# RSRN: Rich Side-output Residual Network for Medial Axis Detection

Chang Liu, Wei Ke, Jianbin Jiao, and Qixiang Ye*
University of Chinese Academy of Sciences, Beijing, China
{liuchang615, kewei11}@mails.ucas.ac.cn, {jiaojb, qxye}@ucas.ac.cn

## Abstract

*In this paper, we propose a Rich Side-output Residual Network (RSRN) for medial axis detection for the ICCV 2017 workshop challenge on detecting symmetry in the wild. RSRN uses the rich features of fully convolutional network by hierarchically fusing side-outputs in a deep-to-shallow manner to decrease the residual between the detection result and the ground-truth, which refines the detection result hierarchically. Experimental results show that the proposed RSRN improves the performance compared with baseline on both SKLARGE and BMAX500 datasets.*

## 1. Introduction

Medial axes are pervasive in visual objects such as nature creatures and artificial objects. Symmetric parts and the connections between them, or their interconnection, constitute a powerful part-based decomposition of objects shapes, providing valuable cues for tasks like image segmentation, foreground extraction, object proposal, and text-line detection. The problem of medial axis detection has received increased attention in recent years. Liu et al. have held symmetry detection competitions in CVPR 2011 [8] and CVPR 2013 [6], promoting the symmetry detection in natural images. In ICCV 2017, Liu et al. held a new competition, in which medial axis becomes one task [1].

In this paper, we build a fully convolutional network, named Rich Side-output Residual Network (RSRN), for medial axis detection. RSRN is motivated by the success of image-to-mask deep learning approaches, i.e., Side-output Residual Network (SRN) [5] for object symmetry detection and Richer Convolutional Features (RCF) [7] for edge detection. SRN takes a coarse to fine strategy of stage side-outputs of VGG [10], and RCF takes the rich side-outputs of VGG, to refine the classification map hierarchically .

Experimental results show that the proposed RSRN respectively achieves 6.08% and 0.67% improvement compared with the baseline RCF and SRN on SKLARGE validation dataset. With multi-scale combination, we further obtain 2.35% performance gain.

*Corresponding author

## 2. Methodology

To make the paper self-contained, SRN [5] and RCF [7] are first reviewed. The architecture of Rich Side-output Residual Network (RSRN) is then described.

### 2.1. Review of SRN and RCF

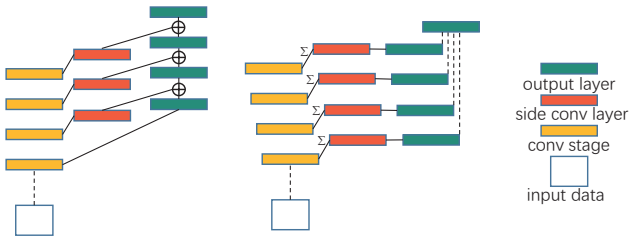


Figure 1. Illustration of (a) SRN and (b) RCF.

Side-outputs of convolutional stages are deeply supervised in both SRN [5] and RCF [7]. The difference between SRN and RCF is the strategy of constructing the side-output.

*SRN:* It takes a coarse-to-fine strategy with Residual Units (RU), Fig. 1(a). For each convolutional stage, the side convolutional layer is added on the last layer of the stage. Hierarchical fusing is then utilized to reduce the residual between the input of RU and the ground-truth. The final output is the output of the last stacked RU.

*RCF:* It uses all the convolutional layers in each stage considering the receptive fields have trivial difference. Instead of adding the side convolutional layer directly, RCF first uses an element-wise summarization of convolutional layers in one stage, then computes side-outputs. The final output is the weighted average of the side-outputs, as shown in Fig. 1(b).

## 2.2. Rich side-output residual network

To take advantages of SRN and RCF, there are two modes to integrate them together. The first one integrates the Residual Unit to the side-output of RCF. The second uses rich side-outputs of each convolutional layer to construct Rich Side-output Residual Network (RSRN). Lastly, a multi-scale combination strategy is used.

