Scale-adaptive Convolutions for Scene Parsing *

Rui Zhang[1,3], Sheng Tang[1,3], Yongdong Zhang[1,3], Jintao Li[1,3], Shuicheng Yan[2]
[1] Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, 100190.
[2] Artificial Intelligence Institute, 360 company, Beijing, China, 100025.
[3] University of Chinese Academy of Sciences, Beijing, China, 100039.
zhangrui@ict.ac.cn; ts@ict.ac.cn; jtli@ict.ac.cn; zhyd@ict.ac.cn; yanshuicheng@360.cn.

Abstract

Many existing scene parsing methods adopt Convolutional Neural Networks with fixed-size receptive fields, which frequently result in inconsistent predictions of large objects and invisibility of small objects. To tackle this issue, we propose a scale-adaptive convolution to acquire flexible-size receptive fields during scene parsing. Through adding a new scale regression layer, we can dynamically infer the position-adaptive scale coefficients which are adopted to resize the convolutional patches. Consequently, the receptive fields can be adjusted automatically according to the various sizes of the objects in scene images. Thus, the problems of invisible small objects and inconsistent large-object predictions can be alleviated. Furthermore, our proposed scale-adaptive convolutions are not only differentiable to learn the convolutional parameters and scale coefficients in an end-to-end way, but also of high parallelizability for the convenience of GPU implementation. Additionally, since the new scale regression layers are learned implicitly, any extra training supervision of object sizes is unnecessary. Extensive experiments on Cityscapes and ADE20K datasets well demonstrate the effectiveness of the proposed scale-adaptive convolutions.

1. Introduction

As a significant and challenging task in computer vision, accurate scene parsing is a crucial step towards better scene understanding. The goal of semantic scene parsing is to associate one of the semantic categories to each pixel in a scene image. Recently, approaches based on Convolutional Neural Networks (CNNs) achieve great success in scene parsing. Based on the idea of transfer learning, they employ CNNs [15, 25, 11] pre-trained on large classification datasets [7] to convolutional-deconvolutional frameworks for pixel-level labeling, such as Fully Convolutional Networks (FCNs) based frameworks [21, 3] and Deconvolutional Networks (DeconvNets) based frameworks [23, 1].

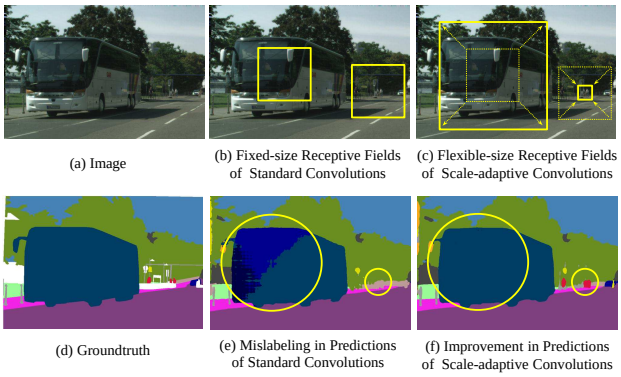

Figure 1. Illustration of the comparison between standard convolutions and scale-adaptive convolutions. The standard convolutions have fixed-size receptive fields (b), which lead to inconsistent predictions of large objects and invisibility of small objects (e). In comparison, the proposed scale-adaptive convolutions learn flexible-size receptive fields (c), which can adaptively expand to cover large objects or shrink to focus on small objects, so as to obtain preferable parsing predictions (f).

However, there is a huge difference between classification and scene parsing, which damages the performance of parsing predictions during transferring models. The CNNs with standard convolutions can only handle a single scale due to the fixed-size receptive fields, as shown in Figure 1(b). This has little influence on classification task, since samples of classification datasets are not only often object-centric but also resized to a uniform size (e.g. 224×224) before being fed into CNNs. Differently, as shown in Figure 1(e), scene images usually contain stuff (e.g. sky, wall) and objects (e.g. people, cars) with various sizes, leading to two critical drawbacks [23]: (1) objects which are enough larger than the receptive fields often have inconsistent parsing predictions, since the receptive fields may cover only small part of the large objects; (2) small objects are often ignored and mislabeled to the background because the re-

*Corresponding author: Sheng Tang.

ceptive fields cover too much background instead of focusing on the small objects.

To conquer these limitations, we propose the scale-adaptive convolutions which are capable of automatically learning flexible-size receptive fields dealing with objects of various sizes. Different from the standard convolutions, the scale-adaptive convolutions need scale coefficient maps, which are learned from additional scale regression layers. Each coefficient in the scale coefficient maps is applied to adjust the size of the associated convolutional patch (sub-region multiplied by the convolutional kernel). Therefore, the size of associated receptive field can be adjusted automatically according to the size of the object. Then feature vectors are sampled from the convolutional patch to perform element-wise multiplication with convolutional kernels.

In the proposed scale-adaptive convolutions, all of the convolutional patches share the same convolutional parameters, but have their own scale coefficients individually. The scale coefficients are position-adaptive and scale-aware, which means feature vectors in different positions have different scale coefficients to acquire flexible-size receptive fields, as shown in Figure 1(c). Specifically, for large objects, the scale coefficients will be larger than 1, so that the receptive fields will expand to cover the entire objects. Otherwise, the scale coefficients smaller than 1 will be learned for small objects so as to shrink the receptive fields to focus on the small objects. Thus, the scale-adaptive convolutions can alleviate the problem of inconsistent predictions and invisible small objects, as illustrated in Figure 1(f). Furthermore, all the processes of scale-adaptive convolutions not only are differentiable, but also can be efficiently implemented in parallel on GPUs. Thus the convolutional parameters and the scale coefficients can be learned in an end-to-end way. In addition, the scale coefficients can be learned automatically and implicitly, so that any extra training supervision of object sizes is unnecessary. Our proposed scale-adaptive convolutions can be regarded as the generalization of standard convolutions. When all of the scale coefficients are set as 1, the scale-adaptive convolutions will be degenerate to standard convolutions.

