [20], [21], to enhance high-level small-scale features with multiple lower-level features layers. The strategy of integrating multi-scale representation cannot ensure the feature alignment of the same object and the features are not interpretable enough for semantic segmentation [10], [25]. Post-processing, such as Markov Random Field and Conditional Random Field-based post-processing [22], [23], is another strategy to improve the small object segmentation. Since post-processing is a separate part of the training of the segmentation model, the network cannot adapt its weights based on the post-processing outputs [24], [25].

In this paper, we present a novel small-object sensitive segmentation strategy without relying on increasing the data scale, enlarging the image/feature sizes, or modifying the

- *Shengtian Sang, Md Tauhidul Islam, and Lei Xing are with the Department of Radiation Oncology, Stanford University, Stanford, CA 94305 USA. E-mail: {sangst, tauhid, lei}@stanford.edu.*
- *Yuyin Zhou is with the Department of Computer Science and Engineering, University of California, Santa Cruz, CA 95064 USA. E-mail: zhouyuyiner@gmail.com.*

Manuscript received 27 January 2022; revised 10 July 2022; accepted 22 September 2022. Date of publication 30 September 2022; date of current version 3 April 2023.
This work was supported by NIH under Grants 1R01CA227713 and 1R01CA256890.
(Corresponding author: Lei Xing.)
Recommended for acceptance by B. Han.
This article has supplementary downloadable material available at https://doi.org/10.1109/TPAMI.2022.3211171, provided by the authors.
Digital Object Identifier no. 10.1109/TPAMI.2022.3211171

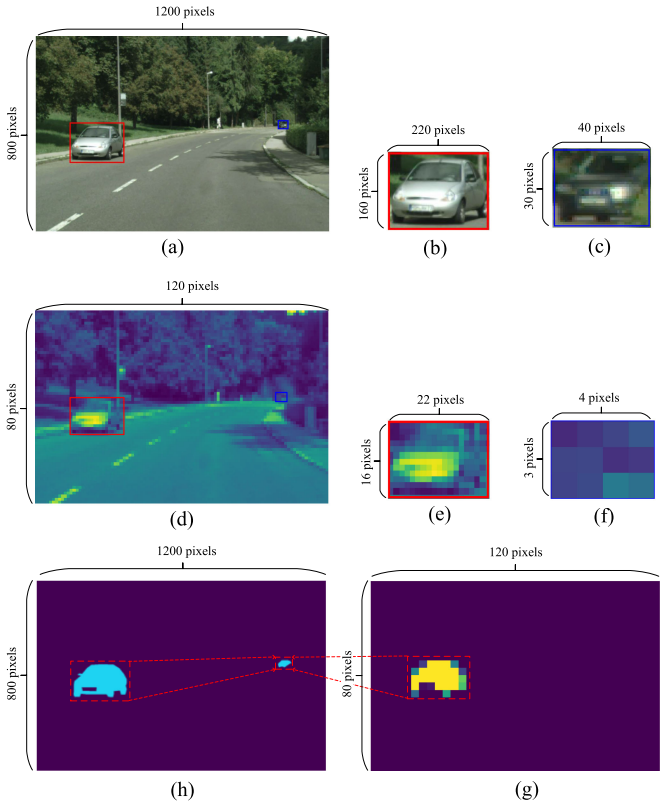


Fig. 1. Example of the convolution operation. The example employs convolution with a kernel of size 10 by 10 with parameters all set one, where stride is 10. The output of this convolution operation is one-hundredth of the original input pixels. (*a*) The original image of size 800×1200 pixels. (*b*) The larger car in the original image has a resolution of 160×220 pixels. (*c*) represents the smaller car which has a resolution of 30×40 pixels. (*d*) The original image's feature map that generated by the convolution operation. It is one-tenth the length and width of the original image, respectively. (*e*) The feature map of the large car. (*f*) The feature map of the small car. (*g*) The relationship between the small car (c) and the feature map (d). (*h*) Improving the performance by utilizing the obtained relations and the large car's output.

network architecture. Noticing that the same object category often shares similar imaging characteristics, we propose to leverage the relation among the small and large objects within the same category to compensate for the information loss from the feature propagation. For example, Figs. 1b and 1c show that the large and small cars are very similar despite their vastly different sizes, the output of the large car can be used to correct the results of the small car region if we know the different image regions represent the same type of object. However, directly calculating the degree of similarity from the input image can be quite challenging since the size of different objects can be vastly different. We hereby propose to quantify this relation by delving deeper into the feature space. This is motivated by the fact that the size of the small object in imaging space and the large object in feature space have more comparable size. For example, Figs. 1c and 1e show that the small car in the original image and the large car in the feature map have similar sizes and characteristics. We can derive the relation between the small and large cars by exploiting the original image of small car and the feature map of large car. To this end, we present Across Feature Map Attention (AFMA), which represents the similarities of objects in the same category by calculating the cross-correlation matrices between the intermediate feature patches and the image patches. The relation can then be utilized to enhance small object segmentation. For example, combining the relation of the small and large cars (Fig. 1g) and the output of the large car can compensate for the information loss of the small car (Fig. 1h). To further improve the quality of the obtained relation (i.e., AFMA), we propose to use the *gold* AFMA, which is computed based on the groundtruth segmentation mask, to constitute extra supervision.

As shown in Fig. 2, our method is an efficient plug-in which can be easily applied to a wide range of popular segmentation networks. To demonstrate its effectiveness, we comprehensively evaluate our method on eight segmentation models and two urban street scene datasets. The experimental results show that our method consistently improves the segmentation accuracy, especially for the small object classes. To conclude, the main contributions of this paper are:

- We propose a novel method, i.e., Across Feature Map Attention, to fully exploits the relation of objects in the same category, for enhancing small object segmentation.
- To the best of our knowledge, we present the first method to characterize attention by finding the relationship between different levels of feature maps.
- Unlike previous methods which applies data augmentation or multi-scale processing, our AFMA provides much more interpretable features (see Sections 4.4 and 4.5).
- The proposed AFMA is lightweight and can be easily plugged into a wide range of architectures. For instance, DeepLabV3, Unet, Unet++, MaNet, FPN, PAN, LinkNet, and PSPNet achieve 2.5%, 4.7%, 3.0%, 3.0%, 2.5%, 5.0%, 4.0%, and 2.9% improvement with only less than 0.1% parameters increment.

We release our codes as well as data processing procedures for the public datasets, so that other researchers can easily reproduce our results.[1]

2 Related Work

2.1 Semantic Segmentation

Most semantic segmentation approaches introduced in recent years have focused on increasing segmentation accuracy based on the FCN architecture [2], which focuses on drawing information from the input image and then using these derived features to construct the final segmentation image. *Encoder-decoder* is the widely adopted structure to improve FCN by considering spatial details and contexts. SegNet [26], a fully convolutional encoder-decoder architecture-based method, upsamples its lower-resolution input feature map(s) by using pooling indices computed in the max-pooling step. Furthermore, LinkNet [27], W-Net [28], HRNet [29], Stacked Deconvolutional Network [30], etc., adopt transposed convolutions or feature reuse strategies to overcome the shortcoming of fine-grained image information loss based on encoder-decoder architecture. Based on the encoder-decoder structure, Unet [14] integrates the feature maps of the encoder and decoder by dense *skip connections* to fully leverage the features from each layer. The

1. https://github.com/ShengtianSang/AFMA

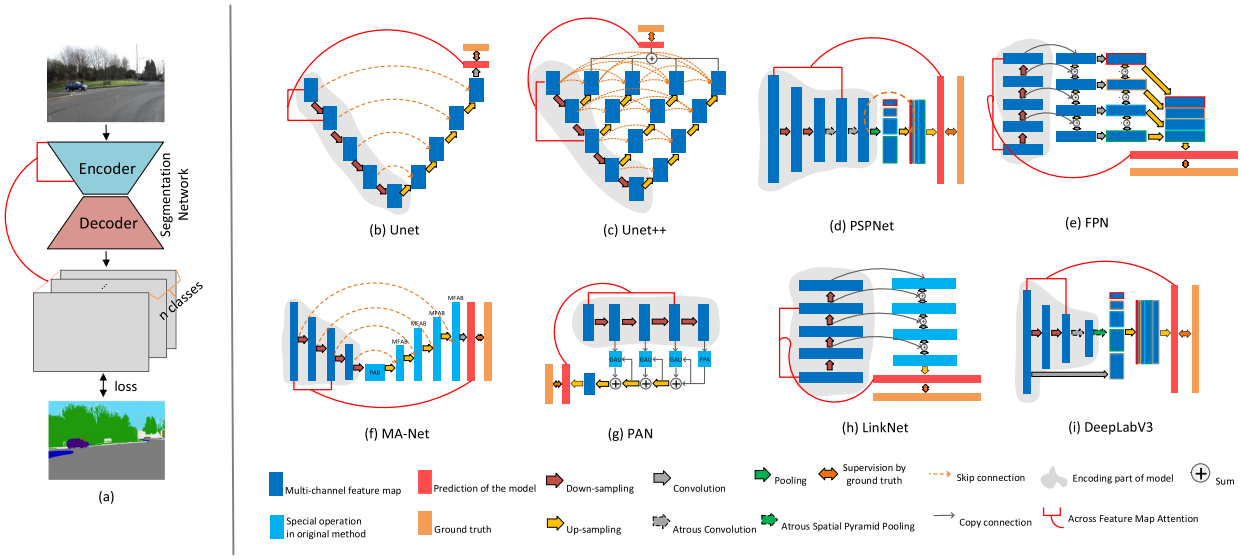


Fig. 2. The overview of our method. (*a*) represents an overview of combining the AFMA method with a general semantic segmentation method. The encoder of the segmentation model is input to the AFMA method, and its output is applied to the output of the segmentation method. (*b-i*) presents a detailed illustration of combining the AFMA method with different semantic segmentation models. It can be observed that the AFMA approach is adaptable to different types of architectures of various semantic segmentation models and can work on different layers of the encoder's feature maps.

features from each layer in the encoder part are connected to the symmetrical layers in the decoder part. Many extensions of UNet, such as Unet++ [31],mUnet [32], 3D-Unet [33] and stacked UNets [34], have been proposed for various problem areas. Subsequently, *pyramid pooling and dilated convolution* (also called atrous convolution) have been widely used to enhance the resolution of feature maps and enlarge the receptive field. Pyramid Scene Parsing Network (PSPNet) [18] adopts a multi-scale network for better learning the global context representation of a scene. DeepLab [21], DeepLabV3 [17], multiscale context aggregation [20], densely connected Atrous Spatial Pyramid Pooling [35], and Efficient Network [36] use large rate dilated/atrous convolutions to enlarge the receptive field and capture broader scope context information. *Attention mechanisms* have also been explored in semantic segmentation to assess the importance of features at different positions and scales [37]. Pyramid Attention Network (PAN) [38] combines attention mechanisms and spatial pyramids to extract specific dense features for semantic segmentation. Multi-scale Attention Net (MANet) [39] exploits self-attention to integrate local features with their global dependencies adaptively. Gated Fully Fusion [3] uses a gating mechanism (similar to attention mechanism) to fuse features from different feature maps selectively. The gating mechanism measures the usefulness of features and control information propagation through gates accordingly. In this paper, we select eight representative and widely used semantic segmentation models related to encoder-decoder, skip connection, pyramid pooling and dilated convolution, or attention mechanism as baseline models in our experiments.

2.2 Small Object Segmentation

The information of small and thin items will be lost as the network deepens due to the convolutional and pooling processes. Some specific methods for small object segmentation have been proposed [6], [9], [10], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25]. One common strategy is to scale up the input images to improve the resolution of small objects or generate high-resolution feature maps [6], [9], [10], [12], [13]. The strategy relying on data augmentation or feature dimension increment generally results in significant time consumption for training and testing. Another promising strategy is to develop network variants, such as skip connections [14], hypercolumns [15], [16], feature pyramids [17], [18], [19], dilated convolution [20], [21], to use multi-scale feature layers, which has the effect of zooming in small objects. Although the feature pyramid structure and dilated convolutions help overcome this issue, the small objects still cover too little practical information to be effectively recognized [4]. Post-processing, such as Markov Random Field and Conditional Random Field-based post-processing [22], [23], is another strategy to improve the small object segmentation. Since post-processing is a separate part of the training of the segmentation model and the network cannot adapt its weights based on the post-processing outputs [24], [25]. Changing the loss function is another way to improve small object segmentation. In [26], the loss function adopts the median frequency balancing weights for training, and it proposes to assign different training weights to objects of various sizes. Guo et al. [25] presented a small object boundary-sensitive loss function to improve small object recognition. The advantage of changing the loss function is that it does not introduce extra computational cost to the segmentation model. However, the improvement of small object segmentation is not interpretable enough for semantic segmentation. Here, we present a new method that could improve the small object segmentation with a very small extra computation cost and is also more convenient for interpretation.

