SUPERPIXEL CONVOLUTION FOR SEGMENTATION

Teppei Suzuki, Shuichi Akizuki, Naoki Kato, Yoshimitsu Aoki

Keio University
Department of Electrical and Electronic Engineering
3-14-1 Hiyoshi, Kohoku, Yokohama, Kanagawa, Japan

ABSTRACT

In this paper, we propose a novel segmentation algorithm based on convolutional neural networks (CNNs) on superpixels. CNNs are powerful methods for several computer vision tasks, but spatial information disappears through the pooling process. Moreover, since pooling compresses different types of pixels into a single value, pooling sometimes negatively affects the results of inference in segmentation task. We use superpixel pooling instead of general pooling to resolve this problem. However, general CNNs can't use superpixel images in which the adjacency relationships between pixels are broken. Therefore, we define CNNs and Dilated Convolution on superpixels. Finally, we show the effectiveness of proposed method on an HKU-IS dataset.

Index Terms— Convolutional Neural Networks, Superpixel, Segmentation, Saliency Object Detection

1. INTRODUCTION

Convolutional Neural Networks (CNNs) have brought about great changes in the computer vision field. The trigger was that AlexNet [1] overwhelmed handcrafted feature-based methods on ILSVRC in 2012. Since then, CNNs have been adapted into several tasks in computer vision field.

CNNs provide some efficient methods for segmentation task. As a representative approach, J.Long *et al.* proposed Fully Convolutional Networks (FCNs) [2]. FCNs achieved state-of-the-art segmentation, but FCNs have the problem that output resolution is less than input resolution. To solve this problem, V.Badrinarayanan *et al.* proposed SegNet [3], which adapted an encoder-decoder model. The encoder-decoder model can output a segmentation mask that has the same resolution as the input. Moreover, SegNet has skip connections, which prevent spatial information from vanishing by pooling. However, since skip connection of SegNet only transports pooling indices, some spatial information is lost. O.Ronneberger *et al.* proposed a more efficient architecture called U-Net [4]. U-Net is also an encoder-decoder model that uses skip connections. A difference for SegNet is that information is transported by the skip connection. The skip connections of U-Net transport the feature maps of the encoder to the decoder. It enables transportation of spatial information before it is lost.

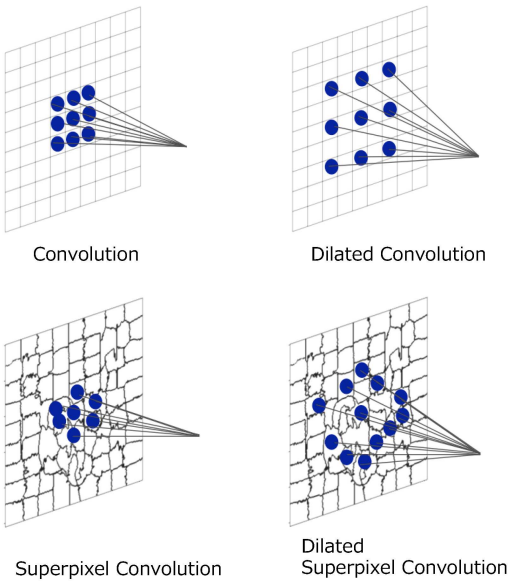

Fig. 1. Convolution and our superpixel convolution.

In segmentation tasks, the problem is that pooling compresses different type of pixels such as foreground and background pixels into a single value. For this reason, segmentation results near the foreground boundary worsen. In this paper, we propose an alternative algorithm based on superpixels to prevent spatial information from vanishing. Our method is inspired by the superpixel pooling Network [5]. The superpixel pooling compresses the feature map based on superpixels. Let $S \in \{s_0, s_1, \ldots, s_R\}$ be superpixels of an input image, where s_r indicates a set of pixels belonging to the r-th superpixel. Let $\mathbf{z} \in \mathbb{R}^{C \times H \times W}$ be the input feature map of superpixel pooling layer. Superpixel average/max pooling is then given by

$$\hat{z}_{c,r} = \frac{1}{n_r} \sum_h \sum_w \mathbb{1}(z_{c,w,h} \in s_r) z_{c,h,w} \tag{1}$$

$$\hat{z}_{c,r} = \max_{z_{c,h,w} \in s_r} z_{c,h,w} \tag{2}$$

where n_r indicates the number of pixels in the r-th superpixel. $\mathbb{1}(z_{c,h,w} \in s_r)$ is an indicator function that is 1 if the r-th superpixel s_r includes the pixel of the feature map $z_{c,h,w}$, and 0 otherwise.