We employ the scale-adaptive convolutions in the popular FCN with dilated convolutions framework [3] and then perform experiments on two challenging scene parsing benchmarks, including Cityscapes dataset [5] and ADE20K dataset [34]. Experimental results show the effectiveness of our proposed scale-adaptive convolutions.

2. Related Work

Recently, effective and efficient approaches based on CNNs achieve remarkable success in scene parsing and semantic segmentation tasks. Most of them apply CNNs [15, 25, 11] pre-trained on large scale classification datasets [7] to obtain dense parsing predictions. Among them, FCNs based methods [21, 3] perform learning and inference at whole-image-level with efficient dense output and end-to-end training. DeconvNets based methods [23, 1] employ multiple deconvolutional layers and uppooling operators to upsample the low-resolution predictions and capture details gradually. However, the sizes of the receptive fields of these models are fixed, so that they can only capture a uniform scale. During scene parsing, this causes inconsistent labeling of large objects and invisibility of small objects.

In order to address the limitation of fixed-size receptive fields, many approaches based on multi-scale fusion are presented. Almost all of these approaches predict results of multiple preset scales or acquire features from receptive fields of multiple preset sizes, instead of learning the optimal sizes of receptive fields directly. These approaches can be roughly divided into three types: shared-structure, skip-structure and paratactic-structure. Shared-structure based approaches [4] apply a shared model but produce objects with multiple sizes through resizing the input samples. Skip-structure based approaches [10, 17, 22, 8] exploit features of multi-size receptive fields from intermediate layers, since the former layers have the smaller receptive fields. Paratactic-structure based approaches generate multi-stream features with different sizes of receptive fields through spatial pyramid pooling [19, 33] or multi-rate dilation [3]. Multi-scale features or predictions learned from both of the three types of approaches are aggregated with feature concatenation [10, 22, 13], weighted summation [17, 3, 32] or attention models [4]. However, the scales utilized in most of these approaches are usually preset with experts, rather than learned from the samples. A total of 3-5 kinds of scales are employed, which cannot cover all the possible scales for all the objects in scene images. Predictions of multiple scales are needed before fusion, which is computing and memory expensive.

Some approaches try to directly capture and resize objects to match the size of receptive fields. Among them, some approaches [23, 9] utilize off-the-shelf region proposal methods [26, 35] to capture objects. These proposals are acquired from low-level features, so that they may be fragmentary. Some other approaches [6, 16] insert the Region Proposal Networks (RPNs) [24] into CNNs to learn object proposals and resize the feature patches of proposals through the ROI-pooling operator. The bounding-boxes of objects are needed during training RPNs. Thus, these methods are appropriate to handle the instance segmentation task. However, these methods have not been applied in scene parsing, since most of the scene parsing datasets do not provide instance-level annotations.

In this paper, we propose the scale-adaptive convolutions to adaptively acquire flexible-size receptive fields for objects with various sizes in scene images. The appropriate

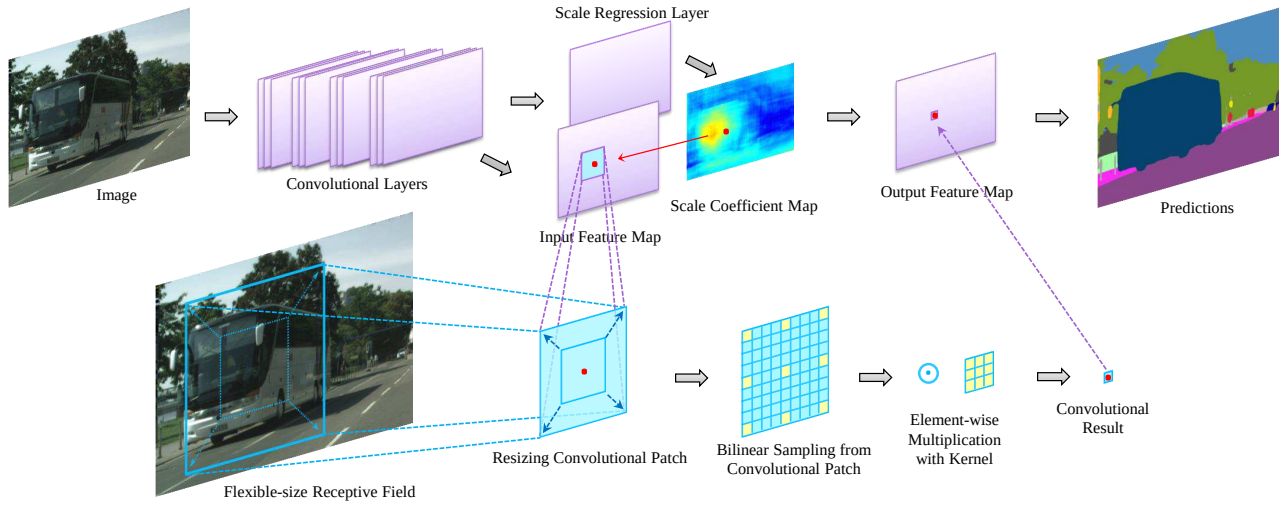

Figure 2. Overview of the scale-adaptive convolutions. We apply it to the last layer of CNNs for example. For each position (the red point as an example), the associated scale coefficient learned from the scale regression layer is employed to resize the associated convolutional patch, so as to obtain a flexible-size receptive field. Then feature vectors are bilinear sampled from the convolutional patch to perform element-wise multiplication with the kernels (adding with the bias is omitted in the figure).

scales of receptive fields can be learned adaptively instead of fusing features of multiple preset scales. The scales are learned implicitly, so that instance-level supervision is unnecessary. Besides, multi-scale fusion can be regarded as data argumentation and model ensemble, so that it can be applied together with the proposed scale-adaptive convolutions to boost the stability and robustness of the predictions and further improve the performance.