2.3 Attention Networks

The attention mechanism is widely used in semantic segmentation to select significant features. Several methods adopt *global attention* to utilize the global scene context for

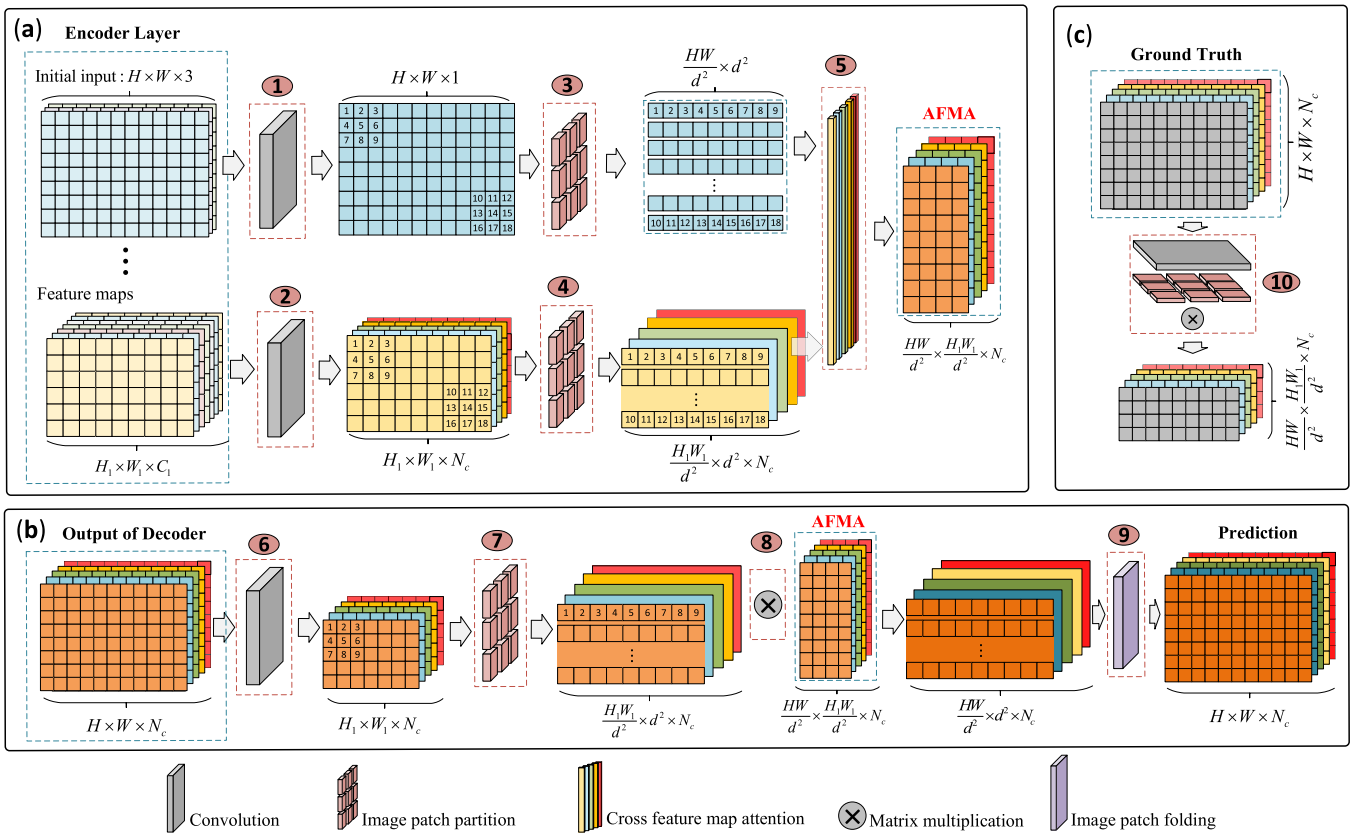


Fig. 3. The framework of our method. (*a*) Calculate the Across Feature Map Attention. The inputs are the initial image and i-th layer feature maps of the encoder. (*b*) Output Modification. The generated AFMA in (a) is used to modify the output of the decoder's predicted masks. (*c*) The process of generating gold AFMA.

segmentation. Pyramid Attention Network [38] presents a global attention upsampling method to extract specific dense features for semantic segmentation. Global Recurrent Localization Network [40] obtains attention maps from encoded features to attend to the global contexts. PiCA-Nets [41] presents global and local attention modules for capturing global and local settings in low- and high-resolution, respectively. Chen et al. [42] used hierarchical structures of attention maps to attend to global contexts at all scales. *Pixel attention* is used to capture the relationship between two pixels. The Multi-scale Attention Net [39] uses pixel attention to integrate local elements with their global dependencies. Wang et al. [42] designed a non-local operation that computes interactions between two locations to capture long-range relationships directly. Squeeze-and-excitation first uses global average pooling and then passes through multilayer perceptrons to obtain *Channel attention* to improve models' performance. DFN [43] uses global average pooling to bring channel-wise attention into the network while selecting the selection of more discriminative features. Attention Complementary Network (ACNet) [44] is a channel attention-based module that extracts weighted features from initial image and depth branches. Zhang et al. [45] progressively utilized both spatial and channel-wise attention to integrate multiple contextual information of multi-level features. In this paper, we propose *across feature maps attention* to find the relation between small and large objects, for enhancing small object segmentation.

3 Method

3.1 Overview

As shown in figure Fig. 2a, the pipeline consists of a default segmentation network and our proposed method. The default segmentation network can be any segmentation model, and our approach takes the model's encoder as input, and its output is added to the output of the decoder.

Specifically, the general semantic segmentation architecture consists of an encoder and a decoder. The encoding part is used for feature representation learning, while the decoder is for pixel-level classification. Our method first computes the Across Feature Map Attention (AFMA), which aims to quantify the relation between each input image and the corresponding feature maps, i.e., the output from the intermediate encoding layers (see Section 3.2). Then the derived AFMA is used to modulate the output from the decoder (see Section 3.3). In addition, we propose to compute the *gold* AFMA (see Section 3.4) based on the groundtruth segmentation mask for better guiding the learning of AFMA. During the training, the overall objective is comprised of both the standard segmentation loss and an additional AFMA loss in order to let the learned AFMA approach the gold AFMA (see Section 3.5). The overall framework of our method is illustrated in Fig. 3.

3.2 Across Feature Map Attention (AFMA)

The proposed AFMA aims to model the relationship between small and large objects. Specifically, AFMA computes the

cross-correlation between the patches of original image and the patches of corresponding feature maps. By compensating for the information loss from the feature propagation, AFMA effectively boosts the segmentation performance, especially from those small objects. Given a color image $\boldsymbol{I} \in \mathbb{R}^{W\times H\times 3}$ as input, the encoder of the segmentation model extracts the feature maps $\{F_i|i=1,\ldots,L\}$, where $\boldsymbol{F}_i \in \mathbb{R}^{H_i\times W_i\times C_i}$ and W_i, H_i, C_i denote width, height, and the number of channels of the feature maps, respectively. Our model takes the first level feature map of encoder F_1 (the original input image) and i-th layer feature map $F_i(2\leqslant i\leqslant L)$ as input to generate the AFMA, we introduce the detailed process as following the serial numbers shown in Fig. 3a.

Step ①. One 7×7 convolutional layer $Conv_1^1$ with 64 filters and one 3×3 convolutional layer $Conv_1^2$ with 1 filter are used to embed the first level feature maps (the image $\boldsymbol{I}$). In particular, we formulate this procedure as Eq. (1):

$$\boldsymbol{R}_{img} = Conv_1^2(Conv_1^1(\boldsymbol{I})), \tag{1}$$

where $\boldsymbol{R}_{img} \in \mathbb{R}^{H\times W\times 1}$.

Step ②. Given the i-th level feature maps $\boldsymbol{F}_i \in \mathbb{R}^{H_i\times W_i\times C_i}$, one 1×1 convolutional layer $Conv_2$ with N_c filters convert the i-th level feature maps to per category features. Eq. (2) describes the procedure

$$\boldsymbol{R}_{ith} = Conv_2(\boldsymbol{F}_i), \tag{2}$$

where $\boldsymbol{R}_{ith} \in \mathbb{R}^{H_i\times W_i\times N_c}$, N_c is the number of categories to predict. We hypothesize $\boldsymbol{R}_{ith}^k \in \mathbb{R}^{H_i\times W_i\times 1}$ $(1\leqslant k\leqslant N_c)$ only contains the information associated with the k-th category.

Step ③. In this step, we split $\boldsymbol{R}_{img}$ into fixed-size patches. The procedure is described as Eq. (3)

$$\boldsymbol{P}_{img} = \boldsymbol{\phi}(\boldsymbol{R}_{img}, d). \tag{3}$$

The procedure follows [46], $\boldsymbol{\phi}$ is the image partition operation which reshapes the $\boldsymbol{R}_{img} \in \mathbb{R}^{H\times W\times 1}$ into a sequence of flattened 2D patches $\boldsymbol{P}_{img} \in \mathbb{R}^{\frac{HW}{d^2}\times d^2}$. The j-th $(1\leqslant j\leqslant \frac{HW}{d^2})$ vector of $\boldsymbol{P}_{img}$ contains a $d\times d$ resolution patch of the image[2].

Step ④. Similar to step ③, we partition each channel of $\boldsymbol{R}_{ith}$ into a sequence of flattened patches, respectively.

$$\boldsymbol{P}_{ith}^k = \boldsymbol{\phi}(\boldsymbol{R}_{ith}^k, d), \tag{4}$$

$$\boldsymbol{P}_{ith} = \boldsymbol{P}_{ith}^1 \| \boldsymbol{P}_{ith}^2 \cdots \| \boldsymbol{P}_{ith}^{N_c}, \tag{5}$$

where $\boldsymbol{P}_{ith}^k \in \mathbb{R}^{\frac{H_iW_i}{d^2}\times d^2}$ $(1\leqslant k\leqslant N_c)$ is the sequence of flattened 2D patches of $\boldsymbol{R}_{ith}^k$, $\|$ represents concatenation and $\boldsymbol{P}_{ith} \in \mathbb{R}^{\frac{H_iW_i}{d^2}\times d^2\times N_c}$.

Step ⑤. The dot product is adopted between $\boldsymbol{P}_{img}$ and each $\boldsymbol{P}_{ith}^k$ of $\boldsymbol{P}_{ith}$ to determine the relationship between each image patch of the original image and the k-th category-related feature map. Eqs. (6) and (7) describe the procedure

$$\boldsymbol{A}_{ith}^k = \boldsymbol{P}_{img} \otimes (\boldsymbol{P}_{ith}^k)^{-1}, \tag{6}$$

$$\boldsymbol{A}_{ith} = \boldsymbol{A}_{ith}^1 \| \boldsymbol{A}_{ith}^2 \cdots \| \boldsymbol{A}_{ith}^{N_c}, \tag{7}$$

where $\boldsymbol{A}_{ith} \in \mathbb{R}^{\frac{HW}{d^2}\times\frac{H_iW_i}{d^2}\times N_c}$. $(\boldsymbol{P}_{ith}^k)^{-1}$ is the transposed matrix of $\boldsymbol{P}_{ith}^k$, and $\boldsymbol{A}_{ith}^k \in \mathbb{R}^{\frac{HW}{d^2}\times\frac{H_iW_i}{d^2}}$ represents the associations between each image patch of the original image and each image patch of the k-th category-related feature map.

3.3 Probability/Output Modulation

The obtained attention map $\boldsymbol{A}_{ith}$ is then applied to modulate the output of the decoder, so as to enhance the segmentation from those small objects (Fig. 3b). The general segmentation network outputs a predicted masks $\boldsymbol{M}_{mask} \in \mathbb{R}^{W\times H\times N_c}$ for the input image $\boldsymbol{I} \in \mathbb{R}^{W\times H\times 3}$, where

$$\{\boldsymbol{M}_{mask}^k(j)\}_{j=1}^{WH} = \{\hat{m}_{mask}^k(j), \hat{p}_{mask}^k(j)\}_{j=1}^{WH}. \tag{8}$$

The N_c predicted masks $\boldsymbol{M}_{mask}^k \in [0,1]^{H\times W}$ $(1\leqslant k\leqslant N_c)$ are softly exclusive to each other, *i.e.*, $\sum_{k=1}^{N_c} \boldsymbol{M}_{mask}^k = \boldsymbol{1}^{H\times W}$, and each pixel's predicted mask $\hat{m}_{mask}^k(j)$ denotes the probability $\hat{p}_{mask}^k(j)$ of assigning class k to the j-th pixel.

Step ⑥. We use fixed-sized average pooling to compress the output of decoder $\boldsymbol{M}_{mask} \in \mathbb{R}^{H\times W\times N_c}$ to the same size as $\boldsymbol{R}_{ith} \in \mathbb{R}^{H_i\times W_i\times N_c}$. The procedure is described as Eqs. (9) and (10)

$$\boldsymbol{R}_{mask}^k = \boldsymbol{\psi}\left(\boldsymbol{M}_{mask}^k, \frac{H}{H_i}, \frac{W}{W_i}, \frac{H}{H_i}, \frac{W}{W_i}\right), \tag{9}$$

$$\boldsymbol{R}_{mask} = \boldsymbol{R}_{mask}^1 \| \boldsymbol{R}_{mask}^2 \cdots \| \boldsymbol{R}_{mask}^{N_c}, \tag{10}$$

where $\boldsymbol{R}_{mask}^k \in \mathbb{R}^{H_i\times W_i\times 1}$ and $\boldsymbol{R}_{mask} \in \mathbb{R}^{H_i\times W_i\times N_c}$. $\boldsymbol{\psi}(input, k_H, k_W, s_H, s_W)$ denote average pooling with kernel size (k_H, k_W) and stride (s_H, s_W). For each pixel $\boldsymbol{R}_{mask}^k(p_1, p_2)$ $(1\leqslant p_1\leqslant H_i, 1\leqslant p_2\leqslant W_i)$ from $\boldsymbol{R}_{mask}^k$ is:

$$\boldsymbol{R}_{mask}^k(p_1,p_2) = \frac{1}{k_H\times k_W}\sum_{m=1}^{k_H}\sum_{n=1}^{k_W} \boldsymbol{M}_{mask}^k(s_H\times p_1 + m, s_W\times p_2 + n). \tag{11}$$

We can know from Eq. (11) that similarly to $\boldsymbol{M}_{mask} \in [0,1]^{H\times W\times N_c}$, the N_c compressed masks $\boldsymbol{R}_{mask}^k \in [0,1]^{H_i\times W_i}$ are softly exclusive to each other, *i.e.*, $\sum_{k=1}^{N_c} \boldsymbol{R}_{mask}^k = 1^{H_i\times W_i}$. The value of each pixel $\boldsymbol{R}_{mask}^k(p_1,p_2)$ represents the probability of assigning class k to the compressed pixel.

Step ⑦. Same as steps ③ and ④, we partition each channel of $\boldsymbol{R}_{mask}$ into a sequence of flattened patches.

$$\boldsymbol{P}_{mask}^k = \boldsymbol{\phi}(\boldsymbol{R}_{mask}^k, d), \tag{12}$$

$$\boldsymbol{P}_{mask} = \boldsymbol{P}_{mask}^1 \| \boldsymbol{P}_{mask}^2 \cdots \| \boldsymbol{P}_{mask}^{N_c}, \tag{13}$$

where $\boldsymbol{P}_{mask}^k \in \mathbb{R}^{\frac{H_iW_i}{d^2}\times d^2}$ and $\boldsymbol{P}_{mask} \in \mathbb{R}^{\frac{H_iW_i}{d^2}\times d^2\times N_c}$.

