Scale-Aware Segmentation of Multiple-Scale Objects in Aerial Images

1st Jingbo Lin
College of Information and Computer Engineering
Northeast Forestry University
Harbin, China
ljbxd180612@gmail.com

2nd Weipeng Jing
College of Information and Computer Engineering
Northeast Forestry University
Harbin, China
weipeng.Jing@outlook.com

3rd Houbing Song
Department of Electrical, Computer, Software, and Systems Engineering
Embry-Riddle Aeronautical University
Daytona Beach, FL 32114 USA
Houbing.Song@erau.edu

4th Guangsheng Chen
Northeast Forestry University
Harbin, China
kjc_chen@163.com

***Abstract*—Semantic segmentation is a fundamental task extensively used in the analysis of high-resolution aerial images. Due to the objects with various scales widely appearing in high-resolution aerial images, the fixed square receptive field in most existing DCNNs cannot work well and usually leads to unexpected predictions. To alleviate this problem, we propose a light-weight scale-aware module (SWD), which is end-to-end differentiable and can be embedded in most existing networks. In our module, we employ the re-sampling scheme to make each element of convolutional patches adjust its position, which explicitly adjusts the receptive field size to fit different scales of target objects. Further, considering re-sampled feature maps as weighted maps implicitly introduces spatial attention mechanism. As a result, with only 0.2M additional parameters, the network embedded with our module can automatically adjust its receptive field to better recognize objects in various scales. In the experiment, we evaluate our method on the ISPRS Vaihingen Dataset, especially the analysis of buildings category, which usually in different scales. We further compare it to the spatial attention mechanism and mainstream networks that utilize multi-scale information. The experimental results and comprehensive analysis demonstrate the effectiveness and efficiency of our proposed method.**

***Index Terms*—Neural Networks, Feature Extraction, Semantic, Image Segmentation, Training, Urban Areas.**

I. Introduction

Semantic segmentation is important in high-resolution aerial images. It is widely used in environmental monitoring and urban planning. With the development of remote sensing technologies, advanced sensors can provide more and more high-resolution and high-quality images. Due to the limitation of expressing capacity, the conventional segmentation method that heavily depends on handcraft feature can no longer work well with large-scale dataset and the complex feature of ground objects. In recent years, the deep convolutional neural networks (DCNNs) have shown their outstanding performance in many vision tasks. From the milestone work of Long *et al.* [1] in 2014, the fully convolutional neural networks (FCNs) have been extensively employed in semantic segmentation tasks. Inspired by Ronneberger *et al.* [2], the architecture of encoder-decoder is widely used and has many variations. To get better predictions, the DeepLab family utilize dense conditional random field (dense-CRF) as post-processing and propose atrous spatial pyramid pooling (ASPP) module [3] to aggregate multi-scale feature better. Further, RefineNet [4] adopt multi-stage architecture to refine the confidence map stage by stage.

Different from the indoor and outdoor scenes, high-resolution aerial images usually have more complex objects with large margin difference in scale. For DCNNs, the lack of receptive field will cause incomplete identification of large-scale objects; Too large receptive field will introduce much irrelevant information, and it usually leads to small-scale objects unrecognized. Therefore, the existing methods used for scene parsing tasks cannot be directly transferred. There still exist challenging issues in semantic segmentation of remote sensing domain. The networks used for segmentation in high-resolution aerial images should have scale-aware ability.

Multi-scale information aggregation is a conventional method that makes network scale-aware available. Li *et al.* [5] adopt the weighted summation method in their adaptive multi-scale deep fusion module. Lu *et al.* [6] propose a supervised strategy, utilizing the semantic label to aggregate information progressively. Different from above methods that are focused on the aggregation strategy, we introduce the scale-aware ability by an embedded module, and there are some similar works for typical scene parsing tasks in recent years. Wei *et al.* [7] use two affine transformation layers to regulate the size of receptive field automatically. Similarly, Zhang *et al.* [8] introduce scale-adaptive mechanism by additional scale regression layers that can dynamically infer the position-adaptive scale coefficients to shrink or expand the convolutional patches. However, these methods are adopted in typical scene parsing tasks. Furthermore, these methods have inherited limitation since the scale coefficient treats the horizontal and vertical axes