In this paper, we adopt this superpixel pooling into general CNNs to resolve the pooling problem. However, a problem arises here. The dimension of superpixel pooling's output is $\hat{\mathbf{z}} \in \mathbb{R}^{C \times R}$, where C and R are the number of channels and regions. In other words, the output of superpixel pooling is not an image. Moreover, the adjacency relationship between pixels is broken. Therefore, we propose novel convolutional neural networks based on an adjacency matrix which can adapt the output of superpixel pooling.

Our contributions are as follows. (i) We propose superpixel convolution, which can adapt the output of superpixel pooling. Moreover, this proposed algorithm is a generalized convolution on superpixels. (ii) We then expand superpixel convolution into dilated superpixel convolution which is inspired by dilated convolution [6]. Dilated convolution is able to expand the receptive field without extra trainable parameters. Our dilated superpixel convolution algorithm is designed to focus on efficient receptive field expansion. Finally, in Fig 1, we visualize general convolution and our superpixel convolution.

2. PROPOSED METHOD

In this section, we describe our proposed superpixel convolution algorithm. In Section 2.1, we describe convolution on superpixels (Superpixel Convolution) and in Section 2.2 we describe dilated convolution on superpixels (Dilated Superpixel Convolution) as an expansion of the method.

2.1. Superpixel Convolution

We define convolutional neural networks on superpixels. Our approach is similar to graph convolution, but our definition is a generalization of CNNs.

First of all, we consider each superpixel as a node of the graph and let $A_{i,j} \in \{0, 1\}$ be an adjacency matrix. The i-th row of the adjacency matrix indicates the adjacency relationship of the i-node of the graph and $A_{i,j}$ is 1 if the i-th node and j-th node are adjacent, and 0 otherwise. In the other words, the i-th row of the adjacency matrix indicates a pattern of convolution kernel possessed by the i-th node has. So, in the following, we will call $\hat{\mathbf{A}}^k$ the convolution kernel matrix.

$$\hat{\mathbf{A}}^k = (\mathbf{A} + \mathbf{I})^k \tag{3}$$

$\mathbf{I}$ is an identity matrix and $\hat{\mathbf{A}}^k$ indicate the connection between nodes up to distance k. Note that $\mathbf{A}^k$ is calculated as boolean. We can define a convolution kernel by an element-wise product between $\mathbf{A}^k$ and a weight matrix.

We consider trainable parameters $\mathbf{W}$. In deep convolutional neural networks, weight parameters are defined as a

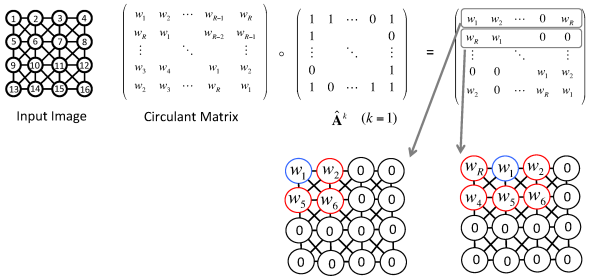

Fig. 2. Position invariance by a circulant matrix. Each node of the input image indicates one pixel. The potion of the weight corresponds to the pixel of interest (blue nodes).