Step ⑧. In this step, the attention map $\boldsymbol{A}_{ith}$ is applied on top of $\boldsymbol{P}_{mask}$ to spatially modulate the output probability.

$$\hat{\boldsymbol{M}}_{ith}^k = \boldsymbol{A}_{ith}^k \otimes \boldsymbol{P}_{mask}^k, \tag{14}$$

$$\hat{\boldsymbol{M}}_{ith} = \hat{\boldsymbol{M}}_{ith}^1 \| \hat{\boldsymbol{M}}_{ith}^2 \cdots \| \hat{\boldsymbol{M}}_{ith}^{N_c}, \tag{15}$$

where $\hat{\boldsymbol{M}}_{ith}^k \in \mathbb{R}^{\frac{HW}{d^2}\times d^2}$ and $\hat{\boldsymbol{M}}_{ith} \in \mathbb{R}^{\frac{HW}{d^2}\times d^2\times N_c}$. Since $\boldsymbol{P}_{mask}$ contains the predicted masks for compressed image patch, and $\boldsymbol{A}_{ith}$ represents the relation between the initial image patches (containing relative small objects) and the feature patches (containing relative large objects). $\hat{\boldsymbol{M}}_{ith}^k$ represents the influence of the output of large objects on small objects.

2. To make the patch size (d,d) divisible by the feature map size of (H_i, W_i), bottom-right padding is employed on the feature map if needed.

Step ⑨. We fold the $\hat{\boldsymbol{M}}_{ith} \in \mathbb{R}^{\frac{HW}{d^2} \times d^2 \times N_c}$ to the same size of $\boldsymbol{M}_{mask} \in \mathbb{R}^{H \times W \times N_c}$:

$$\boldsymbol{O}_{ith}^{k} = \boldsymbol{\phi}^{-1}(\hat{\boldsymbol{M}}_{ith}^{k}, d), \tag{16}$$

$$\boldsymbol{O}_{ith} = \boldsymbol{O}_{ith}^{1} \| \boldsymbol{O}_{ith}^{2} \cdots \| \boldsymbol{O}_{ith}^{N_c}, \tag{17}$$

where $\boldsymbol{O}_{ith}^{k} \in \mathbb{R}^{H \times W \times 1}$ and $\boldsymbol{O}_{ith} \in \mathbb{R}^{H \times W \times N_c}$. $\boldsymbol{\phi}^{-1}(\hat{\boldsymbol{M}}_{ith}^{k}, d)$ denotes the operation of folding the input by $d \times d$. $\boldsymbol{O}_{ith}$ represents the modification generated from the i-th feature maps, and the final prediction $\boldsymbol{Pre}$ of our method is:

$$\boldsymbol{Pre} = \boldsymbol{M}_{mask} + \boldsymbol{O}_{ith}, \tag{18}$$

where $\boldsymbol{Pre} \in \mathbb{R}^{H \times W \times N_c}$.

3.4 Gold AFMA Computation

We define the *gold* AFMA $\boldsymbol{A}_{gt} \in \mathbb{R}^{\frac{HW}{d^2} \times \frac{H_i W_i}{d^2} \times N_c}$, which is computed based on the groundtruth segmentation mask. During the training process, the gold AFMA will be used to provide supervision for $\boldsymbol{A}$, for further improving the quality of the attention map.

Step ⑩. Fig. 3c illustrates the process of calculating the gold AFMA. For the groundtruth labels $\boldsymbol{M}_{gt} \in \mathbb{R}^{H \times W \times N_c}$, the N_c groundtruth masks $\boldsymbol{M}_{gt}^{k} \in \{0,1\}^{H \times W}$ do not overlap with each other, *i.e.*, $\sum_{k=1}^{N_c} \boldsymbol{M}_{gt}^{k} = \boldsymbol{1}^{H \times W}$

$$\boldsymbol{R}_{gt}^{k} = \boldsymbol{\psi}\left(\boldsymbol{M}_{gt}^{k}, \frac{H}{H_i}, \frac{W}{W_i}, \frac{H}{H_i}, \frac{W}{W_i}\right), \tag{19}$$

$$\boldsymbol{A}_{gt}^{k} = \boldsymbol{\phi}(\boldsymbol{M}_{gt}^{k}, d) \otimes \boldsymbol{\phi}(\boldsymbol{R}_{gt}^{k}, d)^{-1}, \tag{20}$$

$$\boldsymbol{A}_{gt} = \boldsymbol{A}_{gt}^{1} \| \boldsymbol{A}_{gt}^{2} \cdots \| \boldsymbol{A}_{gt}^{N_c}, \tag{21}$$

where $\boldsymbol{A}_{gt} \in \mathbb{R}^{\frac{HW}{d^2} \times \frac{H_i W_i}{d^2} \times N_c}$. Each value of $\boldsymbol{A}_{gt}$ represents the gold relationship between original image's patch and feature maps' patch.

Fig. 4 is an illustration of the gold AFMA calculation. Assume we have an initial input image of 8×12 which contains two types of objects: square and cross. The groundtruth labels are two 8×12 matrices denoted as $\boldsymbol{M}_{gt}^{square}$ and $\boldsymbol{M}_{gt}^{cross}$, respectively. According to Eq. (19), we use average pooling operation $\boldsymbol{\psi}(\boldsymbol{M}_{gt}, 2, 2, 2, 2)$ to compress the groundtruth into $\boldsymbol{R}_{gt}^{square} \in \mathbb{R}^{4 \times 6}$ and $\boldsymbol{R}_{gt}^{cross} \in \mathbb{R}^{4 \times 6}$, respectively. Then following Eq. (20), image partition operation $\boldsymbol{\phi}(\boldsymbol{R}_{gt}^{square}, 2)$ and $\boldsymbol{\phi}(\boldsymbol{R}_{gt}^{cross}, 2)$ are first acted on the compressed groundtruth, which obtain two flatten patched image matrices $\boldsymbol{P}_{gt}^{square} \in \mathbb{R}^{6 \times 4}$ and $\boldsymbol{P}_{gt}^{cross} \in \mathbb{R}^{6 \times 4}$. Similarly, we perform image partition operations $\boldsymbol{\phi}(\boldsymbol{M}_{gt}^{square}, 2)$ and $\boldsymbol{\phi}(\boldsymbol{M}_{gt}^{cross}, 2)$ to the original image, and the partition image patches $\boldsymbol{P}_{GT}^{square} \in \mathbb{R}^{24 \times 4}$ and $\boldsymbol{P}_{GT}^{square} \in \mathbb{R}^{24 \times 4}$ are obtained. Following Eq. (20), the $\boldsymbol{A}_{gt}^{square} \in \mathbb{R}^{24 \times 6}$ and $\boldsymbol{A}_{gt}^{cross} \in \mathbb{R}^{24 \times 6}$ are generated by matrix multiplication operation. Fig. 4 shows four examples of calculating the values of $\boldsymbol{A}_{gt}^{square}$ and $\boldsymbol{A}_{gt}^{cross}$:

Square Related AFMA. As shown in Fig. 4, there is no square-related pixel (the pixels that make up squares) in image patch ① (yellow dashed line). Thus we can observe that the $\boldsymbol{A}^{①} \in \mathbb{R}^{1 \times 6}$ are all zeros, which means there is no relationship between image patch ① and all other image patches of $\boldsymbol{M}_{gt}^{square}$. For image patch ② (red dashed line), all pixels in this image patch belong to square. As a result, we can see from Fig. 4 that the $\boldsymbol{A}^{②} \in \mathbb{R}^{1 \times 6}$ is strongly related to the image patch of the compressed large square (the value is 1) and also has a relatively strong relationship with the image patch of the compressed smaller square (the value is 0.25).

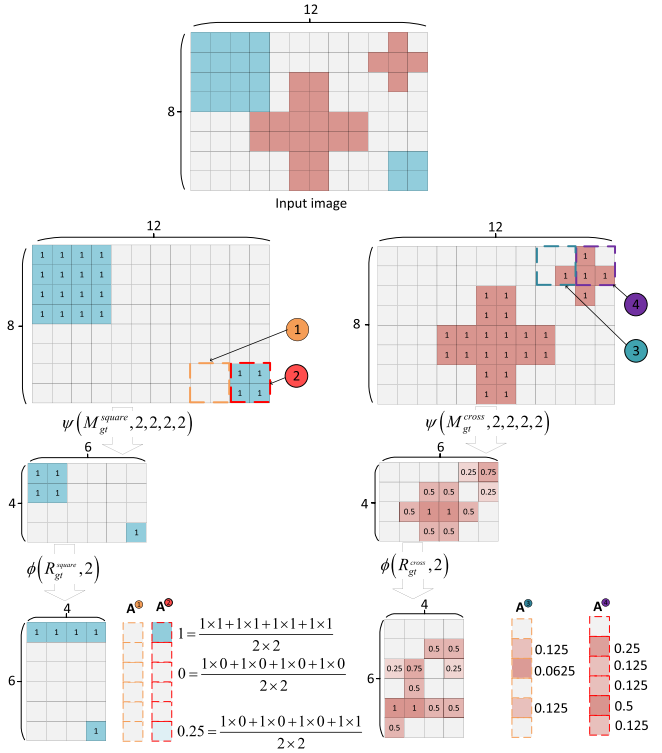


Fig. 4. An illustration of calculating gold AFMA. The 0 in the gold standard masks is not presented.

Cross Related AFMA. The Fig. 4 shows that there are 1 and 3 cross-related pixels in image patch ③ (blue dashed line) and ④ (purple dashed line), respectively. As a result, image patch ④ has stronger relations with each compressed large cross than image patch ③ ($\boldsymbol{A}^{④}$'s value is greater than $\boldsymbol{A}^{③}$ at each position). From the above examples, we know that the more the image patch of the original image contains pixels of a specific object, the more it is related to the image patch of feature maps of that object.

The $\boldsymbol{A}^{①}$, $\boldsymbol{A}^{②}$, $\boldsymbol{A}^{③}$ and $\boldsymbol{A}^{④}$ are used on the output of the general segmentation model's decoder (as shown in Section 3.3). Since the values of $\boldsymbol{A}^{①}$ are zeros, the predicted masks of the large square have no impact on the results of image patch ① but have a significant effect on the results of image patch ② due to ② has a strong relationship with the image patch of the compressed large square. Similarly, the prediction of the large cross will have more influence on the output of image patch ④ than ③ since image patch ④ of the original image contains more small cross-related pixels.

3.5 Training Losses

The overall training objective consists of two loss terms: 1) the standard segmentation loss, which aims to minimize the difference between the prediction and the groundtruth segmentation mask; 2) the AFMA loss, which aims to minimize the difference between the learned AFMA and the gold AFMA. For the segmentation loss, we follow [26] and adopt the median frequency balancing weighted sigmoid cross-entropy loss for training:

$$\mathcal{L}_{seg} = \frac{-1}{N_c \cdot H \cdot W} \sum_{k=1}^{N_c} \sum_{h=1}^{H} \sum_{w=1}^{W} \boldsymbol{\eta}_k \big[s^{gt}(k,h,w) \log s^{pred}(k,h,w) + (1 - s^{gt}(k,h,w)) \log (1 - s^{pred}(k,h,w)) \big], \tag{22}$$

where $s^{gt}(k,h,w)$ denotes the groundtruth (0 or 1) of class k at pixel (h,w), and $s^{pred}(k,h,w)$ denotes the the k-th probability of the final prediction $\boldsymbol{Pre}$ at (h,w). $\boldsymbol{\eta}_k$ denotes the median frequency weight assigned to category k. For the AFMA, we use mean square error (MSE) loss for training:

$$\mathcal{L}_{afma} = \frac{1}{N_c \cdot L_1 \cdot L_2} \sum_{k=1}^{N_c} \sum_{l_1=1}^{L_1} \sum_{l_2=1}^{L_2} \Big[\boldsymbol{A}_{ith}^{k}(l_1,l_2) - \boldsymbol{A}_{gt}^{k}(l_1,l_2) \Big]^2, \tag{23}$$

where $\boldsymbol{A}_{ith}$ (as shown in Eq. (7)) and $\boldsymbol{A}_{gt}$ (as shown in Eq. (21)) are the predicted and the gold AFMA, respectively. Then the overall training objective consists of the two losses:

$$\mathcal{L} = \mathcal{L}_{seg} + \mathcal{L}_{afma}. \tag{24}$$

4 Experiments

4.1 Datasets and Evaluation Measures

We evaluate our proposed method on two well known urban street datasets: CamVid [47] and Cityscape [48]. The CamVid dataset is a road scene segmentation dataset of practical interest for various autonomous driving-related problems. The dataset consists of 367 training images, 100 validation images, and 233 testing images. In total, 11 semantic classes are annotated by pixel-level. The resolution of the images is 720×960. We define *sign symbol*, *pedestrian*, *pole*, and *bicyclist* as small-object classes based on the item size [10] as other's defined in [25]. The remaining seven object classes are all denoted as large-object classes. The Cityscapes dataset is a recently released dataset for semantic urban street scene understanding. The dataset includes 5,000 precisely annotated pixel-level images: 2,975 training images, 500 validation images, and 1,525 testing images. The resolution of the image is 1,024×2,048. In addition, Cityscapes provides 20,000 coarsely annotated images. In this paper, we consider only the fine annotations for training. There are a total of 19 semantic classifications in Cityscapes. We define *pole*, *traffic light*, *traffic sign*, *person*, *rider*, *motorcycle*, and *bicycle* as small-object classes based on the object size [10]. All the other 12 object classes are designated as large-object classes. CamVid dataset provides the ground-truth segmentation labels for the training, validation, and testing. Cityscapes only provides the ground-truth labels for the training and validation datasets, the ground-truth for the testing dataset is withheld from the user. The testing results of the Cityscapes dataset are obtained via online submissions.

The metrics used for segmentation performance evaluation in this paper are: Class intersection over union (IoU), mean intersection over union (mIoU), mean small-object class intersection over union (mIoU_S), and mean large-object class intersection over union (mIoU_L).