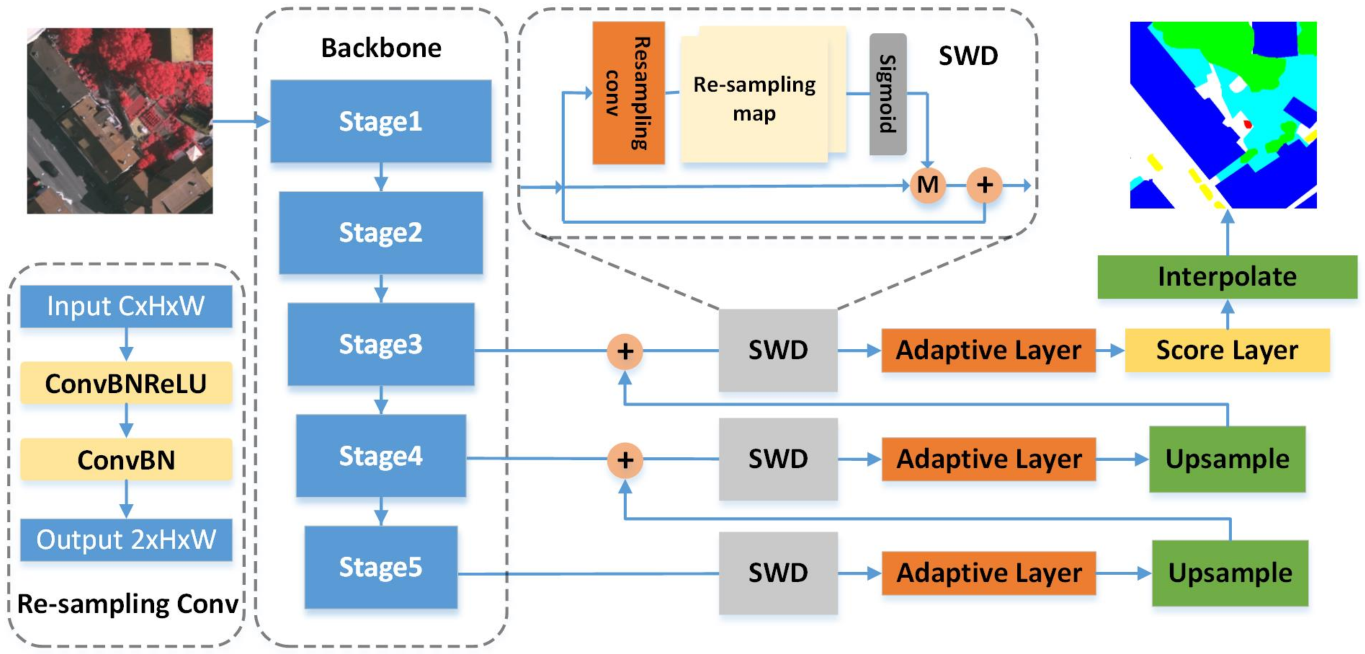

Fig. 1. The overall architecture of FCN8s embedded with scale-aware module (SWD). 'M' means element-wise multiplication, '+' means element-wise addition, 'Rconv' represents re-sampling convolutions, and 'adaptive layer' represents conv1x1 that used to uniform channel dimension.

equally in 2D space, the coordinates of pixels in convolutional patches can only zoom in or out in same proportion. Under this situation, the revised receptive field cannot fit well with the objects in various scales, the recognition will be affected by the interference information in limited revised receptive field (an example is given in Section III.C). Therefore, there needs a method that treats horizontal and vertical axes differently to better recognize multiple-scale objects in high-resolution aerial images.

In this paper, we propose a scale-aware module (SWD), which can be trained end-to-end and easily embedded in most existing networks. Different from the previous works, our module alleviates the limitation by learning two-dimensional re-sampling maps that consider horizontal and vertical axes differently. In our module, each pixel of convolutional patches is re-sampled based on the re-sampling maps, which depends on both network parameter and input data. That is to say, our module has not only scale-aware ability to fit objects with various scales but also has data-aware ability to adapt different input data. We evaluate our module on ISPRS Vaihingen Dataset and the experimental results show the effectiveness of our proposed module.

The rest of the paper is organized as follows. Section II presents our proposed methodology. Experimental settings and results are provided in Section III. Section IV concludes the paper.