$C' \times C \times h \times w$ tensor, which indicate output channels, input channels, height, and width respectively, and a convolution operation has position invariance. We generalize this mechanism for superpixel convolution. Let $\mathbf{W}_{c',c}$ be a $R \times R$ circulant matrix. Superpixel convolution is then given by

$$\hat{\mathbf{z}}_{c'}^{l+1} = f\left(\sum_{c} (\mathbf{W}_{c',c} \circ \hat{\mathbf{A}}^k)\hat{\mathbf{z}}_c^l\right) \tag{4}$$

where $\circ$ indicates the element-wise product and $\hat{\mathbf{z}}^l \in \mathbb{R}^{C \times R}$ indicates the l-th layer output. f is an activation function such as relu. Since $\mathbf{W}_{c',c}$ is a circulant matrix, the number of trainable parameters is $c' \times c \times R$. Moreover, as shown in Fig 2.1, it gives position invariance to superpixel convolution.

In the above definition, we can calculate convolution on the superpixel. However, superpixel convolution has many trainable parameters. General convolution is in proportion to kernel size, but our superpixel convolution is in proportion to the number of superpixels. Therefore, we modify a weight parameter $\mathbf{W}_{c',c}$ as

$$W_{c',c,r} = \sum_{p} \alpha_{c',c,p} w_{p,r} \tag{5}$$

where p is the number of connected nodes. When calculating superpixel convolution, we transform $\mathbf{W}_{c',c} \in \mathbb{R}^R$ into a circulant matrix $\mathbf{W}_{c',c}^{circ} \in \mathbb{R}^{R \times R}$. By representing a weight parameter $\mathbf{W}$ by an inner product between two trainable parameters, the number of trainable parameters is $c' \times c \times p + p \times R$. If $C \times C' \gg p$ and $R \gg p$, we can reduce the number of trainable parameters. Especially, if $C \times C' \gg R$, we can equate the number of parameters of superpixel convolution with CNNs.

2.2. Dilated Superpixel Convolution

We expand superpixel convolution into dilted superpixel convolution. In this paper, we define dilated superpixel convolution based on the

Table 1. Model parameters. Parameters in parentheses indicate general CNN parameters. SConv indicates superpixel convolution, and DSConv indicates dilated superpixel convolution. Kernel size indicates height×width, Output size indicates channels×height×width. All blocks have shortcut connections, similar to ResNet [7].

Modules	Kernel size	Output size
Conv1_{1,2}		
Conv	3×3	64×256×256
ReLU		64×256×256
Conv2		
Conv	3×3	32×256×256
ReLU		32×256×256
Max Pooling	256 region (16×16)	32×256 (32×16×16)
Block1_{1,2,3}		
Conv	1×1	8×256 (8×16×16)
Tanh		8×256 (8×16×16)
SConv	1 (3×3)	8×256 (8×16×16)
Tanh		8×256 (8×16×16)
Conv	1×1	32×256 (32×16×16)
Tanh		32×256 (32×16×16)
Block2_{1,2,3}		
Conv	1×1	16×256 (16×16×16)
Tanh		16×256 (16×16×16)
DSConv	2 (3×3, dilate 2)	16×256 (16×16×16)
Tanh		16×256 (16×16×16)
Conv	1×1	64×256 (64×16×16)
Tanh		64×256 (64×16×16)
Block3_{1,2,3}		
Conv	1×1	32×256 (32×16×16)
Tanh		32×256 (32×16×16)
DSConv	4 (3×3, dilate 4)	32×256 (32×16×16)
Tanh		32×256 (32×16×16)
Conv	1×1	128×256 (128×16×16)
Tanh		128×256 (128×16×16)
Conv3		
Conv	1×1	32×256 (32×16×16)
ReLU		32×256 (32×16×16)
Up-Sampling		32×256×256
Concatenation	(with Conv2 output)	64×256×256
Conv4		
Conv	1×1	64×256×256
ReLU		64×256×256
Conv5		
Conv	1×1	1×256×256
Sigmoid		1×256×256

idea of effective expansion of the receptive field. We modify a kernel matrix $\hat{\mathbf{A}}^k$ as follows.

$$\hat{\mathbf{A}}^k_{Dilate} = \hat{\mathbf{A}}^k - \hat{\mathbf{A}}^{k-1} + \mathbf{I} \tag{6}$$

$\hat{\mathbf{A}}^k_{Dilate}$ is a convolution kernel matrix for dilated superpixel convolution. We can calculate dilated superpixel convolution to replace the convolution kernel matrix in (4) with $\hat{\mathbf{A}}^k_{Dilate}$.