4.2 Implementation Details

We directly trained all models from scratch on CamVid and Cityscapes without pre-training the backbone on the ImageNet [49]. We have not used any coarse labeled images or any extra data in this work. In all experiments, we conduct our models by PyTorch,[3] and all baseline segmentation models used in the experiments are implemented using SMP.[4]

We use mini-batch stochastic gradient descent (SGD) with momentum 0.9, weight decay of $5e^{-4}$ and adaptive learning rates. The batch size is set to 16, 8 for the CamVid and Cityscapes dataset, respectively. Data augmentation contains random horizontal flip with probability 0.5, random crop, and random resize with scale range [0.5, 2.0]. The cropped resolution is 480× 640 for training CamVid and is 640× 800 for Cityscapes. The whole training process terminates in 500 epochs for CamVid and 200 epochs for Cityscapes. The initial learning rate is set to $10e^{-3}$ and it will be divided by 10 after 200, 300, and 400 epochs for CamVid and 100, 150 epochs for Cityscapes.

We perform all experiments under CUDA 11.1 and on a computer equipped with Intel(R) Xeon(R) Platinum 8180 (28 cores, 3.4 GHz) CPU, 4 NVIDIA GTX 8000 GPUs, and 256G RAM.

4.3 Comparison to Baselines and Existing Methods

To demonstrate the effectiveness of the proposed method, we evaluate the proposed method using eight widely used segmentation network and other existing small-object segmentation methods. The eight networks could be categorized as four types (Section 2.1): 1) Fully convolutional encoder-decoder architecture-based models including *LinkNet*; 2) Methods fusing low-level and high-level features by skip connections including *Unet* and *Unet++*; 3) Pyramid pooling and dilated convolution based models including *PSPNet*, *DeepLabV3* and *FPN*; 4) Attention mechanism adopted methods including *PAN* and *MANet*. Our proposed method is also compared with other existing small-object segmentation method: ISBEncoder [25], SegNet [26], ALE [50], DLA [51] SuperParsing [52], Liu&He [53], Deeplab-LFOV [21], and FoveaNet [54].

Table 1 shows the quantitative results of using the CamVid dataset for evaluation. Compared to the general models without the AFMA, the IoU scores of the small objects would be significantly improved by applying the AFMA module to the baseline segmentation network. When the AFMA is combined to the DeepLabV3, Unet, Unet++, MaNet, FPN, PAN, LinkNet, and PSPNet baseline segmentation networks, it demonstrates 2.5%, 4.7%, 3.0%, 3.0%, 2.5%, 5.0%, 4.0%, and 2.9% improvements for small-object classes (mIoU_S), respectively. The AFMA enhances baseline model performance on all small objects (sign symbol, pedestrian, pole, bicyclist) and improves significantly for specific types of small objects. For example, AFMA enhances the performance of PAN and LinkNet on recognizing pedestrians by nearly 10% (9.2% and 9.3%, respectively). Table 1 also illustrates 2.2%, 1.6%, 1.4%, 1.5%, 0.5%, 3.0%, 1.4% and 2.3% improvements for large-object segmentation (mIoU_L)

3. https://github.com/pytorch
4. https://github.com/qubvel/segmentation_models.pytorch

TABLE 1
The Comparison Results of Small Object Classes (Left) and Large Object Classes (Right) on CamVid Testing Dataset

Models	signsymbol	pedestrian	pole	bicyclist	$mIoU_S$	building	tree	sky	car	road	pavement	fence	$mIoU_L$	mIoU
DeepLabV3	53.6	57.8	37	65.9	53.6	80.6	74.7	89.6	84.2	93.8	79.8	46.8	78.5	69.4
$DeepLabV3_{AFMA}$	**57.5**	**58.4**	**40.1**	**68.2**	**56.1**	**82.6**	**76.2**	**89.7**	**87.2**	**94.5**	**82.0**	**52.9**	**80.7**	**71.8**
Unet	52.1	57.0	38.5	53.8	50.3	80.9	74.3	91.5	83.4	**93.0**	**79.4**	44.2	78.1	68.0
$Unet_{AFMA}$	**54.8**	**59.3**	**41.2**	**64.5**	**55.0**	**82.2**	**76.3**	**91.7**	**86.1**	92.9	79.2	**50.0**	**79.7**	**70.7**
Unet++	54.4	57.6	40.8	61.7	53.6	80.6	75.1	**92.2**	81.7	93.3	79.6	42.9	77.9	69.1
$Unet++_{AFMA}$	**57.3**	**60.6**	**42.7**	**65.8**	**56.6**	**82.1**	**76.0**	92.1	**84.1**	**93.3**	**81.1**	**46.7**	**79.3**	**71.1**
MaNet	51.2	58.4	36.1	63.4	52.3	80.4	**76.0**	91.4	82.7	91.4	77.6	45.8	77.9	68.6
$MaNet_{AFMA}$	**56.9**	**59.5**	**40.9**	**64.0**	**55.3**	**82.2**	75.3	**91.7**	**84.4**	**93.2**	**80.1**	**49.2**	**79.4**	**70.7**
FPN	49.0	57.5	39.7	62.6	52.2	80.5	75.7	**90.8**	83.6	93.9	80.2	**44.5**	78.4	68.9
FPN_{AFMA}	**54.1**	**59.8**	**40.1**	**64.9**	**54.7**	**81.4**	**76.5**	90.1	**84.1**	**94.1**	**81.8**	44.3	**78.9**	**70.1**
PAN	51.4	49.5	38.1	57.8	49.2	79.7	74	89.9	79.8	93	80.4	42	77.0	66.9
PAN_{AFMA}	**54.7**	**58.7**	**39.4**	**64.1**	**54.2**	**82.1**	**75.9**	**90.3**	**86.9**	**94.1**	**82.1**	**48.5**	**80.0**	**70.6**
LinkNet	51.2	52.7	38.3	64.3	51.7	80.2	**75.9**	**92.1**	85.2	**93.6**	**81.3**	40.8	78.4	68.7
$LinkNet_{AFMA}$	**53.7**	**62.0**	**42.0**	**65.1**	**55.7**	**81.8**	75.7	91.7	**85.9**	93.4	80.1	**49.8**	**79.8**	**71.0**
PSPNet	50.0	54.0	36.6	61.1	50.4	74.5	69.8	**90.7**	80.0	89.6	79.2	46.1	75.7	66.5
$PSPNet_{AFMA}$	**55.3**	**56.4**	**38.5**	**63.0**	**53.3**	**82.7**	**76.5**	90.0	**84.2**	**93.8**	**80.8**	**50.9**	**79.8**	**70.2**

For each object class, the number with red indicates that our method obtained better performance, and the blue indicates that the baseline method obtained better performance on the corresponding object class.

when applying AFMA to the DeepLabV3, Unet, Unet++, MaNet, FPN, PAN, LinkNet, and PSPNet, respectively. However, for some types of large objects, AFMA causes a slight degradation in the performance of the baseline models. For example, for the segmentation of sky, AFMA causes degradation of 0.1%, 0.7%, 0.4% and 0.7% for Unet++, FPN, LinkNet, and PSPNet, respectively. For the tree, road, and pavement, AFMA causes a performance loss of about 1% for some of the baseline models. The explanations are given in Section 4.8. The total mIoU improvements for baseline models are 2.4%, 1.3%, 2%, 2.1%, 1.2%, 3.7%, 2.3%, and 3.7%, respectively. The results of other existing small-object segmentation methods are shown in Supplementary Tables 1, and 2, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2022.3211171.

For the Cityscapes dataset, The upper table of Table 2 shows our method improves 2.1%, 3%, 4.8%, 3.8%, 4%, 3.8%, 2.9% and 4.1% improvements for small objects ($mIoU_S$) on DeepLabV3, Unet, Unet++, MaNet, FPN, PAN, LinkNet, and PSPNet baseline models, respectively. For the pole and traffic light, the AFMA improves the performance by more than 4% for Unet++, MaNet, FPN, PAN, and PSPNet. The bottom table of Table 2 shows 2.2%, 1.2%, 1.1%, 1.2%, 1.2%, 2.0%, and 1.6% improvements for large-object segmentation ($mIoU_L$) when using AFMA on the DeepLabV3, Unit, MaNet, FPN, PAN, LinkNet, and PSPNet, respectively. However, similar to the results on the CamVid dataset, AFMA causes a slight degradation in the performance of the Unet++ on Terrain (0.4%) and Truck (0.1%). The overall mIoU improvements for the baseline models are 2%, 1.8%, 1.7%, 2%, 2.2%, 2.2%, 2.3% and 2.5%.

To visually demonstrate the effectiveness of the proposed AFMA, Fig. 5 and supplementary material, available online, present representative segmentation results of all eight baseline methods with or without AFMA on the CamVid testing set. The examples illustrate that AFMA-based methods are more accurate for small-object classes, such as the distant car, pole, and sign symbols, which are poorly segmented or absent when utilizing only the baseline models. The segmentation results of the remote vehicle in the image given in Figs. 5a and 5b, for example, show that the eight baseline models cannot segment the car in the distance. But after adding the AFMA, these models can segment this small car. Figs. 5c and 5d show the examples of the model identifying sign symbols and poles. The examples show that the baseline models combined with our method can better identify the thin objects . Fig. 5 demonstrates that employing the proposed AFMA to the baseline segmentation network can better capture the missing components and render more accurate segmentation results.

4.4 Visualizing and Understanding AFMA

We demonstrate the impact of AFMA on segmentation by visualizing the attention maps attained by different models and explain why our models are better at recognizing small objects. Fig. 6a shows the AFMA between the original image patch containing a small car (red square in Fig. 6a) and all image patches of feature maps. From Eq. (7) in Section 3.2, we know that for each image patch in the initial image, our method will get N_c AFMAs between the image patch and all N_c category-related feature maps. From the 12 AFMAs of the small vehicle (as shown in Fig. 6), we could see that the original image patch of the small car has no relation with the feature maps of column pole, sidewalk, tree, sign symbol, fence, pedestrian, bicyclist. And it has weak associations with the feature maps of the sky, buildings, and road but has a strong relationship with the feature map of the category car. We zoom in on the car-related AFMA to the same resolution as the original image as shown in (Fig. 6b). We could find that the image patch of the small car in the original image has a strong association with the largest car (the car in the lower-left corner of the image). And it also has a relatively significant association with the next largest car (the car behind the largest car) and a relationship with the middle-size car (the car in the center of the image). Because AFMA learns the relationship between the small car in the distance and these larger cars, we can utilize the

TABLE 2
The Comparison Results of Small Object Classes on Cityscapes Testing Dataset

Models	Pole	Traffic light	Traffic sign	Person	Rider	Motorcycle	Bicycle	mIoU$_S$
DeepLabV3	67.4	73.7	77.7	83.7	70.5	68.8	73.7	73.6
DeepLabV3$_{\mathbf{AFMA}}$	**69.7**	**76.1**	**79.7**	**85.6**	**72.5**	**70.1**	**76.5**	**75.7**
Unet	66.3	73.4	76.7	82.9	68.7	64.3	72.8	72.1
Unet$_{\mathbf{AFMA}}$	**69.8**	**75.8**	**79.4**	**86.5**	**71.0**	**67.5**	**75.7**	**75.1**
Unet++	65.1	72.5	77.0	82.5	69.1	68.7	74.0	72.7
Unet++$_{\mathbf{AFMA}}$	**70.1**	**77.4**	**81.6**	**87.5**	**73.8**	**73.9**	**78.2**	**77.5**
MaNet	65.1	72.6	77.2	83.3	69.2	67.9	73.7	72.7
MaNet$_{\mathbf{AFMA}}$	**70.0**	**76.2**	**81.2**	**86.7**	**72.6**	**71.1**	**77.5**	**76.5**
FPN	64.4	71.8	75.8	82.7	69.1	66.9	73.4	72.0
FPN$_{\mathbf{AFMA}}$	**68.4**	**76.0**	**79.7**	**86.5**	**73.5**	**70.6**	**77.0**	**76.0**
PAN	61.8	68.8	73.8	82.3	66.2	63.3	72.4	69.8
PAN$_{\mathbf{AFMA}}$	**66.3**	**72.8**	**78.1**	**85.7**	**69.7**	**67.4**	**75.6**	**73.6**
LinkNet	63.3	70.1	74.9	82.7	67.3	63.4	72.5	70.6
LinkNet$_{\mathbf{AFMA}}$	**66.3**	**73.1**	**77.3**	**85.4**	**69.8**	**67.3**	**75.0**	**73.5**
PSPNet	63.0	71.3	75.8	82.5	67.1	65.8	72.7	71.2
PSPNet$_{\mathbf{AFMA}}$	**66.9**	**76.0**	**79.5**	**86.7**	**71.5**	**69.9**	**76.6**	**75.3**

