Large Kernel Spatial Pyramid Pooling for Semantic Segmentation

Jiayi Yang[0000−0002−7366−4532], Tianshi Hu[0000−0002−5327−2865], Junli Yang(✉)[0000−0001−8370−7105], Zhaoxing Zhang, and Yue Pan

Beijing University of Posts and Telecommunications
{yangjunli,markyang,ertyoii}@bupt.edu.cn

Abstract. Spatial pyramid pooling is growing to become an important component in the network for semantic segmentation. However, it faces a dilemma of using larger kernels for better global context and computation cost. Recent architectures like ASPP have tried to solve this problem by using atrous convolution to keep large reception field while reducing the cost. However, atrous convolutions bring new problems like the "gridding effect", and the large gap between convolutional points make it hard to extract features of small or narrow objects. Inspired by the idea of stacking small filters to simulate large kernels, we propose a Large Kernel Spatial Pyramid Pooling to address both sufficient receptive field while maintaining efficiency. Our approach is evaluated on PASCAL VOC 2012 dataset and Road Extraction Challenge dataset, and achieved better results than competing architectures.

Keywords: Semantic Segmentation · Spatial Pyramid Pooling.

1 Introduction

Since the design of Spatial Pyramid Pooling (SPP) [4], using parallel convolution to extract features of different sizes has been a trend in computer vision [2] [10]. SPP is a parallel connected convolution module located after the network backbone to extract multi-scale features. Semantic Segmentation has also been using the idea of SPP in many of the modern architectures. PSPNet was one of the first to use them.

SPP is a parallel connected convolution module located after the network backbone to extract multi-scape features. It usually splits the feature map into four parallel channels with different sizes of convolution kernel on each channel. Each parallel convolution outputs feature map of the same size and is concatenated at the end of the SPP module, the overall structure is shown in Fig. 1 As different kernel size results in different sizes of receptive fields.

One of the limitations of SPP is that to extract features of different scales requires different size of large convolutional kernels, which will introduce many parameters to the network . While it may not affect much in tasks like classifications, using large kernel can result in a huge increase of training and inference time in semantic segmentation, due to the pixel-wise segmentation output [5]

and back propagation. This limits the size of kernel used in SPP module and further limits its ability to extract global context.

DeepLab series [1] [2] were introduced to alleviate this problem. In their proposed network, a modified version of Spatial Pyramid Pooling was introduced to solve the problem of computation cost limiting the kernel size used in Spatial Pyramid Pooling. The core innovation of ASPP is that they replaced traditional convolution network with Atrous Convolution [9]. With the same receptive field, ASPP greatly reduced the number of parameters compared to traditional SPP. However, atrous convolution of ASPP brings out another problems.

In this paper, we propose an improved SPP architecture, called Large Kernel Spatial Pyramid Pooling, to deal with the problems mentioned above:

- large convolution kernel in SPP module of introduces significant amount of additional parameters.
- Atrous convolution fails to extract local context information.