3. IMPLEMENTATION

To show the effectiveness of our superpixel convolution, we compare our method and a general CNN. All of the models

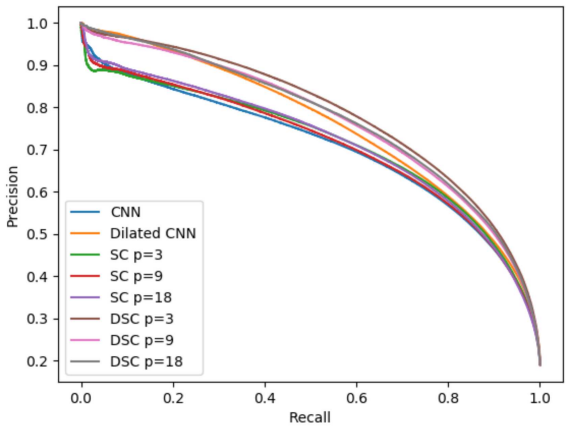

Fig. 3. Precision-recall curve.

in the experiment are the same architecture (Table 1). The superpixel is calculated by SLIC [8]. We optimize models using Adam [9] with binary cross entropy.

$$L_{BCE}(\mathbf{m}, \mathbf{t}) = \frac{1}{H \times W} \sum_h \sum_w (t_{h,w} \log(m_{h,w}) + (1 - t_{h,w}) \log(1 - m_{h,w})) \tag{7}$$

where $m_{h,w} \in (0, 1)$ and $t_{h,w} \in \{0, 1\}$ indicate an estimated mask image and a ground-truth mask image. We train with learning rate 1e-3, β_1 0.5, β_2 0.999, weight decay 5e-4, and batch size 6. All models are trained for 30 epochs, and we evaluate models every epoch against validation data. We use the best model for comparison.

4. EXPERIMENT

We evaluate our model on an HKU-IS [10] dataset. This dataset contains 2500 training data, 500 validation data, and 1447 test data. First, we comparison superpixel convolution (SC) and dilate suerpixel convolution (DSC). Finally, we demonstrate the effectiveness of our proposed method in comparison to CNNs.

We use mean absolute error (MAE) and F-scores for evaluation. These metrics are defined as

$$MAE = \frac{1}{H \times W} \sum_h \sum_w |s_{h,w} - t_{h,w}| \tag{8}$$

$$F_\beta = \frac{(1 + \beta^2) Precision \times Recall}{\beta^2 Precision + Recall} \tag{9}$$

$s_{h,w} \in \{0, 1\}$ indicates the raw output of the model. As suggested by previous works evaluating on HKU-IS datasets, we set β^2 to 0.3 to emphasize the importance of the precision value.