Models	Road	Sidewalk	Building	Wall	Fence	Vegetation	Terrain	Sky	Car	Truck	Bus	Train	mIoU$_L$	mIoU
DeepLabV3	95.9	85.1	91.4	60.9	62.0	92.1	71.6	93.2	93.6	76.8	90.1	85.7	83.2	79.7
DeepLabV3$_{\mathbf{AFMA}}$	**97.8**	**87.2**	**94.0**	**63.7**	**64.6**	**94.2**	**73.7**	**95.3**	**96.0**	**79.3**	**91.7**	**87.7**	**85.4**	**81.9**
Unet	96.1	84.9	91.2	58.6	62.4	92.1	70.3	93.3	93.7	76.0	88.9	86.3	82.8	78.9
Unet$_{\mathbf{AFMA}}$	**97.6**	**86.3**	**92.3**	**60.5**	**63.2**	**92.9**	**71.8**	**95.0**	**95.2**	**77.0**	**89.9**	**86.6**	**84.0**	**80.7**
Unet++	96.3	84.6	91.0	55.6	61.0	91.6	**71.0**	92.8	93.9	**78.6**	90.6	86.4	82.8	79.1
Unet++$_{\mathbf{AFMA}}$	**96.5**	**84.3**	**91.5**	**54.8**	**61.2**	**91.6**	70.6	**93.0**	**94.3**	78.5	**90.7**	**86.6**	**82.8**	**80.8**
MaNet	97.0	84.3	91.6	53.3	61.7	92.5	71.1	94.4	95.0	74.0	89.3	83.9	82.3	78.8
MaNet$_{\mathbf{AFMA}}$	**98.5**	**85.7**	**92.8**	**55.1**	**63.1**	**92.8**	**72.3**	**95.2**	**96.1**	**74.2**	**89.7**	**84.9**	**83.4**	**80.8**
FPN	95.4	83.7	90.9	61.3	60.9	90.3	70.0	92.7	92.6	76.9	88.3	85.3	82.3	78.5
FPN$_{\mathbf{AFMA}}$	**96.3**	**85.1**	**91.6**	**62.7**	**61.7**	**91.7**	**71.1**	**93.0**	**93.6**	**78.6**	**89.7**	**86.4**	**83.5**	**80.7**
PAN	96.8	84.0	91.8	57.7	59.2	92.1	69.9	93.7	94.5	75.4	88.2	82.8	82.2	77.6
PAN$_{\mathbf{AFMA}}$	**98.4**	**85.5**	**92.3**	**59.1**	**60.3**	**92.7**	**71.0**	**94.6**	**95.6**	**77.1**	**89.1**	**84.6**	**83.4**	**79.8**
LinkNet	96.4	83.1	91.0	55.4	58.6	90.5	70.2	93.2	93.8	73.5	87.7	81.0	81.2	77.3
LinkNet$_{\mathbf{AFMA}}$	**98.1**	**85.5**	**92.5**	**56.9**	**61.0**	**93.0**	**71.7**	**94.7**	**95.8**	**76.1**	**90.1**	**83.0**	**83.2**	**79.6**
PSPNet	96.7	85.9	91.5	56.7	61.8	92.0	70.7	94.0	94.4	76.3	89.7	81.9	82.6	78.4
PSPNet$_{\mathbf{AFMA}}$	**99.1**	**87.3**	**93.1**	**58.3**	**63.9**	**94.1**	**71.9**	**95.0**	**95.8**	**77.6**	**91.6**	**83.2**	**84.2**	**80.9**

The number with red indicates AFMA combined method obtained the better performance. The upper table shows the small-object results and the bottom table shows the results for large objects.

predicted results of the large cars to correct the results of the image patch location representing the small car. The similar AFMA obtained by four baseline models in Fig. 6 shows that our approach can steadily learn the relationship between original image and feature maps. The results of all other models including MaNet, DeepLabV3, Unet, and Unet++ are shown in supplementary Fig. 1, available online.

The AFMA learned from different image patches containing other objects is further given in Fig. 7, where we only show the most significant AFMA obtained by FPN, PAN, PSPNet, and LinkNet. The results of other baseline methods and the AFMA associated with all different categories are shown in the supplementary material, available online. Fig. 7 shows that the AFMA obtained from four original image patches containing sky (blue square), tree in the distance (red square), building (purple square), and road (gold square), respectively. As shown in Fig. 7 and Supplementary Figs. 2– 5, available online, all models learn the relationships between the original image patch and the feature maps of the corresponding category. For example, for the image patch representing the sky in the original image (blue square in Fig. 7), FPN, PAN, PSPNet, and LinkNet models obtain the relationships between that image patch and the feature map of the sky. These relationships are noticeable, and the borders of the trees and buildings that the sky touches are also visible. For the image patch representing the distant tree (red square), all models can obtain a clear association between this patch and the feature map of the large tree in the initial image. For example, we can see that the contours of the larger tree from the AFMA of PSPNet and LinkNet models. Similarly, for the patches of the road (gold square) and building (purple square), our models learn the relationship between them and their corresponding category-related feature maps. As can be seen in Fig. 7, the AFMA associated with roads and buildings are clearly outlined.

The above examples show that for a specific type of object, our method learns the relationship between the original image patch containing a small amount of information

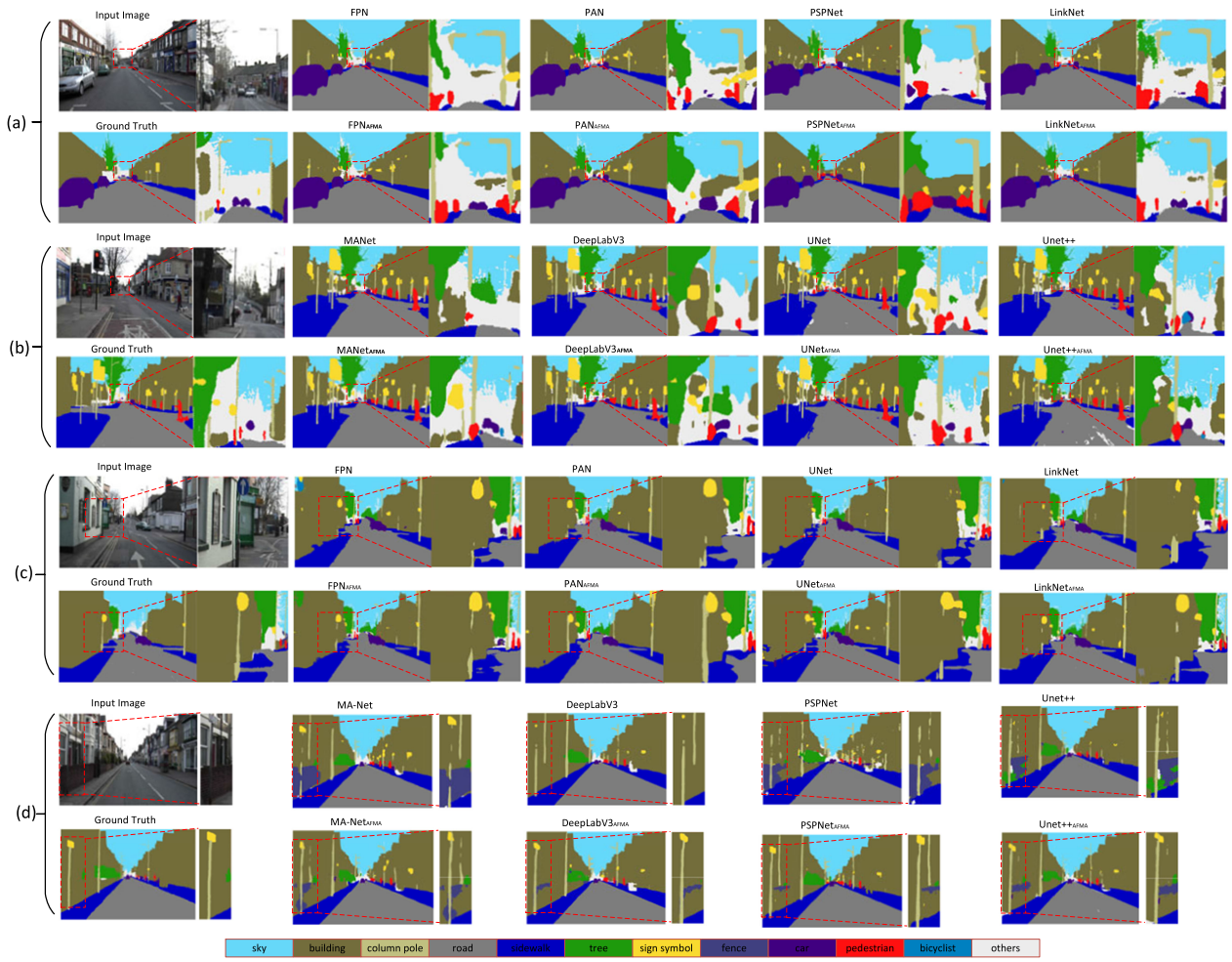


Fig. 5. Examples of semantic segmentation results on CamVid testing dataset. For visualization purpose, we zoomed in on the segmentation results and the dashed rectangles are enlarged for highlighting improvements. (*a*) Examples of FPN, PAN, PSPNet, LinkNet and our methods for segmenting the distant car. (*b*) Examples of MaNet, DeepLabV3, Unet, Unet++ and our methods for segmenting the distant car. (*c*) Examples of FPN, PAN, Unet, LinkNet and our methods of segmenting the poles and sign symbols . (*d*) Examples of MaNet, DeepLabV3, PSPNet, Unet++ and our methods for segmenting the poles and sign symbols. The first and second row of each sub-figure present the results of baseline models and our models, respectively.

about that object and the feature map patch containing a large amount of information about the object's category, which yields the relationship between the small and large objects of the same type in the original image. Since existing semantic segmentation methods work well for large object segmentation, we use the results of large objects to guide the results of pixels in the locations of small objects.

4.5 On the Depth of AFMA

The AFMA proposed in this paper acts on both the original image and the feature maps of the encoder. As shown from Fig. 2b, existing encoding backbones such as residual networks include multiple feature map layers of varying dimensions, which allows the AFMA to be combined with the original segmentation model in various ways. In this section, we evaluate the impact of the depth of AFMA on segmentation performance. We first conducted experiments on baseline models combining AFMA obtained from different depth feature maps. Second, we qualitatively evaluated the impact of depths of AFMA on segmentation by visualizing attention maps. Third, we explored the effect of the depth of AFMA on various types of semantic segmentation models.

First, Table 3 presents the results of AFMA utilizing feature maps of varying depths. According to the table: 1) All AFMA models of different depths improve the results of the baseline segmentation models, demonstrating that leveraging the relationship between the original image and the feature maps can steadily improve the segmentation model's performance. 2) Using deeper feature maps is beneficial for identifying large objects, such as using AFMA with depth 3 or 4, Unet, Unet++, MaNet, LinkNet, PSPNet all achieve better results in large object segmentation than the AFMA model with depth 2. It's because, for the same size of feature patches, the feature patch of deeper feature maps contains larger objects than the shallow feature maps' patch. 3) AFMA obtained from shallower feature maps can get better results for some types of small objects, such as sign symbols, bicycles, and pedestrians. For example, AFMA with depth 2 achieves the best results in DeepLabV3, MaNet, PAN, and PSPNet for sign symbols. However, the overall results show no particularly significant difference in the performance of AFMA with different depths for small object segmentation, which may be primarily since the small objects in the close distance in the image have a larger size. For example, the

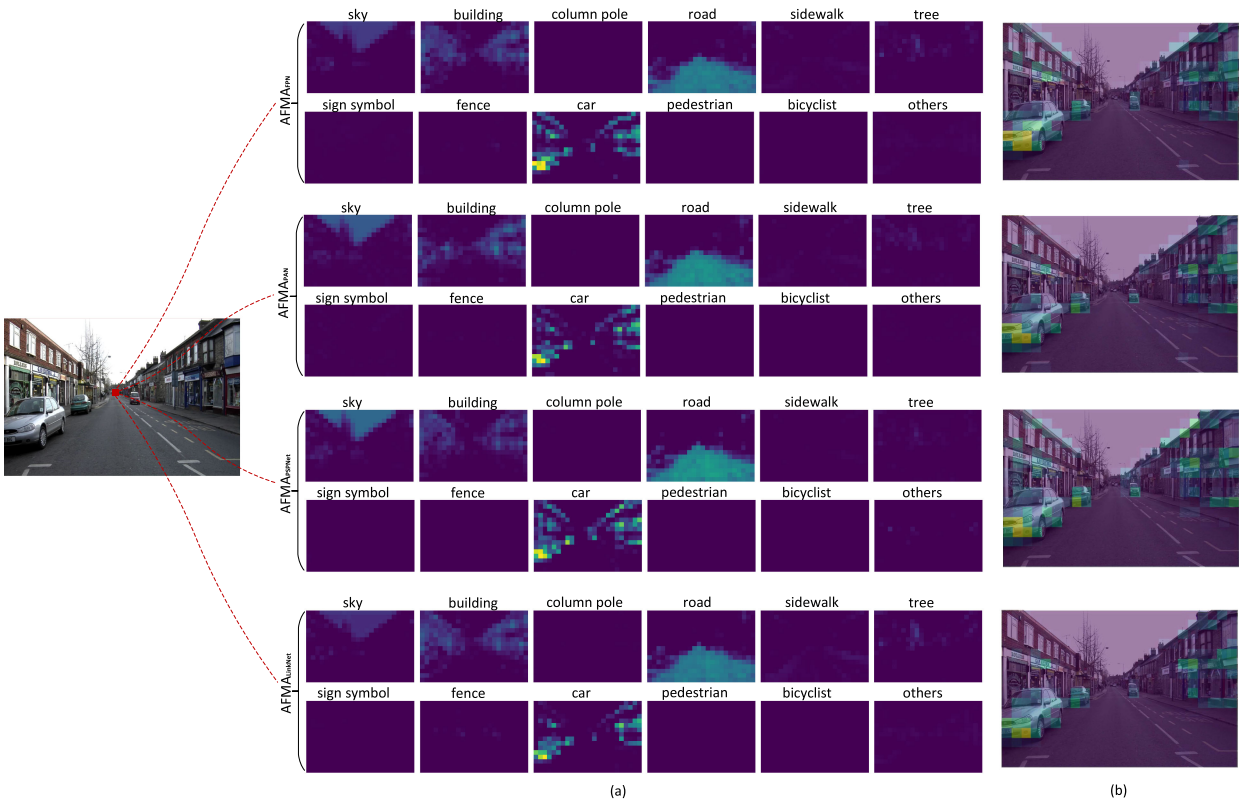


Fig. 6. An illustration of AFMA. (*a*) The left is the original image, and the red rectangle is the image patch containing the small car in the distance. The AFMAs of the red image patch of FPN, PAN, PSPNet, and LinkNet are shown in the middle part. Since there are 12 categories in CamVid, we could know from Section 3.2 that our method obtained 12 AFMA of each category for the image patch. (*b*) The AFMAs of the car category are enlarged and superimposed to the initial input image for visualization purpose.

pedestrians or poles in the close have a larger size, which is not small objects in the strict sense.