II. Methodology

A. Overview

The overall architecture of FCN embedded with our module is shown in Fig. 1. As a feature extractor, the backbone can be most of networks used for classification. Following with FCN8s [1], the down sampling rate is 32, and we employ SWD at the end of each stage. We add channel Adaptive Layers after each SWD that perform channel projection to aggregate multi-stage feature. At last, we use a simple score layer and up sampling layer to get the final prediction.

B. Scale-Aware Module

The structure of our module (SWD) is shown in the top of Fig. 1. In this subsection, we will elaborate the implementation details and feasibility of end-to-end training of our module. First of all, our module learns the re-sampling information by re-sampling convolutions (shown in the left of Fig. 1). We take $I \in \mathbb{R}^{h \times w \times c}$ as input, where h and w is spatial size of feature maps, c is the number of input channels. Going through re-sampling convolutions, we will get the re-sampling information $S \in \mathbb{R}^{h \times w \times 2}$ for the whole input feature maps I in horizontal and vertical axes, and then the full re-sampling maps is obtained by merging with re-sampling information S. It is worth noting that, we initialize the re-sampling maps by mapping the coordinates of each pixel in original feature maps to range of [-1, 1]. That is, the coordinate of top-left position is (-1, -1) and the bottom-right position is (1, 1). In order to guarantee pixels tune their spatial position based on the original and avoid excessive offset, we employ normal distribution $\mathcal{N}$ with small *std* as initialization strategy of the re-sampling convolutional layers (following [8]), it can be formulated as:

$$\omega \sim \mathcal{N}(0, \sigma^2), \sigma \ll 1 \tag{1}$$

where ω is weights of re-sampling convolutional layers. We take σ as 0.001 (avoiding the large offset) and ignore the bias term [9] in our experiments (for simplicity and efficiency). Moreover, we also perform *clamp* operation during training to avoid the unexpected excessive offset. Under this initialization strategy, the generated re-sampling information is small, and the re-sampled pixels can move start from original position gradually during training.

After generating the re-sampling maps, our module re-samples pixels of the original feature maps based on the re-sampling maps. Inspired by the work of Zhang *et al.* [8], we also adopt bilinear interpolation algorithm as our sampling algorithm. Considering the filters $F \in \mathbb{R}^{k_h \times k_w \times c_1}$ and the corresponding convolutional patches $P \in \mathbb{R}^{k_h \times k_w \times c_2}$, where k_h and k_w is the spatial size of filters, c_1 and c_2 is the number of channels of filters and convolutional patches, respectively. The standard convolution without the SWD-transformed feature maps can be expressed as:

$$conv(F,P) = \sum_{c_1,c_2} \sum_{i,j} (f^{ij,c_1} \times p^{ij,c_2}) \tag{2}$$

where f^{ij,c_1} and p^{ij,c_2} are the pixels at the position (i, j) of c_1-th, c_2-th channel in filters F and corresponding patches P, respectively. Due to our module performs on spatial-wise, we only consider the spatial dimension in the following formulations for simplicity. Mapping the spatial position of pixels in original feature maps into [-1, 1], the coordinates of pixel p^{ij} in re-sampling maps are:

$$\begin{cases} x^{ij} = ori_x + s_x \\ y^{ij} = ori_y + s_y \end{cases} \tag{3}$$

where (ori_x, ori_y) is the coordinate of pixel in the original feature maps that mapped into [-1, 1], s_x and s_y is horizontal and vertical re-sampling information learned by re-sampling convolutions. Then we perform bilinear interpolation $bili$ to get re-sampled feature maps based on re-sampling maps, the value of pixels in re-sampled feature maps can be calculated:

$$\begin{cases} val(p^{ij}) = \sum_{q \in Q} bili(p^{ij}, q) \cdot q \\ bili(p^{ij}, q) = bili_x(x^{ij}, q_x) \cdot bili_y(y^{ij}, q_y) \end{cases} \tag{4}$$