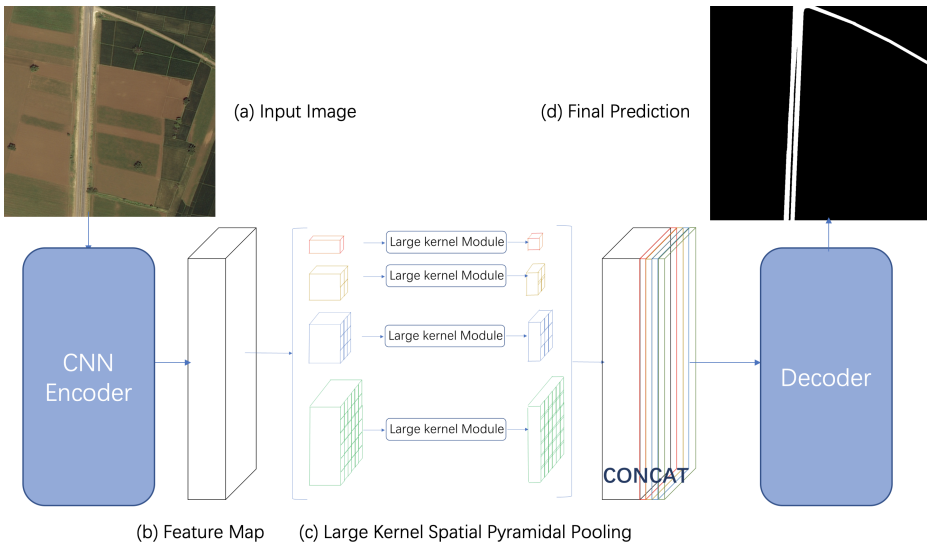

Fig. 1. A network structure with Large Kernel Spatial Pyramid Pooling. It receives feature map extracted from the network backbone. Output of the module fed into the decoder to recover pixel resolution.

To solve these two problems, we describe a novel Large Kernel module inspired by [8] [6]. This new module increases the reception field of the kernel by using a combination of depthwise separable convolution [3] and Global Context Network [6], thereby extracts dense features with large reception field while keeping the number of parameters in an acceptable range. This module can be

replace any other neural network that uses SPP or SPP-like module to achieve better performance.

2 Related Works

In this work, we mainly focus on the Spatial Pyramid Pooling module of the network, and pay less attention on the backbone of the network.

Spatial pyramid pooling: Spatial pyramid pooling layer [4] has been successfully applied to deep convolutional networks to deal with problem of classification and object detection. The SPP layer can pool the features extracted at variable scales and produce fixed-length outputs, which can be used as input of fully-connected layers. What's more, the SPP uses multi-level spatial bins. These advantages of SPP layer help to remove the requirement of fixed-size image input, which means we do not need to crop or wrap the image at the risk of image containing incomplete object or unwanted distortion.

PSPNet: PSPNet [10] works well on scene parsing, which is based on scene segmentation. It is used to predict the label and location for each element. PSPNet make good use of global scene category clues by using global pyramid pooling features. It embeds difficult scenery context features and adopts an effective optimization strategy based on deeply supervised loss. With this improvements PSPNet can do better prediction.

DeepLabv3+: DeepLabv3+ [2] is an effective model in semantic segmentation. It extends the DeepLabv3 model by adding a decoder module to improve the result of semantic segmentation on the boundaries of object. The encoder-decoder structure of DeepLabv3+ can control the resolution of extracted encoder features. With this improvement, DeepLabv3+ can perform better on semantic segmentation along object's boundaries.

Large Kernel Matters: Large kernels play an important role when we perform the classification and localization tasks simultaneously. Large kernel enables the densely connections between feature maps and per-pixel classifiers to make sure the models are invariant to different transformations in classification task.

Rethinking Inception: Inception architecture has much lower computational cost, however, the inception architecture is very complex. To trade off between the computational cost an the complexity, they point out 4 rules: First, we should avoid representational bottlenecks, especially in early stage of the network; Secondly, higher dimensional representations are easier to process locally in the network; Thirdly, we can do spatial aggregation over lower dimensional embeddings; Fourthly, we should balance between the width and depth of the network.

3 Proposed Model

In this section, we start from analyzing spatial pyramid pooling and propose our Large Kernel Spatial Pyramid Pooling (LKSPP).

3.1 Spatial Pyramid Pooling

Spatial Pyramid Pooling is first introduced in [4] to extract multiple features in the task of object detection, and was soon implemented in the task of Semantic Segmentation by PSPNet. In their work, observations were made that traditional FCN often suffer from problems like mismatching relationship, confusion categories and inconspicuous classes. These problems are mainly related to missing of global context information for different sizes of receptive fields.

To alleviate this problem, Pyramid Pooling Module was introduced. Pyramid Pooling Module accepts extracted feature maps from the network backbone and performs convolution on different scales to extract global context information. However, convolution kernels cannot be as large as desired due to the exponentially growing parameters and computation loads. Atrous Spatial Pyramid Pooling was designed to alleviate this problem. Compared with traditional Spatial Pyramid Pooling, Atrous Convolution was introduced to increase the receptive filed while remaining the same computational cost.