Second, Fig. 8a shows that the AFMA learned by the different depths of feature maps. From Fig. 8a, we can see that 1) The deeper depth of AFMA has a coarser attention map because the deeper layer of feature maps has less resolution and, therefore, fewer feature map patches for computing the AFMA. For example, Fig. 8a shows that as the depth of AFMA increase from 2 to 4, the road, sky, and building-related AFMA of FPN, LinkNet, MaNet, PAN, Unet, and Unet++ have less and less resolution. However, we still can recognize the shape of the road, sky, and building. 2) The deeper depth of AFMA has a more straightforward relationship with large objects. The car-related AFMA of depth 2, as shown in Fig. 8a, indicates that the small car not only has associations with the large car in the lower-left corner but also has some incorrectly weak associations with the objects in the upper left and right side image. As the depth of AFMA increases, the relationship between the small car and the large car becomes more evident. When the depth is 3, for example, AFMA shows that the small car has a significant association with the lower-left corner of the original image. And as the depth grows to 4, the AFMA shows that the small car is mainly related to the block containing the large car in the lower-left corner. The AFMA of FPN, Manet, and Unet++, for example, has only three points of non-zero values covering the large car in the lower-left corner, and the AFMA of LinkNet has only one point left directly indicating a more evident relationship. From the car-related examples of FPN and LinkNet in Fig. 8b, we could see that, when the depth is 2, the AFMA covers cars and some parts of buildings. But as the depth increases to 3, the AFMA mainly covers the large vehicle in the left corner, and when the depth is 4, the AFMA only covers the large car. Similarly, for the distant tree, the deeper depth of AFMA, the more explicitly it indicates the association between the small tree in the distance and the large tree in close. For example, the tree-related AFMA of FPN and LinkNet in Figs. 8a and 8b shows that when the depth of AFMA is 2, the AFMA of the distant small tree covers the large tree in the image and a small number of other surrounding objects. As the depth increases, the AFMA of the small tree points more and more clearly to the large tree in the initial image. For example, when the depth is 4, the AFMA of both FPN and LinkNet has only one point (Fig. 8a) pointing to the large tree (Fig. 8b). It might also explain why the performance of AFMA for large object segmentation improves as the feature map layer deepens. Because as the depth increases, feature patches of the same size contain more information of larger objects. Other detailed examples could be found in Supplementary Fig. 6, available online.

Third, the resolution of AFMA of different models does not always decrease when increasing the attention depth. For example, Fig. 8a shows that for both PSPnet and Deeplabv3, the AFMA with depths 3 and 4 have the same resolution. Because the encoding part of PSPnet and DeepLabV3 adopts atrous/dilated convolution, resulting in the same size of the 3rd and 4th feature maps (as shown in Fig. 2b). Therefore the resolution of AFMA for the models with pyramid pooling or dilated convolution does not necessarily

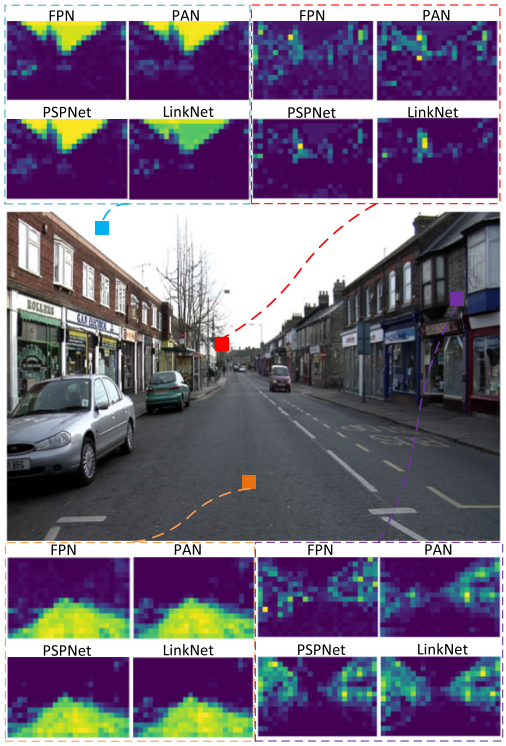


Fig. 7. An illustration of AFMA of different types of image patches. The blue, red, purple, and golden image patches in the initial image point to sky, tree in the distance, building, and road, respectively. The figure shows the four image patches' AFMA obtained by FPN, PAN, PSPNet, and LinkNet.

decrease when increasing the AFMA depth. Although the AFMA of depth 3 and 4 have the same resolution, Fig. 8a and supplementary Fig. 6, available online, show that the deeper AFMA has more clear relationships. For example, compared to the AFMA of depth 3, the AFMA of depth 4 of PSPNet and DeepLabV3 show more clearly that the small car has relationships with the larger car. It also shows more evident associations between the small and large trees in the tree-related AFMA of Fig. 8a.

The detailed size distribution of each category of the CamVid training set

4.6 Analysis on Class-Dependent Depth for AFMA

As described in Section 4.5, the variance of object sizes would still influence the model performance. Since deeper feature patches contain larger objects, the patches associated with a given object are related to the object's size and the depth of feature maps. In this section, we explore the impact of AFMA depth on the categories with different size distributions. Following the criteria of the COCO dataset[55], all objects are classified as small, medium, and large based on their area. The detailed size distribution of the CamVid training set is in Fig. 9. We grouped all categories into three groups: "small group" contains sign symbol, pole, pedestrian, bicyclist, and fence. As shown in Fig. 9, more than half objects in each above category are small in size. Similarly, the "medium group" contains pavement, trees, and cars. And "large group" includes sky, buildings, and roads. We trained different models with varying AFMA depths for each group separately, and the results are in Supplementary Table 3, available online. The results show that: 1) The AFMA at a shallower depth facilitates the segmentation of categories with a predominantly small size. For example, DeepLabv3, Unet, Unet ++, MaNet, LinkNet, and PSPNet, with AFMA depth 2, achieved the best mIoUs for the "small group". Furthermore, these models perform more clearly for sign symbols and

TABLE 3
The Comparison Results of AFMA With Different Attention Depth on Small Object Classes (Left) and Large Object Classes (Right) on CamVid Testing Dataset

Models	signsymbol	pedestrian	pole	bicyclist	mIoUs	building	tree	sky	car	road	pavement	fence	mIoUL	mIoU
$DeepLabV3_2$	**57.5**	58.4	**40.1**	**68.2**	**56.1**	82.6	76.2	89.7	87.2	**94.5**	**82.0**	52.9	80.7	71.8
$DeepLabV3_3$	55.9	**59.3**	39.6	66.8	55.4	**83.0**	**76.9**	**89.8**	**87.8**	**94.5**	82.1	**56.7**	**81.6**	**72.0**
$DeepLabV3_4$	54.4	59.1	38.7	66.1	54.6	82.0	76.6	**89.8**	85.4	94.3	81.5	46.8	79.5	70.4
$Unet_2$	54.8	59.3	41.2	64.5	55.0	82.2	76.3	91.7	**86.1**	92.9	79.2	50.0	79.7	70.7
$Unet_3$	**58.4**	**62.8**	41.2	62.3	56.2	82.5	**77.1**	**91.9**	84.9	**94.1**	**82.2**	**53.6**	**80.9**	**71.9**
$Unet_4$	57.2	62.2	**42.4**	**65.8**	**56.9**	**82.9**	77.0	91.4	**86.1**	93.6	80.1	53.0	80.6	**71.9**
$Unet{+}{+}_2$	57.3	60.6	42.7	**65.8**	56.6	82.1	76.0	92.1	84.1	93.3	81.1	46.7	79.3	71.1
$Unet{+}{+}_3$	54.9	**64.1**	**42.9**	65.3	**56.8**	82.2	76.8	92.1	85.2	**94.5**	**83.1**	**51.4**	**80.8**	**72.0**
$Unet{+}{+}_4$	**58.8**	62.8	42.7	62.6	56.7	**82.8**	**76.9**	**92.4**	**86.9**	94.1	82.1	44.9	80.0	71.5
$MaNet_2$	**56.9**	59.5	40.9	64.0	55.3	82.2	75.3	**91.7**	84.4	93.2	80.1	49.2	79.4	70.7
$MaNet_3$	56.3	59.0	40.5	65.7	55.4	**82.5**	**77.0**	91.1	84.0	**93.5**	**81.2**	**54.1**	**80.5**	71.3
$MaNet_4$	55.2	**60.8**	**42.6**	**66.4**	**56.3**	82.3	75.7	91.6	**87.3**	**93.5**	80.3	50.3	80.1	**71.4**
FPN_2	54.1	**59.8**	40.1	64.9	54.7	81.4	**76.5**	90.1	84.1	**94.1**	**81.8**	44.3	78.9	70.1
FPN_3	53.2	59.5	**41.6**	63.4	54.4	81.6	75.8	**90.6**	85.4	93.4	79.7	**49.6**	79.4	70.3
FPN_4	**54.7**	59.0	40.7	**65.9**	**55.1**	**81.9**	76.2	90.3	**86.7**	94.0	81.1	47.7	**79.7**	**70.7**
PAN_2	**54.7**	**58.7**	39.4	**64.1**	**54.2**	**82.1**	75.9	90.3	**86.9**	**94.1**	**82.1**	**48.5**	**80.0**	**70.6**
PAN_3	52.9	57.9	**41.0**	61.4	53.3	81.8	**77.3**	90.2	83.8	93.5	80.6	47.1	79.2	69.8
PAN_4	52.7	58.3	40.5	63.4	53.7	81.6	75.7	**90.5**	85.6	93.7	80.3	45.4	79.0	69.8
$LinkNet_2$	53.7	**62.0**	42.0	**65.1**	55.7	81.8	75.7	91.7	85.9	93.4	80.1	**49.8**	79.8	71.0
$LinkNet_3$	55.8	58.9	40.7	63.6	54.7	82.0	**76.5**	**91.9**	86.3	**93.7**	**81.0**	47.9	**79.9**	70.8
$LinkNet_4$	**56.4**	60.1	**42.8**	64.0	**55.9**	**82.3**	76.1	91.8	**87.5**	92.8	78.1	**49.8**	79.8	**71.1**
$PSPNet_2$	**55.3**	56.4	38.5	63.0	53.3	82.7	**76.5**	90.0	84.2	93.8	80.8	50.9	79.8	70.2
$PSPNet_3$	54.4	57.1	39.0	**63.8**	**53.6**	82.5	76.1	90.1	**85.4**	**94.3**	**81.9**	**53.9**	**80.6**	**70.8**
$PSPNet_4$	55.1	**57.3**	**40.2**	60.2	53.2	**82.8**	76.2	**90.4**	84.0	94.2	81.8	51.2	80.1	70.3

FOR each object class, the number with the best performance are highlighted in red.The subscript number of each method indicates the depth of AFMA.

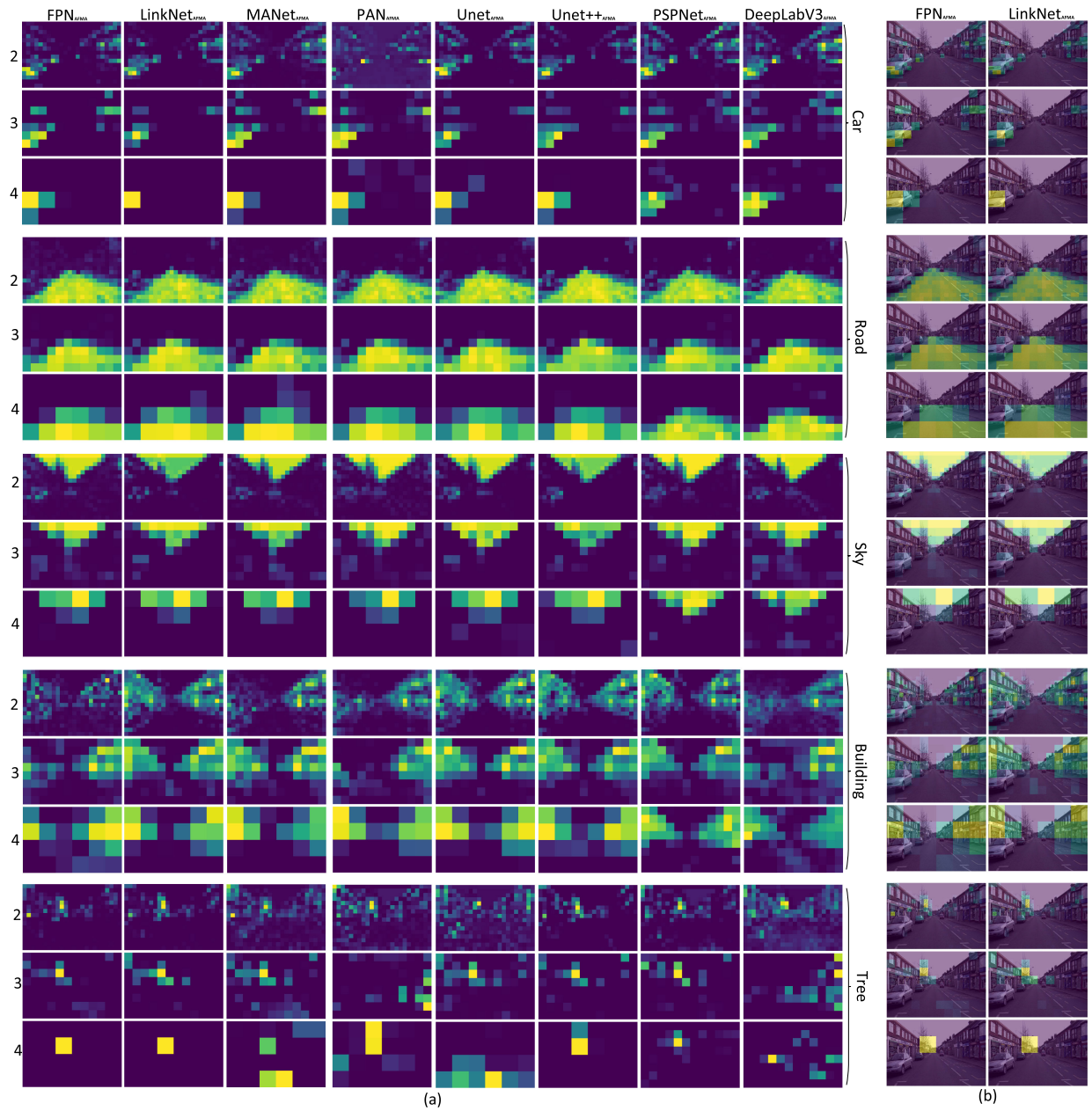


Fig. 8. An illustration of AFMA between different depth of feature maps. (*a*) shows the AFMA obtained from different depth feature maps for image patches containing car, road, sky, building, and tree, respectively. The original image patches containing car, road, sky, building, and tree are the same as shown in Fig. 7. (*b*) shows examples of alignments between the AFMA of FPN, LinkNet and the initial image.

poles. This is because about 80% of sign symbols and poles are small objects (Fig. 9), the information of these objects may be better preserved in the shallower feature patches. 2) In general, the AFMA at a deeper depth is beneficial for segmenting large categories. For example, DeepLabV3, Unet, MaNet, FPN, PAN, and PSPNet with depths 3 or 4 achieve better results than the models with depth 2 in "medium group" and "large group". This is because features obtained at deeper layers correspond to larger receptive fields. The deeper feature patches contain more information about large objects, it might be beneficial for calculating the relationship between the initial patch and feature map patches of large object.