3. Scale-adaptive Convolutions

In this section, we introduce the proposed scale-adaptive convolutions, which are presented to automatically adjust the sizes of receptive fields according to the sizes of objects. They are applied to alleviate the problem of inconsistent large-object predictions and invisible small objects, which are caused by fixed-size receptive fields of standard convolutions dealing with objects of various sizes.

3.1. Overview

Figure 2 provides the overview of the proposed scale-adaptive convolutions. There are two major steps. (1) Scale learning: We present to add a new scale regression layer to learn scale coefficients, which indicate the scaling ratios of associated receptive fields. (2) Adaptive convolutions: We apply the scale coefficients to resize the associated convolutional patches. Then we sample feature vectors from the convolutional patches through bilinear interpolation, in order to perform element-wise multiplication with the convolutional kernels and addition with the bias. Through resizing convolutional patches with the learned scale coefficients, scale-adaptive convolutions can acquire flexible-size receptive fields.

The two steps of the scale-adaptive convolutions will be described and formulated in detail as follows.

3.2. Scale learning

Suppose a scale-adaptive convolution takes the input feature maps $A \in \mathbb{R}^{H_A \times W_A \times C_A}$ (with width W_A, height H_A and C_A channels) and outputs feature maps $B \in \mathbb{R}^{H_B \times W_B \times C_B}$ (with width W_B, height H_B and C_B channels). The scale coefficient map $S \in \mathbb{R}^{H_B \times W_B \times 1}$ has the same size with B but has only one channel. It is learned from a scale regression layer. Reasonable initialization of this regression layer is crucial. We adopt an intuitive and appropriate method for initialization, formulated as:

$$\begin{cases} w_0(a) = \varepsilon \\ b_0 = 1 \\ \varepsilon \sim \mathcal{N}(0, \sigma^2), \sigma \ll 1 \end{cases} \tag{1}$$

where w_0 is the initialized convolutional kernels of the regression layer, b_0 is the initialized bias of the regression layer, a is the position in kernels. From this initialization method, kernels are set to small values close to 0, and bias is set to 1. Thus, the generated scale coefficients are almost close to 1, *i.e.* the scale-adaptive convolutions will start from the standard convolutions and gradually learn the appropriate scale coefficients from samples during training. This initialization method is intuitive and stable to avoid ill-conditioned scale coefficients, as analyzed in Section 3.4.

3.3. Adaptive Convolutions

Scale-adaptive convolutions can be regarded as the generalization of standard convolutions. We firstly introduce standard convolutions. Specifically, we discuss convolutions with dilation [30] (also known as Atrous Convolutions [3]), which is widely used in scene parsing and seman-

tic segmentation tasks to enlarge the size of low-resolution predictions before deconvolutions. Then, we describe the formulation of our scale-adaptive convolutions, which can be considered as the generalized convolutions with adaptive dilation parameters, as analyzed in Section 3.4.

For a standard convolution, suppose it has kernel $K \in \mathbb{R}^{C_B \times C_A \times (2k+1) \times (2k+1)}$ and bias $b \in \mathbb{R}^{C_B}$. With input feature maps $A \in \mathbb{R}^{H_A \times W_A \times C_A}$ and output feature maps $B \in \mathbb{R}^{H_B \times W_B \times C_B}$, any feature vector $B^t \in \mathbb{R}^{C_B}$ in B at position t is computed from its associated convolutional patch X^t in A and the kernel K, where $t \in [1, W_B \times H_B] \cap \mathbb{Z}$. X^t is a square with the center of (p^t, q^t) and fixed side of $2kd + 1$, where $d \in \mathbb{Z}^+$ is the dilation parameter, $p^t \in [1, H_A] \cap \mathbb{Z}$ and $q^t \in [1, W_A] \cap \mathbb{Z}$ are coordinates in A. A total of $(2k + 1) \times (2k + 1)$ feature vectors are regularly selected from the convolutional patch X^t to perform element-wise multiplication with the kernel K. The coordinates of these feature vectors are:

$$x_{ij} = p^t + id, y_{ij} = q^t + jd \tag{2}$$

where $i, j \in [-k, k] \cap \mathbb{Z}$. Let $X^t_{ij} = X^t(x_{ij}, y_{ij}) \in \mathbb{R}^{C_A}$ denotes the regularly selected feature vectors. If (x_{ij}, y_{ij}) exceeds the range of A, X^t_{ij} will be set to a zero vector for padding. Let $K^c_{ij} = K(c, i, j) \in \mathbb{R}^{C_A}, c \in [1, C_B] \cap \mathbb{Z}$ is the vector in kernel K to perform element-wise multiplication with X^t_{ij} for output channel c. The process of element-wise multiplication for all of the output channels can be formulated as matrix multiplication with $K_{ij} = K(i, j) \in \mathbb{R}^{C_B \times C_A}$. Thus, the forward propagation is:

$$B^t = \sum_{i,j} K_{ij} X^t_{ij} + b, \tag{3}$$

During backward propagation, given the gradient propagated from B^t, for the standard convolution the gradients are obtained from:

$$g(X^t_{ij}) = (K_{ij})^T g(B^t), \tag{4}$$

$$g(K_{ij}) = g(B^t)(X^t_{ij})^T, \tag{5}$$

$$g(b) = g(B^t), \tag{6}$$

where $g(\cdot)$ denotes gradient function, $(\cdot)^T$ denotes matrix transposition. Note that the gradients of kernel K and bias b should be accumulated with gradients calculated from all of positions in output feature maps B.