3.2 Large Kernel Spatial Pyramid Pooling

To overcome the shortcomings of SPP and ASPP, we introduce Large Kernel Spatial Pyramid Pooling, which proves to find a better balance between computation cost and effectiveness. The structure of Large Kernel Spatial Pyramid Pooling is illustrated in Fig. 2 Each atrous convolution kernel is replaced with the large kernel modulereference Large Kernel Matters. Large Kernel Module was originally inspired by [8] [6], which uses a two channel convolution of $n*1+1*n$ and $1*n+n*1$ followed by batch normalization layer on each channel to avoid the affect of covariate shifting.

Further more, we use depthwise separable convolution instead of traditional convolution to further reduce our computation cost. The computation cost can be calculated with the following equation: The overall structure consists of a network backbone followed by a 4-channel parallel LKSPP module. It collects dense features of multiple sizes and the output feature maps are concat together for the decoder to recover the resolution for prediction, see Fig. 1

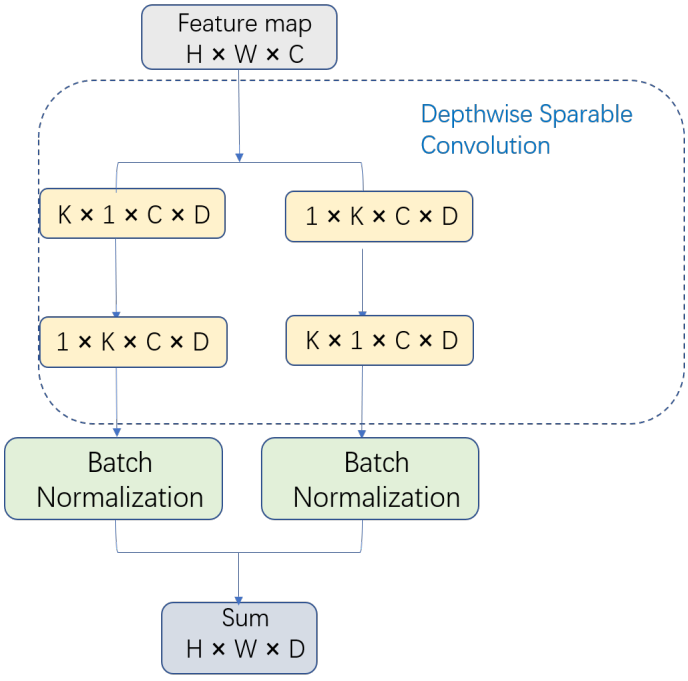

Fig. 2. Structure of Large Kernel Spatial Pyramid Pooling. The convolutional kernels used are depthwise separable convolution followed by batch normalization on each channel. This structure simulates a traditional $K * K$ convolution kernel. The input and output feature maps are C and D respectively.

3.3 Model Size Control

An important feature of LKSPP is that it significantly reduces the parameters needed in the network compared to traditional Spatial Pyramid Pooling.
The parameters existing in a traditional SPP module can be given by the following equation:

$$V_{conv}(K) = K^2 * C * D \tag{1}$$

Where C stands for the number of input feature maps and D for the number of output feature maps. K represents the height and width of the kernel, assuming that they are of the same length.

Parameters can be significantly reduced by using Large Kernel module:

$$V_{lk}(K) = (K * 1 * C * D) * 4 \tag{2}$$

To reduce further reduce parameters, we replace convolutional kernel used in LKSPP with depthwise separable convolution. $K * C$ describes the parameters of the depthwize convolution. Note that in Xception it is described as $K^2 * C$, but since we use a kernel size of $K*1$ or $1*K$, the kernel size is reduced to $K*C$.

The following $C * D$ describes the pointwise convolution with $1 * 1$ convolution kernel. The scale factor 4 represents the two kernels used in the two different channels each in Large Kernel.

$$V_{lk}(K) = (K * C + C * D) * 4 \tag{3}$$

The total equation is given by equation 4, where k_1,k_2,..., k_n represents the kernel size of each parallel channel with a total of n channels. In case of our standard LKSPP, we use a combination of $[k_1, k_2, k_3, k_4] = [1, 4, 8, 16]$.

$$V_{total} = \sum_{k=k_1}^{k_n} V(K) \tag{4}$$

It is not hard to see that when given C and D, parameters traditional convolution grows exponentially with its kernel size K. When we use Large Kernel, The parameters grows linearly with K, significantly reducing parameters when the kernel is very large, Fig. 3 illustrates the growth of parameters with the increase of K. This property is especially useful since we often set K very big in SPP modules.