### 2.2.1 Base architecture

In the base architecture, Residual Units are stacked on the side-outputs of RCF, Fig. 2. We denote the side-output as $s_i$ and the output of RU as $r_i$. $r_i$ is computed as

$$r_i = w_i^r \otimes (r_{i-1} +_c s_i), \tag{1}$$

where $w_i^r$ is a $1 \times 1$ convolutional kernel to implement the weighted-sum function and $+_c$ denotes the relative operation of concatenating. The deeply supervised output $d_i$ (dsnout) is computed as

$$d_i = Sigmoid(r_i). \tag{2}$$

In this case, pixel-wise feature combination is fulfilled before the side-output. An element-wise sum layer connects the two layers, as

$$s_i = w_i \otimes \sum_j s_{i,j}, \tag{3}$$

where $w_i$ is a $1 \times 1$ convolutional kernel to be used as a pixel classifier, and $s_{i,j}$ is the pixel-wise feature map of the $j-$th convolutional layer of $i-$th stage in VGG [10].

### 2.2.2 Architecture of RSRN

Rich side-output contains more information than stage side-output, as its receptive field size is different for each convolutional layer. It motivates us to fuse the rich side-output with RU to build the Rich Side-output Residual Network (RSRN), Fig. 3.

In RSRN, we use the convolutional response map of VGG as the pixel-based hyper-column feature. Each layer has an output, as

$$s_{i,j} = w_{i,j} \otimes c_{i,j}, \tag{4}$$

where $w_{i,j}$ is a $1 \times 1$ convolutional kernel to be used as a pixel classifier and $c_{i,j}$ is the response map of the $j-$th convolutional layer of $i-$th stage in VGG. All 12 RUs are staked in RSRN, and fused together to get the final output.

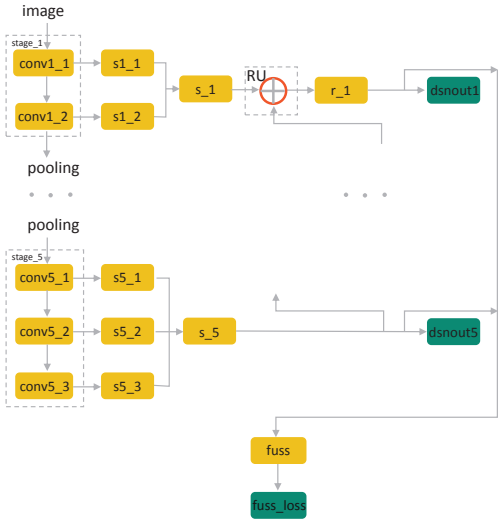


Figure 2. The base architecture integrating RCF and SRN.

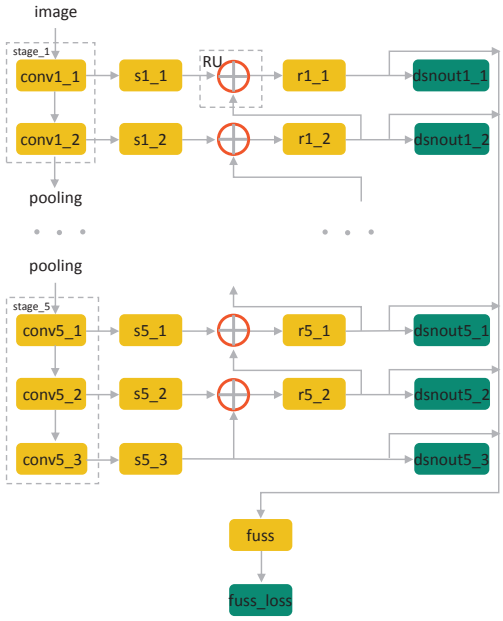


Figure 3. The RSRN architecture.

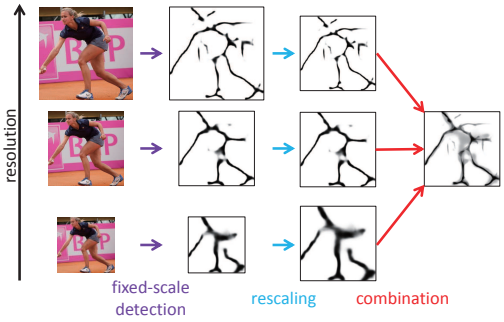


Figure 4. Multi-scale combination.

#### 2.2.3 Multi-scale combination

Multi-scale combination has been successfully used in image segmentation [3] and edge detection [7]. This motivates us to utilize it in medial axis detection, Fig. 4. Specifically, we resize input images at three scales to obtain images at different resolutions. For each scale, RSRN is used to detect the medial axis. And all the outputs are averaged after rescaling. In Fig. 4, it's illustrated that the medial axis of the body is well extracted with high resolution, and the medial axes of arms are well extracted with low resolution.