For a scale-adaptive convolution, suppose s^t is the scale coefficient associated with B^t in S at position t, the same with the position of B^t in B. Suppose for B^t the associated convolutional patch in A is Y^t, which is also a square with the same center (p^t, q^t) of X^t, but its side will change along with the scale coefficient s^t to $2\lceil kds^t \rceil + 1$. Similarly, a total of $(2k+1) \times (2k+1)$ feature vectors are regularly selected from the convolutional patch Y^t to perform element-wise multiplication with the kernel K, but the coordinates of these feature vectors change to:

$$x'_{ij} = p^t + ids^t, y'_{ij} = q^t + jds^t \tag{7}$$

Note that p^t, q^t, d, i, j are all integers in Equation (7), but the scale coefficient s^t is a real value, so that the coordinates x'_{ij}, y'_{ij} may not be integers. Inspired by the Spatial Transformer Networks [12], we sample these feature vectors through bilinear interpolation. Suppose the convolutional patch after bilinear interpolation is Z^t, formulated as:

$$Z^t_{ij} = \sum_{n,m} Y^t_{nm} \text{max}(0, 1 - |x'_{ij} - m|) \text{max}(0, 1 - |y'_{ij} - n|), \tag{8}$$

where $Y^t_{nm} = Y^t(n, m)$, $n, m \in [-\lceil kds^t \rceil, \lceil kds^t \rceil] \cap \mathbb{Z}$. The forward propagation of convolution is:

$$B^t = \sum_{i,j} K_{ij} Z^t_{ij} + b, \tag{9}$$

During backward propagation, the gradients change to:

$$g(Z^t_{ij}) = (K_{ij})^T g(B^t), \tag{10}$$

$$g(K_{ij}) = g(B^t)(Z^t_{ij})^T, \tag{11}$$

$$g(b) = g(B^t). \tag{12}$$

For bilinear interpolation of Equation (8), the partial derivatives are:

$$\frac{\partial Z^t_{ij}}{\partial Y^t_{nm}} = \text{max}(0, 1 - |x'_{ij} - m|) \text{max}(0, 1 - |y'_{ij} - n|). \tag{13}$$

The partial derivatives of coordinates can also be obtained according to the equations:

$$\begin{cases} \dfrac{\partial Z^t_{ij}}{\partial x'_{ij}} = \displaystyle\sum_{n,m} Y^t_{nm} \text{max}(0, 1 - |y'_{ij} - n|) \delta_x(m, x'_{ij}) \\ \delta_x(m, x'_{ij}) = \begin{cases} 0 & |m - x'_{ij}| \geq 1 \\ 1 & m \geq x'_{ij}, |m - x'_{ij}| < 1 \\ -1 & m < x'_{ij}, |m - x'_{ij}| < 1 \end{cases} \end{cases} \tag{14}$$

and similarly to Equation (14) for $\frac{\partial Z^t_{ij}}{\partial y'_{ij}}$. Since the coordinates x'_{ij} and y'_{ij} rely on the scale coefficient s^t, to obtain the gradient of s^t, the following partial derivatives of coordinates are needed:

$$\frac{\partial x'_{ij}}{\partial s^t} = id, \frac{\partial y'_{ij}}{\partial s^t} = jd \tag{15}$$

Given these partial derivatives, the gradients of scale coefficient map S and input feature maps A can be obtained from chain rule:

$$g(s^t) = \sum_{i,j} \left(\frac{\partial x'_{ij}}{\partial s^t} \frac{\partial Z^t_{ij}}{\partial x'_{ij}} + \frac{\partial y'_{ij}}{\partial s^t} \frac{\partial Z^t_{ij}}{\partial y'_{ij}} \right)^T g(Z^t_{ij}), \tag{16}$$

$$g(Y^t_{nm}) = \sum_{i,j} \frac{\partial Z^t_{ij}}{\partial Y^t_{nm}} g(Z^t_{ij}), \tag{17}$$

This forms a differentiable mechanism so that the parameters of the scale-adaptive convolutions and the scale regression layers can be learned end-to-end from datasets. Besides, the scale coefficients are learned automatically and implicitly, because the gradients of scale coefficients can be calculated from the gradients of the following layer. Thus, we do not need any extra training supervision of object sizes. Technically speaking, the forward computation and

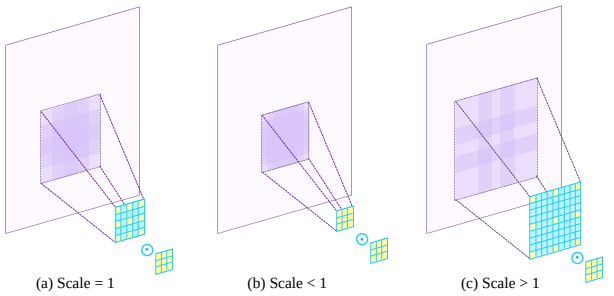

Figure 3. Illustration of flexible-size receptive fields, taking a 3×3 convolution with dilation $= 2$ for example: (a) when scale coefficient is 1, the scale-adaptive convolution degenerates to standard convolution; (b) when scale coefficient is smaller than 1, the convolutional patch shrinks so that the receptive field is zoomed out; (c) when scale coefficient is larger than 1, the convolutional patch expands so that the receptive field is zoomed in.

backward propagation of the scale-adaptive convolutions can be efficiently implemented in parallel on GPUs. The resizing of convolutional patches and the regular selection of feature vectors can be implemented within the "Image to Column" operator.