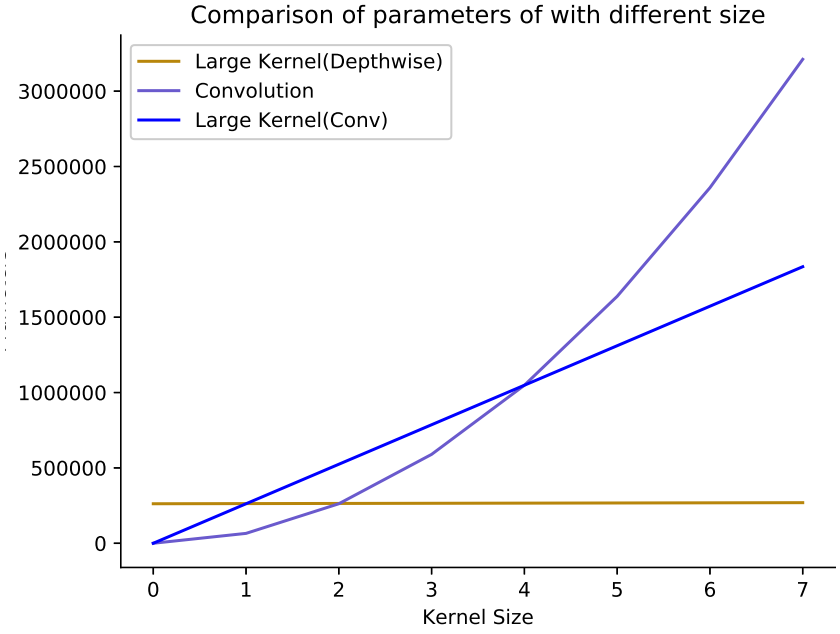

Fig. 3. This figure shows the parameters used in a kernel with different kernel size K. Traditional convolution increases notably after the kernel grows bigger than 2. We assume that the input and output feature map has a depth of 256.

4 Experiment

We firstly evaluate our model, which is pretrained on Pascal VOC 2012 dataset, on Road Extraction Challenge semantic segmentation dataset which contains one foreground road class and one background class. The dataset contains 6226 training, 1243 evaluation and 1101 testing pixel-wise annotated images of size 1024x1024. We train the network on the training set and report the intersection over union (IoU) of the road class.

Then, in order to further prove the effectiveness of our proposed model, we employ another Xception65, which is pretrained on ImageNet-1k dataset, as the network backbone and evaluate this model on augmented [7] Pascal VOC 2012 benchmark, which contains 10582 training, 1449 validation and 1456 testing images of 21 classes. And the performance is evaluated in terms of mean intersection over union (mIoU) across all 21 classes.

4.1 Training setting

The training setting is very similar to [1] [2], but there are some changes. We discuss in details in this subsection.

Learning rate: we use a base learning rate = 0.01 with 'poly' learning rate policy which multiplies the base with $(1 - \frac{iter}{iter_{max}})^{power}$ where power is 0.9 as set by [1] [2].

Batch size and crop size: as constrained by the computation power, we strike a balance between large batch size and more information on each image. And after some experiments, we choose batch size = 3 with crop size = 513 during all the training and testing.

Output stride: recall that output stride is the ratio of input image spatial resolution to final output resolution [1] and [2] shows that the smaller output stride the better result, so we adapt the smallest output stride possible and during out training and testing the output stride = 16.

Data augmentation: during training, we apply HSV and spatial data augmentation as proposed by [HSV transfer] and when evaluating or visualizing, we follow [1] [2] method scaling the input images by 1.75, 2.0 and 2.25 and randomly left-right flipping those images.