## 3. Experimental results

In this section, we discuss the implementation of RSRN in detail and compare RSRN with other approaches for medial axis detection. All the experiments are performed with NVIDIA Tesla K80.

### 3.1. Implementation

**Dataset:** The SKLARGE [9] and BMAX500 [11] are used to evaluate the proposed RSRN.

*SKLARGE* contains 746 training images, 245 validation images, and 500 testing images. SKLARGE is focused on the medial axis of foreground objects. The medial axis is the skeleton of the instance segmentation mask of object. It is challenging with a variety of object categories and scale variance.

*BMAX500* is derived from BSDS500 dataset [2], with 200 images for training, 100 images for validating, and 200 images for testing. Each image in BSDS500 is annotated with several volunteers (usually 5-7 per image). Similar with SKLARGE, by extracting segment skeletons on all the annotated segmentation masks of the image, the medial axis ground-truth is obtained. In BMAX500, both foreground and background are considered to extract a medial axis.

**Protocol:** The precision-recall metric with maximum F-measure is used to evaluate the performance of medial axis detection [12], as

$$F = \frac{2PR}{P+R} \tag{5}$$

Given a threshold, the prediction is firstly transferred to a binary map, with which the F-measure is computed. By adjusting the threshold, the maximum F-measure is obtained.

**Data augmentation:** In [5], Ke et al. discuss the data augmentation for medial axis training in deep learning approaches. We follow the data augmentation manner in [5], by rotating each image every 90 degree and flip it with different axes. The multi-scale augmentation is not used as the one-pixel width ground-truth suffers from image resizing.

**Model parameters:** Following the setting of SRN [5], we train RSRN by fine-tuning the pre-trained 16-layer VGG net [10]. The hyper-parameters of RSRN include: mini-batch size (1), learning rate (1e-6 for SKLARGE and 1e-8 for BMAX500), loss-weight for each RU output (1.0), momentum (0.9), initialization of the nested filters (0), weight decay (0.002), and maximum number of training iterations (20,000, about 2 epochs for SKLARGE and 6 epochs for BMAX500). In the testing phase, a standard non-maximal suppression algorithm [4] is applied on the output map to obtain thinned edges and symmetry.

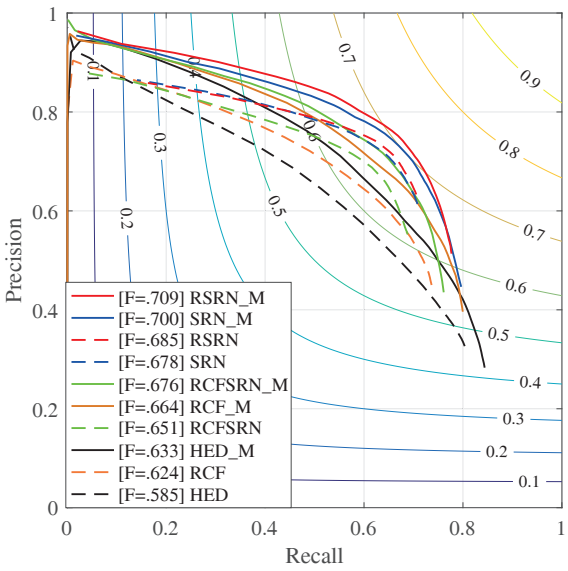


Figure 5. The scatter diagram of precision-recall with the best F-measure.

| methods | F-measure (%) | Runtime (s) |
|---|---|---|
| HED [13] | 58.55 | 0.050 |
| RCF [7] | 62.44 | 0.055 |
| SRN [5] | 67.85 | 0.052 |
| RCFSRN (ours) | 65.09 | 0.059 |
| RSRN (ours) | **68.52** | 0.067 |
| HED_M | 63.32 | 0.202 |
| RCF_M | 66.45 | 0.233 |
| SRN_M | 70.01 | 0.213 |
| RCFSRN_M (ours) | 67.62 | 0.283 |
| RSRN_M (ours) | **70.87** | 0.297 |

Table 1. F-measure on SKLARGE validation dataset.

### 3.2. Performance on SKLARGE

We compare the medial axis detection performance on the SKLARGE validation dataset with HED [13], RCF [7]and SRN [5]. The results are achieved by running the source codes of the methods on a GPU platform. HED is the footstone of all the other methods which are constructed based on HED with additional layers.