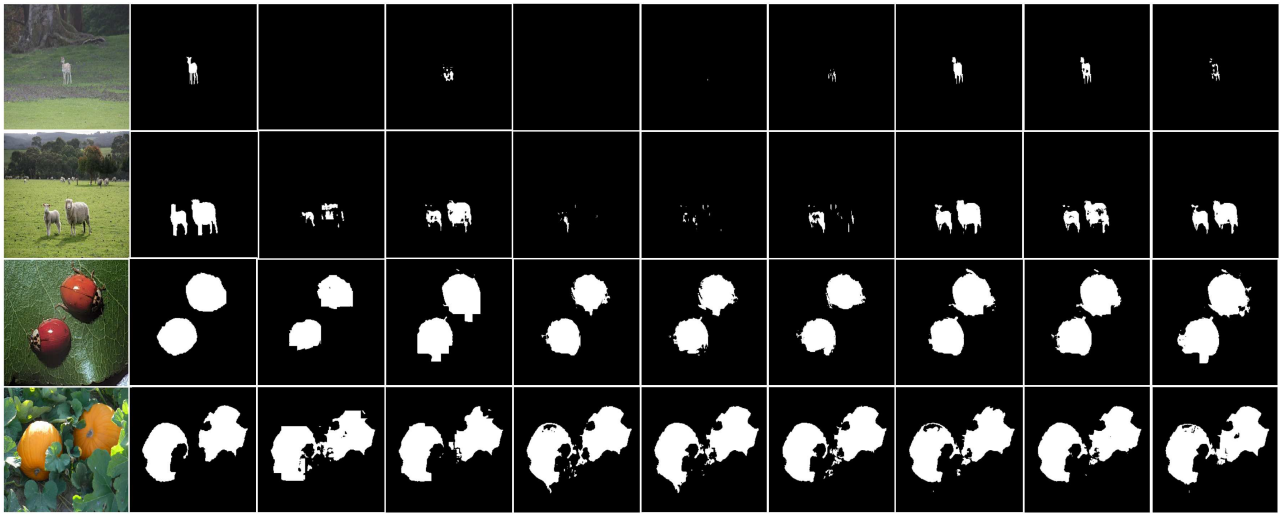

Fig. 4. Visual comparison. From left to right, input image, ground-truth, CNN, dilated CNN, SC ($p = 3, 9, 18$), and DSC ($p = 3, 9, 18$).

Table 2. Comparison between SC and DSC.

Model	p	MAE	F_β-Score
SC no decomp	-	0.143	0.744
SC	3	0.141	0.728
	9	0.140	0.730
	18	0.139	0.733
DSC no decomp	-	0.118	0.794
DSC	3	0.118	0.797
	9	0.121	0.783
	18	0.123	0.783

Table 3. Comparison between SC and CNNs.

Model	MAE	F_β-Score
CNN	0.140	0.719
Dilated CNN	0.125	0.775
SC ($p = 9$)	0.140	0.730
DSC ($p = 9$)	0.121	0.783

4.1. Comparison of SC and DSC

We compare SC and DSC. DSC model has the same parameter as the model in Table 1, and the SC model replaces all dilated superpixel convolution with superpixel convolution with kernel size $k = 1$. We show the results in Table 2. Note that no decomp indicates to calculate without (5).

In SC, the score improved proportionally to the number of weight parameters p. Interestingly, DSC was inversely proportional to p. We have concluded that this may be due to the effect of regularization by p. Especially, a large number of parameters seem to be unnecessary when calculating using the wide range.

4.2. Comparison of SC and CNNs

We compare our SC model and general CNNs. We use two models as baseline models. The first model replacess dilated convolution with a kernel size of $k = 3$ in Table 1. The other model has the same parameters as the model in Table 1. We show the results and the precision-recall curve in Table 3 and Fig 3, respectively. The results show that our SC and DSC models both outperform the baseline models.

Finally, we show examples of outputs in Fig 4. SC and DSC produce more accurate results than the baseline methods. However, the accuracy depends on the superpixel. In this experiment, we used SLIC to calculate the superpixel. Since SLIC calculates the superpixel based on color similarity, it is difficult to distinguish between the foreground and background which have the same color.

5. CONCLUSION

In this paper, we proposed superpixel convolution, a novel convolution framework on superpixels, and Dilated superpixel convolution, which is an extension of superpixel convolution. The main motivation behind Superpixel Convolution was to prevent the loss of important spatial information through segmentation during a pooling operation. Our superpixel convolution method outperformed general CNNs on an HKU-IS dataset, and demonstrating the effectiveness of our approach.

6. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," 2015.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[5] Suha Kwak, Seunghoon Hong, and Bohyung Han, "Weakly supervised semantic segmentation using superpixel pooling network.," in *AAAI*, 2017, pp. 4111–4117.

[6] Fisher Yu and Vladlen Koltun, "Multi-scale context aggregation by dilated convolutions," 2015.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[8] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[9] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," 2014.

[10] Guanbin Li and Yizhou Yu, "Visual saliency based on multiscale deep features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5455–5463.