Besides, we implement Nesterov momentum optimizer with momentum = 0.9 and weight decay 4e-5. All trainings are implemented on 2 NVIDIA GTX 1080 using tensorflow 1.8.

4.2 Road Extraction Challenge dataset

Baseline model: We use DeepLabv3+ with Xception65 as backbone, which is pretrained on Pascal VOC 2012, to train our baseline model. Specifically, the network has an encoder where the Xception65 backbone resides with an ASPP module whose atrous rate is [6, 12, 18] and a simple yet effective decoder. With all training settings modified, except using data augmentation, as indicated by our training protocol section, we alter the evaluation function from calculating

mean IoU to IoU since in this dataset there is only one road class matters. After training the network for 30000 steps, we observe that the loss converges and the result IoU 57.6% is recorded in the first row of Tab. 1.

Adding new loss function: The second row of Tab. 1 is our experiment with a new loss function inspired by [11]. The performance increase from 57.6% to 58.4%.

Adding data augmentation: The third row of Tab. 1 shows the result of adding both HSV and spatial transformation and test time augmentation. The evaluation result increase another 3% to 61.5%.

Model	IoU
Baseline	57.6%
Baseline+new loss function	58.4%
Baseline+new loss function+data augmentation	61.5%
Large Kernel SPP+new loss function+data augmentation	64.6%

Table 1: Road Extraction Challenge test set result of different models

Ablation study: We have also examined the effect of different atrous rates of ASPP module on the network performance. Specifically, we compare different combinations of training and evaluating atrous rate as shown in Tab. 2

Analysis: Here, as introduced in the previous section, the atrous rate in DeepLabv3+ model represents how much context information can be obtained from the atrous convolution operation. When the training atrous rate is set, we can see from Tab. 2 that the larger evaluation atrous rate the better IoU and when the evaluation atrous rate is set, we can also conclude that the larger training atrous rate the better IoU result. It can be seen that atrous rate has strong relationship with the network performance. Therefore, we come up with using large kernel in the ASPP module.

Large Kernel Spatial Pyramid Pooling: We use the DeepLabv3+ having the best performance (with data augmentation and the new loss function added) as a starting point. By modifying all dilated convolutions in ASPP module to Large Kernel convolution, we achieve a 3% performance gain to an IoU of 64.6%. We also compare the solo effect of using Large Kernel Spatial Pyramid Pooling without data augmentation. The results are recorded in the fouth row of Tab. 1 and we present the inference result of both baseline and LK SPP model in the following Fig. 4.

Model	Train atrous rate	Eval atrous rate	TTA	IoU
DeepLabv3+	[2,4,8]	[3,6,9]		56.7%
DeepLabv3+	[2,4,8]	[6,12,18]		57.3%
DeepLabv3+	[2,4,8]	[24,28,32]		57.9%
DeepLabv3+	[2,4,8]	[24,28,32]	✓	60.4%
DeepLabv3+	[4,8,12]	[3,6,9]		57.0%
DeepLabv3+	[4,8,12]	[6,12,18]		57.8%
DeepLabv3+	[4,8,12]	[24,28,32]		58.1%
DeepLabv3+	[4,8,12]	[24,28,32]	✓	60.6%
DeepLabv3+	[6,12,18]	[3,6,9]		57.3%
DeepLabv3+	[6,12,18]	[6,12,18]		57.6%
DeepLabv3+	[6,12,18]	[24,28,32]		58.7%
DeepLabv3+	[6,12,18]	[24,28,32]	✓	61.7%

Table 2: Effect of different atrous rate on training and evaluation using baseline DeepLabv3+ model