where bilinear interpolation has two dimensions $bili_x$ and $bili_y$, $bili_{x|y}(p,q) = max(0, 1 - |p_{x|y} - q_{x|y}|)$ and Q denotes the pixels that participant interpolation. Instead of directly using the re-sampled feature maps to participate the following calculation, we employ residual learning scheme [10]. The re-sampled feature maps are converted to weighted maps by sigmoid layer, then it performs element-wise multiplication with the original feature maps. On one hand, it maintains the original information and better merges with the re-sampling information. On the other hand, the residual learning is benefit for training and optimizing. Following (2) (3) (4), the convolution with SWD-transformed feature maps can be expressed:

$$\begin{cases} conv(F,P) = conv(F, trans(P)) \\ trans(P) = P + P \cdot sig(val(rsmap(P))) \end{cases} \tag{5}$$

where $conv(\cdot,\cdot)$ is standard convolutional operation followed (2), $trans(\cdot)$ represents the SWD-transformation, $rsmap(\cdot)$ denotes the calculation of the re-sampling maps as (3), and sig means $sigmoid$ function. To elaborate the differentiable of our proposed method, we denote M_F as the corresponding matrix of filters F in spatial-wise, and M_P as the related convolutional patches. For standard convolution, the output O is obtained by the forward propagation:

$$O = conv(M_F, M_P) = M_F \cdot M_P \tag{6}$$

Where O is output feature maps. In backward propagation, the gradient of output O is given as $g(O)$, then the derivation of each variable can be obtained:

$$\begin{cases} g(M_P) = (M_F)^T \cdot g(O) \\ g(M_F) = g(O) \cdot (M_P)^T \end{cases} \tag{7}$$

where $g(\cdot)$ denotes gradient function, and $(\cdot)^T$ is the *matrix transposition*. We denote $g_{var}(A)$ as partial derivative of A w.r.t. var. According to (2) (5) (6) (7), the derivation of each variable in convolution with SWD-transformed feature maps:

$$\begin{cases} g(trans(M_P)) = (M_F)^T \cdot g(O) \\ g_{M_P}(trans(M_P)) = 1 + Z(M_P) + M_P \cdot g_{M_P}(Z(M_P)) \\ g(M_P) = g(trans(M_P)) \cdot g_{M_P}(trans(M_P)) \\ g(M_F) = g(O) \cdot trans(M_P)^T \end{cases} \tag{8}$$

For convenience, we denote Z as $sig(val(rsmap(\cdot)))$. According to (8), we can see that if Z is differentiable then our method is differentiable and can be trained end-to-end. We have already known $sigmoid(x)$ is differentiable, the derivative of x is $sig(x) \cdot (1 - sig(x))$. We then prove Z is differentiable. Following (4), the partial derivative of $val(M_P^{ij})$ w.r.t. q is:

$$\frac{\partial val(M_P^{ij})}{\partial q} = bili_x(x^{ij}, q_x) bili_y(y^{ij}, q_y) \tag{9}$$

In order to avoid repeated description, we only consider partial derivative of $val(M_P^{ij})$ w.r.t. horizontal ordinate of pixel p^{ij}:

$$\frac{\partial val(M_P^{ij})}{\partial x^{ij}} = \sum_q q \cdot bili_y(y^{ij}, q_y) g_{x^{ij}}(bili_x(x^{ij}, q_x)) \tag{10}$$

$$g_{x^{ij}}(bili_x(x^{ij}, q_x)) = \begin{cases} 0, & |x^{ij} - q_x| \geq 1 \\ 1, & x^{ij} > q_x, |x^{ij} - q_x| < 1 \\ -1, & x^{ij} < q_x, |x^{ij} - q_x| < 1 \end{cases} \tag{11}$$

Furthermore, the coordinates of pixels in re-sampling maps are generated by the original coordinates (ori_x, ori_y) and the re-sampling information (s_x, s_y), which is learned by the re-sampling convolutions. The above analysis demonstrates that our method is differentiable, and the parameters of our proposed module can be learned end-to-end from the dataset without any additional supervisions. Moreover, the residual-block like structure makes training and optimizing relatively easy [10]. Employing re-sampled feature maps as weighted maps let our module implicitly introduce spatial-wise attention mechanism. From this perspective, it also makes sense to explain the scale-aware ability of our proposed module. More importantly, our module makes networks have data-aware ability. In conventional methods that applied DCNNs, the performance only depends on the parameters of network, which are learned from training data. But in our method, the re-sampling maps not only depends on re-sampling convolutions (network parameters) but also depends on the input. Therefore, the networks with our module not only have better scale-aware ability but also are adaptive to different input images.