3.4. Analysis and Discussion

In this section, we give some analysis and discussion to scale-adaptive convolutions. First of all, we explain why our scale-adaptive convolutions can acquire flexible-size receptive fields. Then we analyze the ill-conditioned scale coefficients of the scale-adaptive convolutions and explain the importance of reasonable initialization of the scale regression layers. Finally, we discuss some implementation details and application cases of the scale-adaptive convolutions.

Scale-adaptive convolutions obtain flexible-size receptive fields through adjusting the sizes of convolutional patches with the learned scale coefficients. In standard convolutions, for any output feature vector B^t, its associated convolutional patch X^t has fixed-size of $2kd + 1$ with the constant step of regular selection fixed to d, as formulated as Equation (2). Thus, the size of receptive fields of B^t is fixed. However, in scale-adaptive convolutions, the size of the convolutional patch Y^t associated with B^t will adaptively change to $2\lceil kds^t \rceil + 1$ with scale coefficient s^t varied according to different locations, and the variant step of regular selection will change to ds^t, as formulated in Equation (7). When s^t is an integer, this alteration can also be treated as a standard convolution with the dilation of ds^t, so that it is the generalization of the standard convolution with adaptive dilation parameters controlled by s^t. If $s^t < 1$, the convolutional patch will shrink so that the receptive field will be zoomed out, as shown in Figure 3(b). This shrinkage is helpful to remove the background and focus on small objects. If $s^t > 1$, the convolutional patch will expand so that the receptive field will be zoomed in, as shown in Figure 3(c). This expansion is useful to cover the entire structure aimed at large objects. When $s^t = 1$, the scale-adaptive convolution will degenerate to a standard convolution, as shown in Figure 3(a). Overall, different feature vectors at different locations have their individual scale coefficients. The sizes of convolutional patches will be adaptively resized to acquire receptive fields of appropriate sizes. Namely, the scale coefficients are scale-adaptive to the sizes of objects in scene images.

It is intuitive that the scale coefficients of scale-adaptive convolutions should be non-negative, since the resizing of convolutional patches cannot be realized with negative ratios. Therefore, the non-negative transformation of scale coefficients is essential. There are other two types of ill-conditioned scale coefficients of scale-adaptive convolutions. On the one hand, if the scale coefficient is 0, the convolutional patch will shrink to the center point. On the other hand, if the scale coefficient is extremely large, the convolution patch may extend beyond the range of the entire input feature maps. To overcome these problems, we present a reasonable initialization method for the scale regression layers to avoid the ill-conditioned scale coefficients during training, as illustrated in Equation (1). The scale coefficients will start from 1 and gradually increase or decrease to learn the appropriate scale coefficients during training. Moreover, to guarantee the scales are in the proper condition, we set a lower bound and an upper bound to clip the scales in the experiments.

In our opinion, it is reasonable to apply the scale-adaptive convolutions in the top layers close to classifiers, since the top layers capture semantic-level information, which is sensitive to the size of receptive fields. Besides, top layers have larger dilation parameters, so that the feature vectors from bilinear interpolation will be meaningful especially when the scale coefficients are smaller than 1. The scale-adaptive convolutions can also be applied in several adjacent convolutional layers. Furthermore, adjacent layers can share the same scale coefficient map to reduce the number of parameters of the scale regression layers, as well as alter the sizes of the receptive fields smoothly.

We believe the proposed scale-adaptive convolutions are applicable not only to scene parsing, but also to many other tasks. For example: (1) in tasks related to the objects of various sizes, such as object detection [24], object tracking [2] and image caption [14], the scale-adaptive convolutions can be used to obtain flexible-size receptive fields during feature learning of objects so as to benefit recognition and localization; (2) in tasks dependent on local and detail information, such as fine-grained classification [12] and edge detection [29], the scale-adaptive convolutions can be utilized to automatically shrink the receptive fields so as to focus on the discriminative regions to boost recognition.