The performance of AFMA is related to the size distribution of objects and the depth of feature maps. The AFMA depths should be considered when designing models for categories with dominating size distributions. In this study, we only set AFMA at a fixed depth, and developing different AFMA depths for classes with varying distributions is our future work.

4.7 Performance on Objects Grouped by Pixel Size

Section 4.3 evaluates all models on the small/large objects grouped by object category. However, objects of the large-object category, such as cars, might be shown in small objects at far distances and vice versa. In this section, we evaluate our method on small/large objects grouped by pixel size. All objects are classified as small, medium, and large based on the same criteria adopted in Section 4.6. Table 4 shows the quantitative results of all models on the Camvid dataset. It demonstrates that the AFMA-based models significantly outperform baseline methods on small

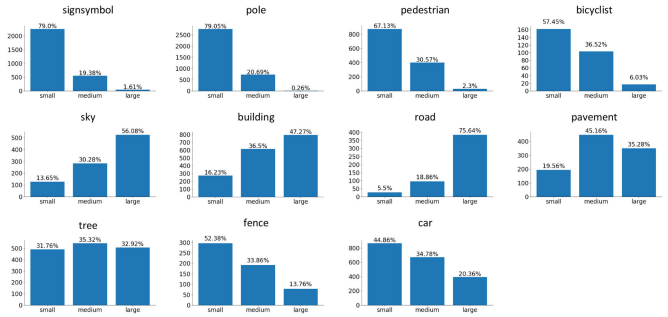


Fig. 9. The size distribution of each category of the CamVid training set. The vertical coordinate indicates the number of objects. The small, medium, and large represent the size of an object less than 32^2 pixels, between 32^2 and 96^2 pixels, and larger than 96^2 pixels, respectively. Detailed image processing and statistical methods are given in the supplementary material, available online.

object segmentation. For example, DeepLabV3, Unet, Unet++, MaNet, FPN, PAN, LinkNet, and PSPNet obtain 4.0%, 4.0%, 3.2%, 3.8%, 2.8%, 3.4%, 4.1%, and 4.7% improvement after combing AFMA, respectively, on small object segmentation after combining AFMA. The results on Cityscapes and the detailed confusion matrix of all methods is summarized in Supplementary Table 4, Supplementary Figs. 16 and 17, respectively, available online. Tables 1 and 4 show similar results: the AFMA could significantly enhance baseline model performance on small objects and a slight improvement on non-small object segmentation, no matter the small/large object definition criteria. This is because the small/large objects grouped by category or pixel size have a similar object distribution. For example, Fig. 9 shows that like the small objects (sign symbol, pole, pedestrian, and bicyclist) defined by category in Section 4.3, most sign symbols, poles, pedestrians, and bicyclists are small pixel-size objects. For example, about 80% of sign symbols and poles are less than 32^2 pixels. And similar to the large objects defined by category, more than 80% of the sky, buildings, roads, and pavements are non-small objects (medium or large). The object size distribution of CamVid's validation, testing, and CityScape dataset can be found in Supplementary Figs. 10 to 15, available online.

TABLE 4
The Comparison Results of Small/Non-Small Objects Grouped by Pixel Size on the CamVid Testing Dataset

Model Name	Object Size		
	small	medium	large
Deeplabv3	14.4	36.0	84.8
DeepLabV3$_{\mathbf{AFMA}}$	**18.4** (4.0%%↑)	**36.9** (0.9%↑)	**86.1** (1.3%↑)
Unet	15.7	35.7	84.6
Unet$_{\mathbf{AFMA}}$	**19.7** (4.0%↑)	**38.2** (2.5%↑)	**85.5** (0.9%↑)
Unet++	17.8	36.9	84.8
Unet++$_{\mathbf{AFMA}}$	**21.0** (3.2%↑)	**38.9** (2.0%↑)	**85.5** (0.7%↑)
MaNet	14.3	34.0	83.9
MaNet$_{\mathbf{AFMA}}$	**18.1** (3.8%↑)	**37.1** (3.1%↑)	**85.4** (1.5%↑)
FPN	15.2	35.6	84.9
FPN$_{\mathbf{AFMA}}$	**18.0** (2.8%↑)	**36.6** (1.0%↑)	**85.4** (0.5%↑)
PAN	13.9	34.1	84.2
PAN$_{\mathbf{AFMA}}$	**17.3** (3.4%↑)	**37.5** (3.4%↑)	**85.9** (1.7%↑)
LinkNet	15.1	34.9	84.8
LinkNet$_{\mathbf{AFMA}}$	**19.2** (4.1%↑)	**37.0** (2.1%↑)	**85.4** (0.6%↑)
PSPNet	14.7	30.1	81.0
PSPNet$_{\mathbf{AFMA}}$	**19.4** (4.7%↑)	**32.4** (2.3%↑)	**81.6** (0.6%↑)

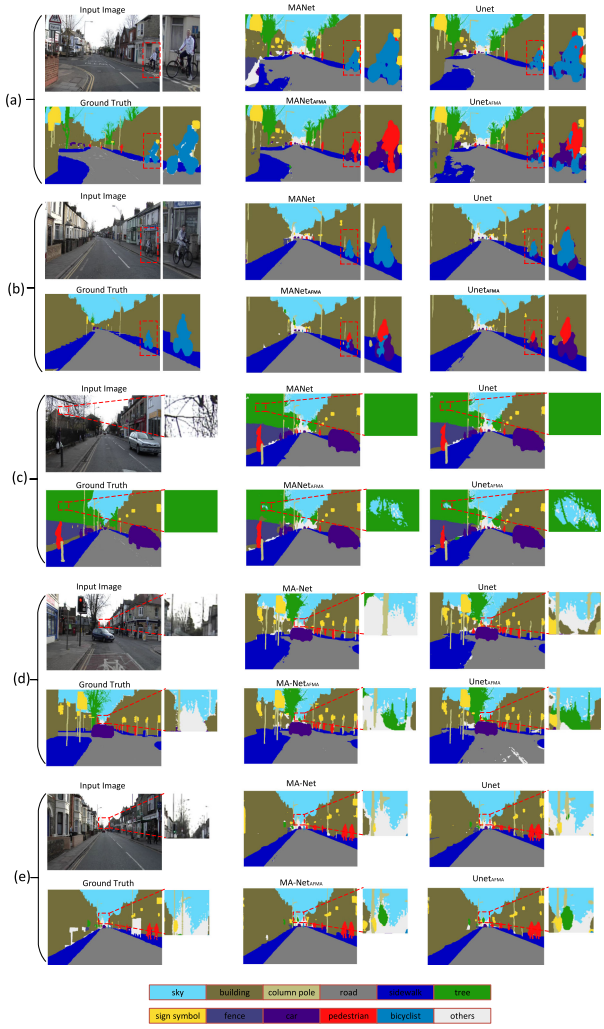


Fig. 10. An illustration of some negative cases of our method, which may be attributable to the mechanism of AFMA. (a) and (b) show examples of the segmentation of bicyclist. (c) Examples of the segmentation of the sky in tree branches. (d) and (e) present examples of false negative results.

4.8 Negative Cases Study

According to Tables 1 and 2, AFMA improves all small object segmentation performance, but it decreases performance in some large objects such as trees, the sky, and cars. In this section, we present a few negative cases of our method, which may be attributable to the mechanism of AFMA. 1) AFMA's segmentation of combined categories (e.g., bicyclist = human + bycicle) is not good. Fig. 10a shows that MANet and Unet properly segment the bicyclist in the original image. However, our methods segment the bicyclist as a pedestrian and a car separately. Because the definition of bicyclist in CamVid consists of a pedestrian and a bicycle, a bicycle that appears alone is labeled as "car". The whole is considered a bicyclist when someone is riding it. Our method computes the AFMA independently for the person and bicycle of the bicyclist, resulting in the model giving the person on the bicycle more similarities to pedestrians and the bicycle more associations to cars. Fig. 10b shows another similar example. Figs. 10a and 10b show that our method pays more attention to the basic subclasses of a combining category, resulting in the false segmentation of combing categories. Other examples could be found in Supplementary Fig. 7, available online.

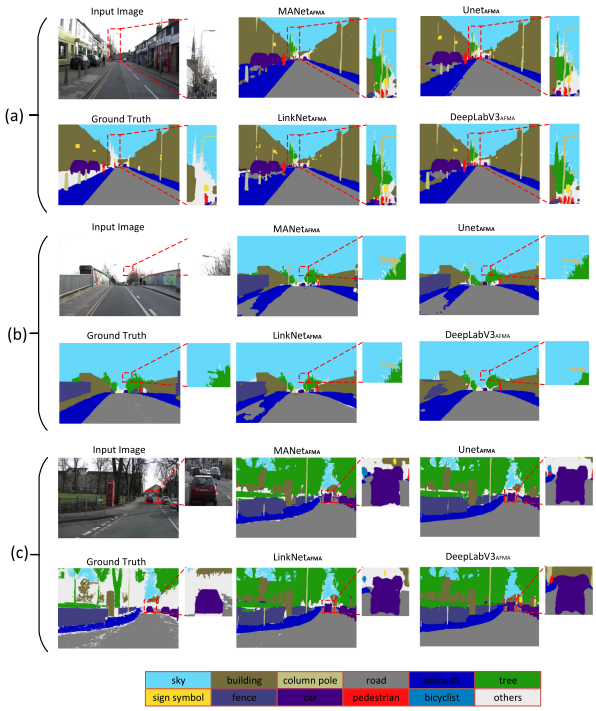


Fig. 11. False predictive examples. (a) and (b) show the segmentation of the distant trees and poles which don't exist in the groundtruth. (c) Segmenting the small cars that groundtruth doesn't label.

2) AFMA loses some local relations and generates locality bias. Fig. 10c shows that the baseline methods segment the sky in the tree branches into trees as same as the groundtruth. After combining AFMA, the models segment the tree branches and the sky separately as tree and sky. It's because AFMA partitions the tree branches and sky of the original image into different image patches, and AFMA adds extra relations to the image patch containing sky with sky-related feature maps. Supplementary Fig. 7, available online, depicts further examples of this kind.

3) AFMA segments objects in the test set that are not labeled. We found that the inconsistency of the small objects' groundtruth labels causes some negative cases. For example, as shown in Figs. 10d and 10e, our method segments the distant small trees and distant small poles, which are not labeled in the testing set. Fig. 11 presents the examples of our method segmenting tiny trees (Fig. 11a), poles (Fig. 11b), and cars (Fig. 11c) in the distance that don't exist in the groundtruth. It shows our method is effective in recognizing various types of small objects. We provide more such negative examples in the Supplementary material (Figs. 7 and 8), available online.

4.9 Parameter Analysis

Since larger models often achieve better results than smaller models in deep learning, we examine the number of parameters in baseline and our models in this section. According to Section 3, the AFMA increases performance primarily by computing the relationship of different levels of feature maps of the baseline model's encoder, which does not contain a significant number of parameters. Table 5 and supplementary Table 5, available online, present the size of all models, demonstrating that AFMA only increases a small number of parameters for the baseline models.

TABLE 5
The Parameter Size and FLOPs

Model	#params (M)	Incre (%)	#FLOPs (G)	Incre (%)
Deeplabv3	58.62	0.10	566.43	2.35
DeepLabV3$_{\mathbf{AFMA}}$	58.68		579.76	
Unet	51.51	0.10	146.59	2.23
Unet$_{\mathbf{AFMA}}$	51.57		159.91	
Unet++	67.97	0.11	585.63	8.45
Unet++$_{\mathbf{AFMA}}$	68.03		598.01	
MaNet	166.43	0.11	221.06	8.33
MaNet$_{\mathbf{AFMA}}$	166.49		234.39	
FPN	45.10	0.09	118.99	2.26
FPN$_{\mathbf{AFMA}}$	45.16		132.23	
PAN	43.25	0.09	127.28	2.23
PAN$_{\mathbf{AFMA}}$	43.31		140.61	
LinkNet	50.17	0.04	146.51	6.03
LinkNet$_{\mathbf{AFMA}}$	50.23		159.83	
PSPNet	48.85	0.04	433.65	7.29
PSPNet$_{\mathbf{AFMA}}$	48.91		450.75	

The FLOPs on CamVid is reported on the 720x960.

The increasing percentage of parameters ranges from 0.04% to 0.14%. The highest increase (0.11%) is observed in Unet++$_{\mathrm{AFMA}}$ and Manet$_{\mathrm{AFMA}}$ on the Cityscapes dataset, while LinkNet$_{\mathrm{AFMA}}$ and PSPNet$_{\mathrm{AFMA}}$ has the least increment (0.04%). Table 5 and Supplementary Table 5, available online, show that the models for the CamVid and Cityscapes datasets are the same size since the models used for the two datasets have the same parameter settings (for example, the patch size of their AFMA is all 10×10). The results reveal that the improvement of the model is not due to the increase in the model size. The FLOPs of all models are reported in Table 5 and Supplementary Table 5, available online. The growth of all FLOPs is almost less than 10%. The Unet on Cityscapes has the highest growth of 11.51%, while PAN and Unet on CamVid have the smallest growth rate of 2.3%.