III. Experiments

A. General Setup

1) Dataset: We evaluate our method on the Vaihingen dataset. It contains 33 patches with average size of around 2500×2000. The dataset has been classified into six classes, including impervious surfaces, buildings, low vegetation, tree, car and clutter/background. We follow the test set provided by the official, we adopt five patches (image id 11, 15, 28, 30, 34) as validation set and the rest of patches as training set. We crop the large patches into slices of 512×512 with 50% overlapped window. We adopt randomly horizontal flip as data augmentation strategy. In addition to the metrics given by the contest, we also use IoU as our metrics, they are formulated as:

$$IoU = \frac{TP}{TP + FP + FN}, F_1 = \frac{2TP}{2TP + FP + FN} \quad (12)$$

2) Configuration: All the experiments were done with CUDA 9.0 and PyTorch 1.1.0. The networks run 200 epochs on Tesla V100, using Adam optimizer with weight decay of 2e-4 and momentum of 0.9. The initial learning rate is 5e-4 and we adopt ”ploy” learning rate policy with power of 0.9. The batch size is 16. We adopt cross entropy loss with weight $W_{class} = \frac{1}{log(P_{class}+c)}$ and we set c to 1.12 as [11].

B. Ablation Studies

In this section, we conduct sets of ablation studies based on the popular framework FCN8s to demonstrate the effectiveness of our module, and we choose pre-trained ResNet34 as our network backbone. The comparison results are shown in Table I. To validate the effectiveness of our module, we firstly embed single SWD before the last score layer of FCN8s as FCN8s-SWD-S, and there has slightly boost (about 0.3%) for each category compared with vanilla FCN8s. Further, we employ our module at the end of each stage of FCN8s as FCN8s-SWD-M, the large margin improvement (about 2%) demonstrates applying our module in multi-stages can get much better performance. Since our module implicitly introduces spatial attention mechanism by considering the re-sampled feature maps as weighted maps (as analyzed in Section II.B), there is a question: is the improvement benefit from the attention mechanism or our proposed scale-aware module? To prove our method is different from the spatial attention mechanism and its effectiveness, we replace SWD in FCN8s-SWD-S and FCN8s-SWD-M into spatial attention module while maintaining the number of parameters unchanged, and we name them as FCN8s-SWD-SC, FCN8s-SWD-MC, respectively. As a result, our method performs much better than the attention mechanism in all categories.

C. Visualization Analysis

As Fig. 2 shown, we further generate class activation mapping (CAM) of building category (which is in different scales) for FCN8s and FCN8s-SWD-M to understand our method. In the first two columns, vanilla FCN8s cannot recognize the small buildings because their feature is dominated by the surrounding objects in the large receptive field, but our module can not only identify the small buildings but also is more sensitive to the areas of large buildings than FCN8s. The recognition of slender objects is a challenging issue in semantic segmentation, these objects need the large receptive field to get enough context information, but the large receptive field with square shape includes too much irrelevant information that usually leads to incorrect classification. In the third column, FCN8s cannot classify the building with slender shape completely, but our module has a better response to the entire building. For large buildings in the fourth and fifth columns, the activation response of FCN8s is concentrated at a part of buildings, but the activation response of our method is relatively more uniform. Thus, our module can perform better on both slender and large objects. Considering the most special case, the buildings occupy most of the imagery, the lack of receptive field will lead to misclassification. As the last column shown, FCN8s has low response in the central part but high response at the boundaries of buildings, so FCN8s can easily misclassify the central region into impervious surface. However, our module still has an expected activation response for the whole building. Hence, our method is more flexible.