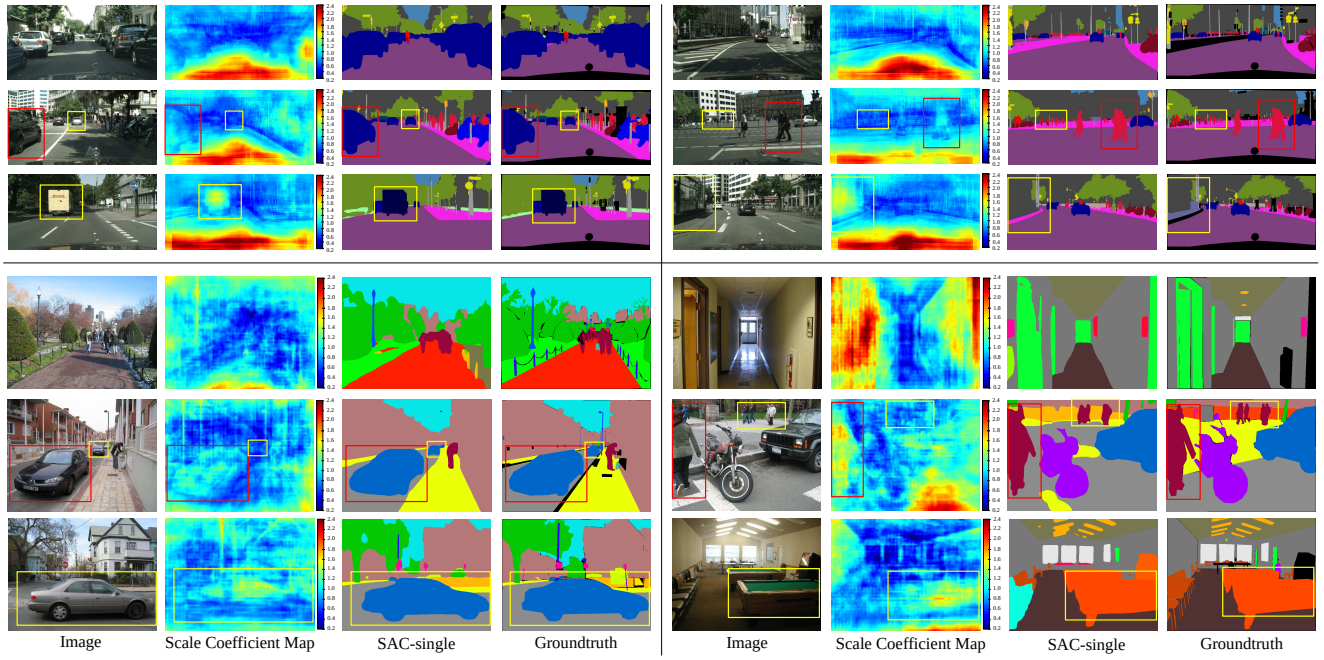

Figure 4. Visualization of scale coefficient maps of SAC-single on Cityscapes validation set (row 1 to 3) and ADE20K validation set (row 4 to 6). Row 1 and 4: the perspective structure of scale coefficient maps. Row 2 and 5: large objects (marked by red boxes) have large scale coefficients to expand the receptive fields, while small objects (marked by yellow boxes) have small scale coefficients to shrink the receptive fields. Row 3 and 6: for large objects (marked by yellow boxes), the scale coefficients associated with the center points are slightly larger, while the scale coefficients associated with the points close to boundaries are slightly smaller.

4. Experiments

In this section, we perform experiments on two challenging scene parsing benchmarks, including Cityscapes dataset [5] and ADE20K dataset [34].

4.1. Experimental Settings

Cityscapes Dataset: The Cityscapes dataset [5] contains 5,000 images, including 2,975 images in training set, 500 images in validation set and 1,525 images in test set. The images in this dataset are collected in street scenes from 50 different cities, with high quality pixel-level annotations of 19 semantic classes and high resolution of 2048×1024. Intersection over Union (IoU) averaged over all the categories is adopted for evaluation.

ADE20K Dataset: The ADE20K dataset [34] is a large-scale dataset recently released by ImageNet Large Scale Visual Recognition Challenge 2016 (ILSVRC2016). This dataset contains 150 semantic classes for scene parsing, with 20,210 images for training, 2,000 images for validation and 3,351 images for testing. Pixel-level annotations are provided for entire images. This dataset is more scene-centric with a diverse range of object categories. The performance is evaluated based on both pixel-wise accuracy and the Intersection over Union (IoU) averaged over all the semantic categories.

Implementation Details: We implement the scale-adaptive convolutions in the widely used FCN with dilated convolutions framework [3]. This network transfers the ResNet101 model [11] pre-trained on ImageNet dataset [7] to the convolutional-deconvolutional structure. The global average pooling layer and the final linear classification layer are replaced with a new 3×3 convolutional layer to generate the M confidence maps (each for one of the M categories) of each spatial location. Dilation convolutions are employed in the last two residual blocks to obtain a higher resolution of direct predictions before deconvolutional layers and maintain more details. Specifically, we remove the last two stride operators in the 4^{th} and 5^{th} residual blocks, while all subsequent convolutional layers in 4^{th} and 5^{th} residual blocks are dilated by a factor of 2 and 4 respectively. Thus the resolution of the direct predictions can be enlarged from 1/32 to 1/8. Then deconvolutional layers [31] are applied to upsample the predictions to the original size. The loss function is the sum of cross-entropy terms for each spatial position in the output, with the unlabeled pixels ignored.

We take the above architecture without scale-adaptive convolutions as the baseline. We propose three schemes to evaluate the performance of the scale-adaptive convolutions. (1) **SAC-single**: The scale-adaptive convolution is employed in the last layer in the framework, *i.e.* the convolutional layer before the softmax layer. A scale regression layer is added by a 3×3 convolutional layer after the 5^{th} residual block to learn the scale coefficients. (2) **SAC-multiple**: The scale-adaptive convolutions are applied in the last layer and all of the 3×3 layers in the 5^{th} residual

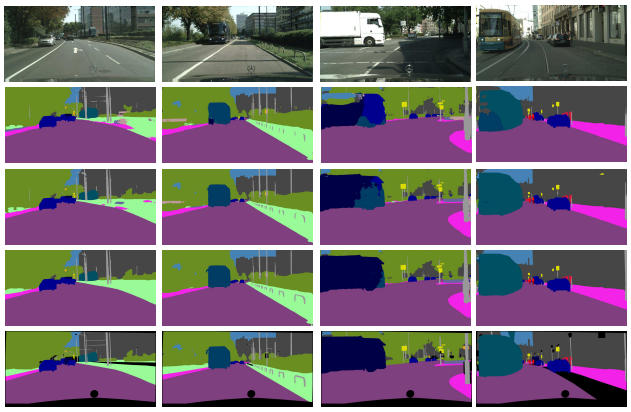

Figure 5. Result illustration of scale adaptive convolutions on Cityscapes validation set. From top to bottom are: image, baseline, SAC-single, SAC-multiple, groundtruth. Consistent predictions for large objects and accurate predictions for small objects can be obtained from scale adaptive convolutions.

block (since it is unnecessary to implement scale-adaptive convolutions in 1×1 layers). Compared with SAC-single, another scale regression layer by a 3×3 convolutional layer is added after the 4^{th} residual block to learn a scale coefficient map, which is shared for all of the 3×3 layers in the 5^{th} residual block. (3) **SAC-single-only** and **SAC-multiple-only**: We implement scale-adaptive convolutions in the same layers with SAC-single and SAC-multiple. Differently, we take the parameters of baseline model as the initialization. During training, we fixed the parameters of residual networks and only update the scale regression layers. This makes the convolutional parameters are the same with the baseline, except the parameters related to the scale regression layers, resulting in the only differences between the sizes of receptive fields. We design these two comparison experiments to demonstrate the advantages of flexible-size receptive fields over fixed-size receptive fields.