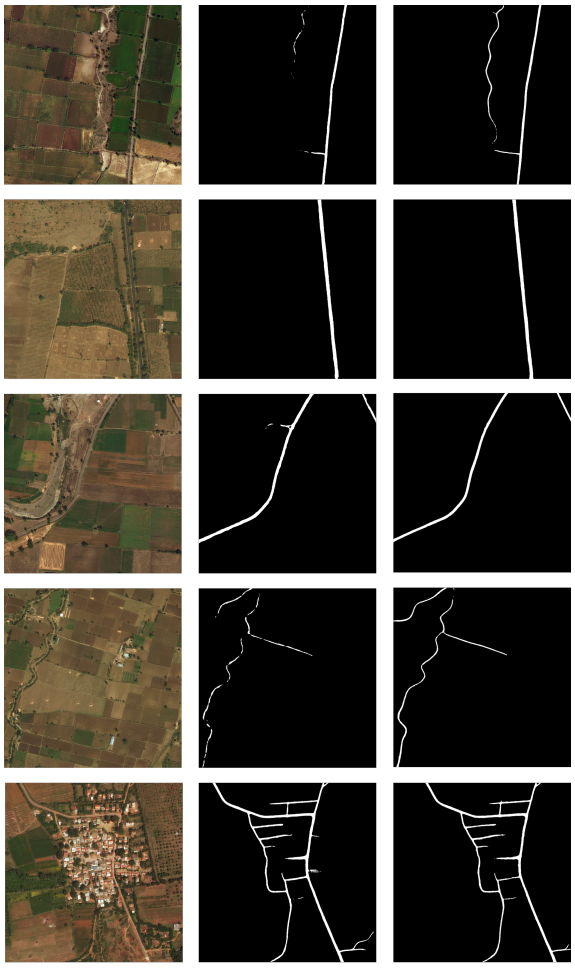

Fig. 4. Visulization result of road extraction. (a) is the orignal images to be infered. (b) is the visualization result of the baseline model and (c) is the inference result of our Large Kernel Spatial Pyramid Pooling model.

4.3 Pascal VOC 2012 dataset

We compare our proposed model with a baseline model on another dataset to prove that the proposed model is truly effective. And due to our limited computation resources, we understand that we cannot reproduce the score that Google's team had made using DeepLabv3+ on Pascal VOC 2012 benchmark. Therefore, rather than fine-tuning the Pascal VOC 2012 pretrained model, which will bring very few improvement, we use ImageNet-1k pretrained model as basic backbone for comparison.

Baseline: We train Google's DeepLabv3+ with the Xception65 backbone, which is now pretrained on ImageNet-1k dataset, on Pascal VOC 2012 dataset. After 30000 steps, we find the loss start to converge and get an mIoU of 38.1%.

Large Kernel Spatial Pyramid Pooling: Next, we implement our Large Kernel mechanism on the original ASPP module and achieve an mIoU of 46.0%, which is significantly higher than the original DeepLabv3+ model. The mean IoU are recorded in Tab. 3

Model	mIoU
Baseline	38.1%
LKSPP	46.0%

Table 3: Result on Pascal VOC 2012 dataset

5 Conclution

In this paper, we propose a novel method for Spatial Pyramid Pooling to alleviate the problem gridding effect brought by Atrous Convolution. We carried out experiments on popular datasets in both natural image segmentation and special tasks like road extraction to evaluate its ability to extract to extract multi-scale features while maintaining efficiency. Results shows that our proposed structures found a good balance between accuracy and efficiency. Our analysis and experiment results shows that Large Kernel Spatial Pyramid Pooling is promising in many tasks containing small objects like road extraction of remote sensing images. We hope our module with implementation details can be adopted to other models and help them to achieve better results.

Acknowledgments. This work is supported by the Research Innovation Fund 201811062 for College Students, and Teaching Reform Project 2019JY-A05 of Beijing University of Posts and Telecommunications.

References

1. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)

2. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 801–818 (2018)
3. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1251–1258 (2017)
4. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence **37**(9), 1904–1916 (2015)
5. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
6. Peng, C., Zhang, X., Yu, G., Luo, G., Sun, J.: Large kernel matters–improve semantic segmentation by global convolutional network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4353–4361 (2017)
7. Sun, T., Chen, Z., Yang, W., Wang, Y.: Stacked u-nets with multi-output for road extraction. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 187–1874. IEEE (2018)
8. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
9. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122 (2015)
10. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2881–2890 (2017)
11. Zhou, L., Zhang, C., Wu, M.: D-linknet: Linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 182–186. IEEE (2018)