D. Evaluation and Comparisons

As shown in Table II, we compare our module with other popular networks based on pre-trained ResNet34, including light-weight RefineNet, DeepLabv3, DeepLabv3+, PSPNet, and UNet. There are also some other methods perform well on Vaihingen Dataset like [12]. However, it is not trained end-to-end and include complicate data augmentation strategy. So we only compare with the networks that utilize multi-scale information. Light-weight RefineNet get worse accuracy compared to other networks, due to its network capacity. DeepLabv3, DeepLabv3+ and PSPNet get similar accuracy by employing atrous spatial pyramid pooling module. Although the performance of UNet is much better than the others, the mirror-like encoder-decoder structure makes it consume large amount of computational resources. With only $0.2M$ additional parameters and without the decoder of UNet, our network achieve similar performance to the heavy-weight UNet in all metrics except car category. Therefore, our method can achieve similar or better recognition accuracy compared to the methods that employ well-designed multi-scale feature fusion module while maintaining the efficiency. In addition, we only adopt IRRG (IR: near infrared, R: red, and G: green) images as training data without DSM or normalized DSM, these additional features, deeper and wider networks, the refinement of decoder, and the post processing will further improve the segmentation accuracy of our method.

TABLE I
THE EXPERIMENT RESULTS OF ABLATION STUDIES, PER-CLASS IOU(%), PER-CLASS F1-SCORE(%), MEAN IOU(%), MEAN F1-SCORE(%), AND OVERALL ACCURACY(%).

Networks	Imp surf		Building		Low Veg		Tree		Car		Avg.		Acc
	IoU	F1	IoU	F1	IoU	F1	IoU	F1	IoU	F1	mean IoU	mean F1	
FCN8s	73.07	84.29	80.23	88.90	56.02	71.46	70.47	82.42	51.10	67.32	66.18	78.88	83.76
FCN8s-SWD-SC	73.13	84.31	80.93	89.33	56.22	71.61	70.63	82.53	53.40	69.24	66.86	79.41	83.98
FCN8s-SWD-S	73.21	84.39	80.77	89.23	56.47	71.79	70.77	82.64	51.17	67.33	66.48	79.08	83.94
FCN8s-SWD-MC	73.25	84.40	80.78	89.23	57.11	72.34	71.50	83.20	51.64	67.68	66.86	79.37	84.17
FCN8s-SWD-M	**75.28**	**85.77**	**82.25**	**90.13**	**58.81**	**73.75**	**72.46**	**83.82**	**53.97**	**69.76**	**68.55**	**80.65**	**85.12**

TABLE II
THE COMPARISON WITH POPULAR NETWORKS WHICH UTILIZE MULTI-SCALE INFORMATION, PER-CLASS IOU(%), PER-CLASS F1-SCORE(%), MEAN IOU(%), MEAN F1-SCORE(%), AND OVERALL ACCURACY(%).

Networks	Imp surf		Building		Low Veg		Tree		Car		Avg.		Acc
	IoU	F1	IoU	F1	IoU	F1	IoU	F1	IoU	F1	mean IoU	mean F1	
LWRefineNet	65.83	79.22	73.69	84.70	47.84	64.21	63.28	77.20	34.88	51.16	57.10	71.30	78.52
DeepLabv3	73.23	84.40	79.80	88.64	54.34	70.02	69.32	81.64	50.58	66.69	65.45	78.28	83.22
DeepLabv3+	73.15	84.33	79.68	88.56	54.39	70.05	69.47	81.74	50.38	66.46	65.41	78.23	83.21
PSPNet	72.79	84.10	80.86	89.26	56.16	71.45	70.87	82.69	47.02	63.68	65.54	78.24	83.87
UNet	**75.37**	**85.79**	81.86	89.86	58.81	73.72	72.09	83.56	**59.54**	**74.29**	**69.53**	**81.44**	85.09
Ours	75.28	85.77	**82.25**	**90.13**	**58.81**	**73.75**	**72.46**	**83.82**	53.97	69.76	68.55	80.65	**85.12**
Ours-resnet101	77.49	87.18	84.53	91.51	61.06	75.54	73.36	84.43	60.03	74.67	71.29	82.67	86.47