During training, standard stochastic gradient descent (SGD) with the mini-batch of 8 samples is adopted. We use the momentum of 0.9 and weight decay of 0.0001, the same with settings during pre-training the classification model. The learning rate is initialized at 0.0005 for 60 epochs and then divided by 10 for another 10 epochs. We randomly crop samples of 500×500 from images during training. Data augmentation through horizontal flip and random resizing between 0.5 and 1.5 are also applied during training. Our experiments are implemented based on MXNet platform, which is efficient concerning GPU memory utilization. All of our networks are trained and tested on four parallel NVIDIA Tesla K40 GPUs.

4.2. Visualization and Analysis

To qualitatively evaluate the performance of scale-adaptive convolutions, we visualize the scale coefficient maps utilized in resizing convolutional patches. We take

Methods	Mean IoU (%)
Baseline(ResNet-101)	74.0
SAC-single-only	74.5
SAC-multiple-only	74.8
SAC-single	75.9
SAC-multiple	76.5
SAC-single + MS	78.2
SAC-multiple + MS	**78.7**

Table 1. Evaluation results of the scale adaptive convolutions on Cityscapes validation set. MS: Multi-scale fusion during testing.

Method	Mean IoU (%)
FCN-8s [21]	65.3
Dilation10 [30]	67.1
DPN [20]	66.8
LRR-4x [8]	69.7
DeepLab [3]	70.4
Adelaide_Context [19]	71.6
RefineNet [18]	73.6
TuSimple [27]	77.6
PSPNet [33]	**78.4**
Model A2,2conv[28]	**78.4**
SAC-multiple(ResNet-101) + MS	78.1

Table 2. Comparison with other state-of-the-art methods on Cityscapes test set. MS: Multi-scale fusion during testing.

the scale coefficient maps from SAC-single for example. As shown in Figure 4 row 1 and row 4, the scale coefficient maps have the perspective structure from the overall view. The scale coefficients closer to the vanishing points are smaller, while the scale coefficients farther away from the vanishing points are larger. This is intuitive due to the similar property of objects. It is clear in the visualization results wherever the vanishing points are. Particularly, objects of the same category with different sizes have different scale coefficients. As shown in Figure 4 row 2 and row 5, large objects have large scale coefficients to expand the receptive fields so as to cover the entire objects, whereas small objects have small scale coefficients to shrink the receptive fields so as to focus on the objects and remove the side-effect of background. Interestingly, for large objects, the scale coefficients associated with the center points are slightly larger, while the scale coefficients associated with the points close to boundaries are slightly smaller, as shown in Figure 4 row 3 and row 6. This is explainable since the smaller receptive fields are helpful to focus on the details and decide the boundaries accurately.

5. Results and Analysis

Results on Cityscapes Dataset: We report the evaluation results of the proposed scale adaptive convolutions on Cityscapes validation set. As concluded from Table 1, we can infer that: (1) Mere size-adaption of receptive fields is effective. As compared with the baseline, the SAC-single-only and SAC-multiple-only bring 0.5% and 0.8% improvement respectively. (2) Joint training both convolutional parameters and scale coefficients gains great im-

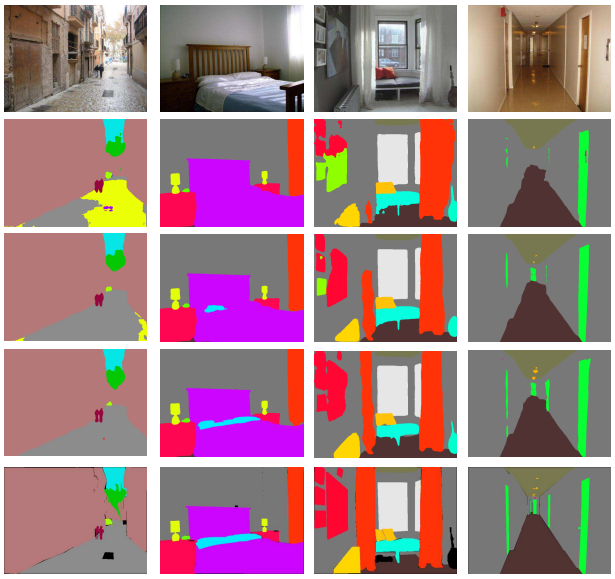

Figure 6. Result illustration of scale adaptive convolutions on ADE20K validation set. From top to bottom are: image, baseline, SAC-single, SAC-multiple, groundtruth. Consistent predictions for large objects and accurate predictions for small objects can be obtained from scale adaptive convolutions.

provement. The SAC-single yields 1.9% improvement over baseline, and outperforms SAC-single-only by 1.4%. This is because the convolutional parameters are jointly learned with dynamical scale coefficients instead of pre-trained with scale coefficients fixed to 1. Similar results are observed for SAC-multiple. (3) Multiple scale-adaptive convolutional layers bring more improvement. The SAC-multiple obtains 0.6% improvement over SAC-single, since it has three more scale-adaptive convolutional layers. In practice, the optimal number of scale-adaptive convolutional layers can be determined by experiment. Figure 5 demonstrates the effectiveness of scale-adaptive convolutions. We also employ the multi-scale fusion during testing, which further improves the performance of SAC-single and SAC-multiple to 78.2% and 78.7%. We believe this is because the multi-scale fusion is an ensemble-based method to acquire more stable predictions, which is complementary with the proposed scale-adaptive convolutions.