The figure of training convergence (Fig. 12) shows that the mIoU improves as the number of epochs increases. After 300 epochs, the loss of all approaches converges. In conclusion, the increased number of parameters and FLOPs demonstrate

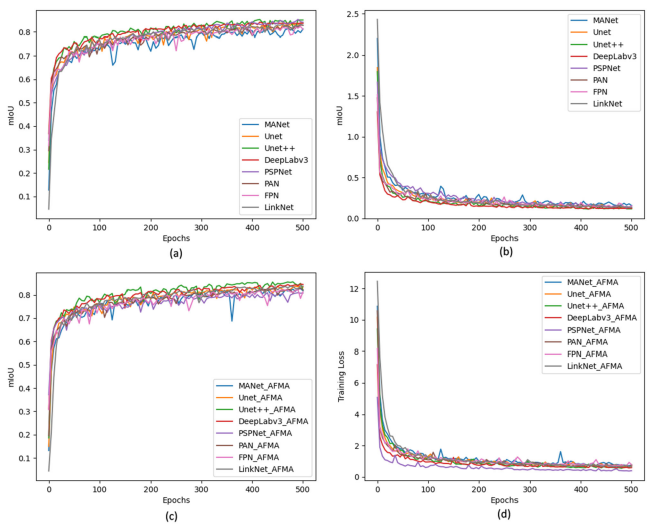


Fig. 12. Training convergence. (*a*) The mIoU of all methods over 500 training epochs on CamVid training dataset. (*b*) The training loss of all methods over 500 training epochs.

that the suggested AFMA could efficiently improve segmentation performance.

5 Discussion

Why not Directly Resize the Input Image to Obtain AFMA? Since the sizes of different objects might be substantially different, our method quantifies the relation between small and large objects by exploiting the original image and its corresponding feature maps. Another more straightforward solution is perhaps to directly resize the original image to a smaller size, then the image patch of the same size can contain larger objects. Similarly, we can also quantify the relation between small and large objects based on the original input and resized images (according to Section 3.2). However, we empirically find that computing the AFMA via resizing the input image leads to much lower performance than our method. This might be due to that the feature map enjoys larger receptive fields and more enriched semantic information than the image patch (of the same size) from the resized image.

The Shape/Size of the Image/Feature Patch. We note that the square image/feature patches can be extended to any shape/size. For example, as shown in Vision Transformer [46] and SwinTransformer [56], the image patch size could be 4×4, 16×16, and 32×32. Furthermore, different categories may favor different patch shape/size. For example, the car category might favor square image/feature patches whereas the column pole object might favor slender rectangle image/feature patches when computing AFMA. We will explore the influence of the size and shape of patches on the segmentation performance in our future work.

What if an Image Only has One Object for a Given Class? Our method exploits the relation among objects to compensate for the information loss of the small object. However, in some applications, there may be only one object of a specific type in an image. For example, a CT image often only contains one organ, such as liver, heart, etc. Here we evaluate the models' performance on objects of each category that individually appear in CamVid and Cityscapes. The results in Supplementary Table 6, available online, show that our approach still improves the performance of all the baseline models on small object segmentation. The reason might be that, for the two datasets, only a small number of objects appear individually (the detailed statistics are in Supplementary Tables 7 and 8, available online). Most images contain more than two objects of the same class, and our method could still learn the relations between categories by other large amounts of co-occur objects. In addition, we tested our method on datasets containing only single object. Interestingly, we empirically find that our method still improves the segmentation performance on liver segmentation (LiTS[5]), skin lesion (Skin Lesion Analysis Towards Melanoma Detection [57]) and birds (The Caltech-UCSD Birds [58]) datasets. For example, PAN with AFMA improves 1.9% mIoU on skin lesion segmentation, and MaNet with AFMA improves 3.6% mIoU on bird segmentation. The detailed results could be found in Supplementary Table 9 and Supplementarty Fig. 9, available online. We believe this is because AFMA can enhance the unrelated relation between the target object and other image patches, therefore eliminating the target object's false positive prediction.

6 Conclusion

This paper proposes Across Feature Map Attention (AFMA), to improve the performance of existing semantic segmentation models for segmenting small objects. The technique first partitions the original image and its feature maps into image patches of the same size. Then it computes attention between the image patches from various level feature maps to obtain the relations between small and large objects. The obtained attention is used to improve the performance of semantic segmentation. The experiment results show that our method can substantially improve the segmentation accuracy of small objects as well as improve the overall segmentation performance. The proposed method is evaluated based on MaNet, Unet, Unet++, LinkNet, PSPNet, PAN, DeepLabV3, FPN, and other existing small-object segmentation methods on CamVid and Cityscapes. Our method achieves considerable improvement on small object segmentation compared with existing methods. Moreover, we also provide a deeper analysis of the experimental results to better understand the mechanism of AFMA. The proposed AFMA is a lightweight model, which can be easily combined with numerous existing segmentation networks while only incurring neglectable additional training/testing time or expense in deployment.

Acknowledgments

Shengtian Sang and Yuyin Zhou are given their equal contribution.

References

[1] S. A. Taghanaki, K. Abhishek, J. P. Cohen, J. Cohen-Adad, and G. Hamarneh, "Deep semantic segmentation of natural and medical images: A review," *Artif. Intell. Rev.*, vol. 54, no. 1, pp. 137–178, 2021.

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

[3] X. Li, H. Zhao, L. Han, Y. Tong, S. Tan, and K. Yang, "Gated fully fusion for semantic segmentation," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11 418–11 425.

[4] Y. Li, Q. Huang, X. Pei, Y. Chen, L. Jiao, and R. Shang, "Cross-layer attention network for small object detection in remote sensing imagery," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 2148–2161, Dec. 2021.

[5] C. Huynh, A. T. Tran, K. Luu, and M. Hoai, "Progressive semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16 750–16 759.

[6] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.

[7] R. Hamaguchi, A. Fujita, K. Nemoto, T. Imaizumi, and S. Hikosaka, "Effective use of dilated convolutions for segmenting small object instances in remote sensing imagery," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2018, pp. 1442–1450.

[8] S. Ramos, S. Gehrig, P. Pinggera, U. Franke, and C. Rother, "Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 1025–1032.

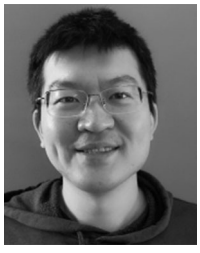
[9] Z. Meng, X. Fan, X. Chen, M. Chen, and Y. Tong, "Detecting small signs from large images," in *Proc. IEEE Int. Conf. Inf. Reuse Integr.*, 2017, pp. 217–224.

[10] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1951–1959.

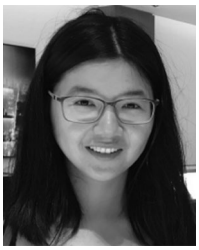
[11] K. Gupta, S. A. Javed, V. Gandhi, and K. M. Krishna, "MergeNet: A deep net architecture for small obstacle discovery," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 5856–5862.

5. https://competitions.codalab.org/competitions/17094

[12] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5325–5334.
[13] X. Chen et al., "3D object proposals for accurate object class detection," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 424–432.
[14] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Interv.*, 2015, pp. 234–241.
[15] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan, "PixelNet: Towards a general pixel-level architecture," 2016, *arXiv:1609.06694*.
[16] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 447–456.
[17] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*.
[18] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6230–6239.
[19] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, "DenseASPP for semantic segmentation in street scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3684–3692.
[20] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," 2015, *arXiv:1511.07122*.
[21] 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.
[22] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with Gaussian edge potentials," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 109–117.
[23] S. Chandra and I. Kokkinos, "Fast, exact and multi-scale inference for semantic image segmentation with deep Gaussian CRFs," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 402–418.
[24] S. Zheng et al., "Conditional random fields as recurrent neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1529–1537.
[25] D. Guo, L. Zhu, Y. Lu, H. Yu, and S. Wang, "Small object sensitive segmentation of urban street scene with spatial adjacency between object classes," *IEEE Trans. Image Process.*, vol. 28, no. 6, pp. 2643–2653, Jun. 2019.
[26] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
[27] A. Chaurasia and E. Culurciello, "LinkNet: Exploiting encoder representations for efficient semantic segmentation," in *Proc. IEEE Vis. Commun. Image Process.*, 2017, pp. 1–4.
[28] X. Xia and B. Kulis, "W-Net: A deep model for fully unsupervised image segmentation," 2017, *arXiv:1711.08506*.
[29] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 173–190.
[30] 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.
[31] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: A nested U-Net architecture for medical image segmentation," in *Proc. Int. Workshop Deep Learn. Med. Image Anal. Multimodal Learn. Clin. Decis. Support*, 2018, pp. 3–11.
[32] H. Seo, C. Huang, M. Bassenne, R. Xiao, and L. Xing, "Modified U-Net (mU-Net) with incorporation of object-dependent high level features for improved liver and liver-tumor segmentation in CT images," *IEEE Trans. Med. Imag.*, vol. 39, no. 5, pp. 1316–1325, May 2020.
[33] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Interv.*, 2016, pp. 424–432.
[34] S. Shah, P. Ghosh, L. S. Davis, and T. Goldstein, "Stacked U-Nets: A no-frills approach to natural image segmentation," 2018, *arXiv:1804.10343*.
[35] X. Zhang et al., "DCNAS: Densely connected neural architecture search for semantic image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13 951–13 962.
[36] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," 2016, *arXiv:1606.02147*.
[37] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3523–3542, Jul. 2022.
[38] H. Li, P. Xiong, J. An, and L. Wang, "Pyramid attention network for semantic segmentation," 2018, *arXiv:1805.10180*.
[39] T. Fan, G. Wang, Y. Li, and H. Wang, "MA-Net: A multi-scale attention network for liver and tumor segmentation," *IEEE Access*, vol. 8, pp. 179 656–179 665, 2020.
[40] T. Wang et al., "Detect globally, refine locally: A novel approach to saliency detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3127–3135.
[41] N. Liu, J. Han, and M.-H. Yang, "PiCANet: Learning pixel-wise contextual attention for saliency detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3089–3098.
[42] Z. Chen, C. Guo, J. Lai, and X. Xie, "Motion-appearance interactive encoding for object segmentation in unconstrained videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1613–1624, Jun. 2020.
[43] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Learning a discriminative feature network for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1857–1866.
[44] X. Hu, K. Yang, L. Fei, and K. Wang, "ACNet: Attention based network to exploit complementary features for RGBD semantic segmentation," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 1440–1444.
[45] X. Zhang, T. Wang, J. Qi, H. Lu, and G. Wang, "Progressive attention guided recurrent network for salient object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 714–722.
[46] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
[47] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 88–97, 2009.
[48] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3213–3223.
[49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
[50] L. Ladickỳ, C. Russell, P. Kohli, and P. H. Torr, "Associative hierarchical CRFs for object class image segmentation," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 739–746.
[51] H. Lu, G. Fang, X. Shao, and X. Li, "Segmenting human from photo images based on a coarse-to-fine scheme," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 3, pp. 889–899, Jun. 2012.
[52] J. Tighe and S. Lazebnik, "SuperParsing: Scalable nonparametric image parsing with superpixels," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 352–365.
[53] B. Liu and X. He, "Multiclass semantic video segmentation with object-level active inference," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4286–4294.
[54] X. Li et al., "FoveaNet: Perspective-aware urban scene parsing," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 784–792.
[55] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
[56] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," 2021, *arXiv:2103.14030*.
[57] N. C. Codella et al., "Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC)," in *Proc. IEEE 15th Int. Symp. Biomed. Imag.*, 2018, pp. 168–172.
[58] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD birds-200–2011 dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2011-001, 2011.



Shengtian Sang received the PhD degree from the College of Computer Science and Technology, Dalian University of Technology, Dalian, China. He is currently a post-doctoral scholar with the Laboratory of Artificial Intelligence in Medicine and Biomedical Physics, Department of Radiation Oncology, Stanford University. His current research interests include medical data mining, medical image computing, and machine learning. In his PhD study, he worked on biomedical literature-based discovery and data mining.



Yuyin Zhou received the PhD degree from the Computer Science Department, Johns Hopkins University, in 2020, and was a postdoctoral researcher with Stanford University from 2020 to 2021. She is currently an assistant professor of computer science and engineering with UC Santa Cruz. Her research interests span the fields of medical image computing, computer vision, and machine learning, especially the intersection of them. She has more than 20 peer-reviewed publications at top-tier conferences and journals including CVPR, ICCV, AAAI, the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Medical Imaging*, MedIA, etc. She has led the ICML 2021 workshop on Interpretable Machine Learning in Healthcare, the ICCV 2021 workshop on Computer Vision for Automated Medical Diagnosis, and co-organized ML4H 2021, the 9th CVPR MCV workshop. She served as a senior program committee for IJCAI 2021 and AAAI 2022, an area chair for MICCAI 2022, CHIL 2022.

Md Tauhidul Islam (Student Member, IEEE) received the BSc and MSc degrees in electrical and electronic engineering from the Bangladesh University of Engineering and Technology (BUET), Dhaka, in 2011 and 2014, respectively. He is a post-doctoral scholar with the Laboratory of Artificial Intelligence in Medicine and Biomedical Physics, Department of Radiation Oncology, Stanford University. In his PhD study, he worked on ultrasound elastography at Ultrasound and Elasticity Imaging Laboratory, Department of Electrical Engineering, Texas A&M University. His current research interests include high dimensional medical data analysis using deep learning, manifold embedding, and interpretability of deep neural networks. His past research interests were in diverse areas of biomechanics, ultrasound imaging, elastography, and signal processing.

Lei Xing received the PhD degree in physics from the Johns Hopkins University, Baltimore, MD, USA, in 1992. He completed the Medical Physics Training with the University of Chicago, Chicago, IL, USA. He is currently the Jacob Haimson & Sarah S. Donaldson professor of medical physics and the director of the Medical Physics Division, Radiation Oncology Department, Stanford University, Stanford, CA, USA. He also holds affiliate faculty positions with the Department of Electrical Engineering, Bio-X and Molecular Imaging Program, Stanford University. He has been a member of the Radiation Oncology Faculty, Stanford University, since 1997. His current research interests include medical imaging, artificial intelligence in medicine, treatment planning, image-guided interventions, nanomedicine, and applications of molecular imaging in radiation oncology.