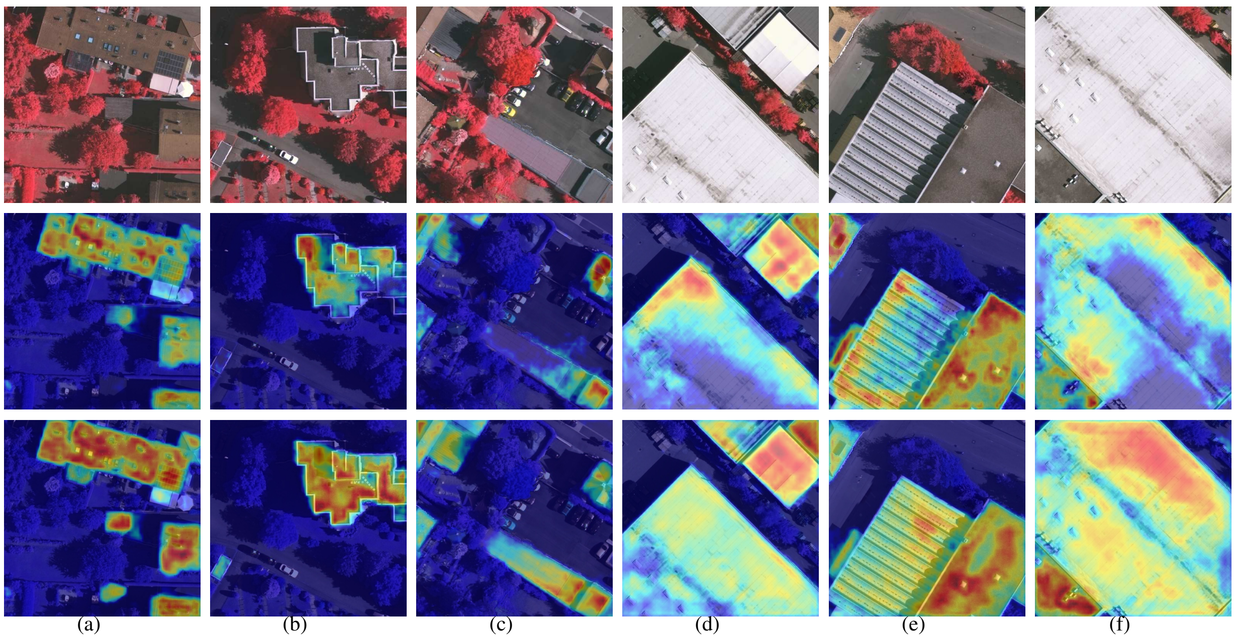

Fig. 2. The class activation mappings (CAM) of buildings with small to large scales, from (a) to (f).

IV. CONCLUSION

In this paper, we propose a light-weight scale-aware module for better recognition of the objects with various scales in high-resolution aerial images. We employ re-sampling scheme to make networks automatically adjust their receptive field to fit objects in different scales. The network embedded with our module not only has scale-aware ability but also is adaptive to different input data, and it is competitive to some heavy-weight networks, which equipped with well-designed feature fusion strategy. Additionally, it can be trained end-to-end and easily embedded in most existing networks. As a result, our method has the potential to address the challenge of multiple-scale objects segmentation in aerial images. For segmentation in aerial images, the networks still need reliable context information to classify the adjacent objects in high similarity. In the future, we will integrate the context-aware module to our works that let the DCNNs better benefit remote sensing.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant 31770768.

References

[1] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, p. 640–651, Apr 2017.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, p. 234–241, 2015.

[3] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," *Lecture Notes in Computer Science*, p. 833–851, 2018.

[4] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[5] G. Li, L. Li, H. Zhu, X. Liu, and L. Jiao, "Adaptive multiscale deep fusion residual network for remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–16, 2019.

[6] X. Lu, H. Sun, and X. Zheng, "A feature aggregation convolutional neural network for remote sensing scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–13, 2019.

[7] Z. Wei, Y. Sun, J. Wang, H. Lai, and S. Liu, "Learning adaptive receptive fields for deep image parsing network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 3947–3955.

[8] R. Zhang, S. Tang, Y. Zhang, J. Li, and S. Yan, "Scale-adaptive convolutions for scene parsing," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 2050–2058.

[9] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *International Conference on Machine Learning*, Jun. 2015, pp. 448–456.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016.

[11] J. Lin, W. Jing, H. Song, and G. Chen, "Esfnet: Efficient network for building extraction from high-resolution aerial images," *IEEE Access*, vol. 7, p. 54285–54294, 2019.

[12] R. Guo, J. Liu, N. Li, S. Liu, F. Chen, B. Cheng, J. Duan, X. Li, and C. Ma, "Pixel-wise classification method for high resolution remote sensing imagery using deep neural networks," *ISPRS International Journal of Geo-Information*, vol. 7, no. 3, p. 110, Mar 2018.