We present the comparison results with other state-of-the-art methods on Cityscapes test set in Table 2. The proposed SAC-multiple with multi-scale fusion achieves 78.1% in terms of mean IoU. Note that the performance of our model is slightly lower than the very recent models reported on Arxiv [33, 28], which employ an effective optimization strategy [33] or a wider CNN architecture [28]. However, we believe our proposed method could be complementary to them.

Results on Cityscapes ADE20K: Table 3 report the evaluation results of proposed scale adaptive convolutions on ADE20K validation set, which shows similar conclusion with the results on Cityscapes dataset. The baseline achieves 39.75% in mean IoU and 79.36% pixel accuracy. In terms of mean IoU, the SAC-single-only and SAC-multiple-only whose convolutional parameters are fixed obtain 0.57% and 1.12% improvement from the flexible-size receptive fields. By jointly updating both convolutional parameters and scale coefficients of scale adaptive convolutions, the SAC-single and SAC-multiple yield 1.93% and 2.63% improvement over baseline. Further improvements of 1.97% and 1.92% are gained by employing multi-scale fusion with SAC-single and SAC-multiple. Compared with other state-of-the-art methods, our model based on SAC-multiple achieves 44.30% in mean IoU and 81.86% in pixel-accuracy, which outperforms all the previous methods. Figure 6 demonstrates the effectiveness of scale adaptive convolutions.

Method	Mean IoU(%)	Pixel Acc.(%)
SegNet [1]	21.64	71.00
Cascade-DilatedNet [34]	34.90	74.52
FCN-8s(ResNet101) [21]	32.51	73.67
DeepLab(ResNet101) [3]	36.64	75.73
RefineNet(ResNet101) [18]	40.02	-
PSPNet(ResNet101) [33]	43.29	81.39
Model A2,2conv [28]	43.73	81.17
Baseline(ResNet-101)	39.75	79.36
SAC-single-only	40.32	79.81
SAC-multiple-only	40.87	79.86
SAC-single	41.68	80.64
SAC-multiple	42.38	80.86
SAC-single + MS	43.65	81.33
SAC-multiple + MS	**44.30**	**81.86**

Table 3. Evaluation results of the scale adaptive convolutions on ADE20K validation set, compared with other state-of-the-art methods. MS: Multi-scale fusion during testing.

6. Conclusion

In this paper we propose the scale-adaptive convolutions to obtain flexible-size receptive fields for scene parsing. The scale-adaptive convolutions employ scale coefficient maps to scale the sizes of convolutional patches so as to resize the receptive fields. The scale coefficients can be automatically and implicitly learned without any extra training supervision. The scale-adaptive convolutions can be efficiently implemented on GPUs in parallel. Experiments show that the proposed scale-adaptive convolutions improve the parsing accuracy and achieve state-of-the-art on the challenging ADE20K and Cityscapes datasets.

Acknowledgement

This work was performed when Rui Zhang was an intern at AI Institute, 360 company. This work was supported by National Natural Science Foundation of China (61525206, 61572472), National Key Research and Development Program of China (2016YFB0800403), and Beijing Natural Science Foundation (4152050).

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv:1511.00561*, 2015.

[2] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr. Fully-convolutional siamese networks for object tracking. In *ECCV*, 2016.

[3] L. 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. *CoRR*, abs/1606.00915, 2016.

[4] L. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille. Attention to scale: Scale-aware semantic image segmentation. In *CVPR*, 2016.

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[6] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016.

[7] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[8] G. Ghiasi and C. C. Fowlkes. Laplacian pyramid reconstruction and refinement for semantic segmentation. In *ECCV*, 2016.

[9] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.

[10] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[12] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015.

[13] X. Jin, Y. Chen, J. Feng, Z. Jie, and S. Yan. Multi-path feedback recurrent neural network for scene parsing. In *AAAI*, pages 4096–4102, 2016.

[14] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *CVPR*, 2016.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[16] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. *arXiv preprint arXiv:1611.07709*, 2016.

[17] X. Liang, C. Xu, X. Shen, J. Yang, S. Liu, J. Tang, L. Lin, and S. Yan. Human parsing with contextualized convolutional neural network. In *ICCV*, 2015.

[18] G. Lin, A. Milan, C. Shen, and I. D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *CoRR*, abs/1611.06612, 2016.

[19] G. Lin, C. Shen, I. D. Reid, and A. van den Hengel. Efficient piecewise training of deep structured models for semantic segmentation. In *CVPR*, 2016.

[20] Z. Liu, X. Li, P. Luo, C. C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *ICCV*, 2015.

[21] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[22] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *CVPR*, 2015.

[23] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *ICCV*, 2015.

[24] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[26] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[27] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. W. Cottrell. Understanding convolution for semantic segmentation. *CoRR*, abs/1702.08502, 2017.

[28] Z. Wu, C. Shen, and A. van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *CoRR*, abs/1611.10080, 2016.

[29] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*.

[30] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[31] M. D. Zeiler, G. W. Taylor, and R. Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *ICCV*, 2011.

[32] R. Zhang, S. Tang, M. Lin, J. Li, and S. Yan. Global-residual and local-boundary refinement networks for rectifying scene parsing predictions. In *International Joint Conference on Artificial Intelligence*, 2017.

[33] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. *CoRR*, 2016.

[34] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ADE20K dataset. *CoRR*, abs/1608.05442, 2016.

[35] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.