The F-measure and the runtime are shown in Table 1 and the Precision-Recall scatter diagram is shown in Fig. 5. HED achieves the F-measure of 58.55%. Base architucture is just a little better than RCF and is not as good as SRN. The initialization of RCF, i.e., the deepest side-output,

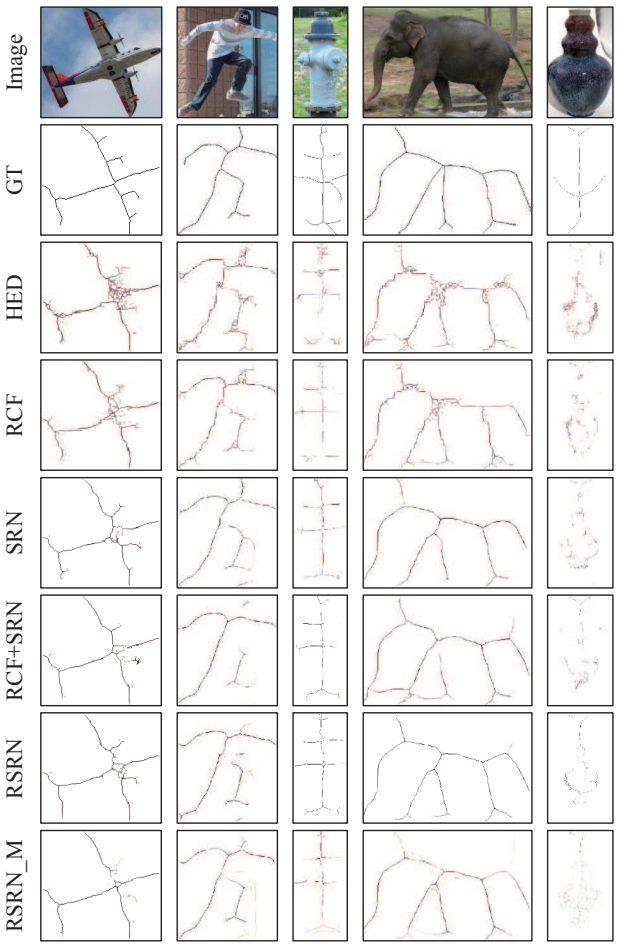


Figure 6. Illustration of medial axis detection results on the SKLARGE validation dataset. First to last rows are image, ground-truth, the results of HED [13], RCF [7], SRN [5], base, RSRN, and RSRN_M, respectively.

is the rough combination of three convolutional layers with different receptive fields in stage-5 of VGG. In RSRN, hierarchical fusing of stage-5 is used to produce the initial results. Our RSRN takes the structure of SRN with the rich side-outputs, and gets the best performance of 68.52%, which improves the baseline RCF and SRN by 6.08% and 0.67%. It is demonstrated that all the convolutional layers contain helpful information, not only the last one in each stage. HED runs fastest with 0.50s per image and all the other methods are a little slower than HED.

From Table 1, it is observed that multi-scale is an efficient way to get better medial axis detection result by pair comparison, at the cost of computational efficiency. The RSRN_M achieves 2.35% performance gain compared with single scale RSRN, and the runtime is almost four times of SRN.

The medial axis detection results are shown in Fig. 6 for qualitative comparison which shows that without RU the HED and RCF produce significant noise.


Figure 7. Illustration of medial axis detection results on the BMAX500 validation dataset. First to last rows are image, ground-truth, the results of SRN [5], and the proposed RSRN, respectively.

| methods | F-measure (%) | Runtime (s) |
|---|---|---|
| SRN | 51.95 | 0.100 |
| SRN_M | 48.99 | 0.397 |
| RSRN | **57.48** | 0.113 |
| RSRN_M | 49.66 | 0.470 |

Table 2. F-measure on the BMAX500 validation dataset.

### 3.3. Performance on BMAX500

The medial axis detection results on the BMAX500 validation dataset are shown in Table 2. We show the results of RSRN and multi-scale RSRN as it achieves the best performance on SKLARGE. From Table 2, RSRN achieves the F-measure of 57.48% and multi-scale RSRN gets the F-measure of 49.66%. In BMAX500, multi-scale is not competitive because BMAX500 considers the medial axis on both foreground and background with complicated images. Fig. 7 shows some detection results.

## 4. Conclusion

In this paper we propose an RSRN approach for medial axis detection. On one hand, we use rich side-outputs of VGG rather than stage side-outputs, which are more informative. On the other hand, we take the advantage of the output residual, reducing the residual between rich side-outputs and the ground-truth, which refines the detection result hierarchically. Experimental results show that RSRN significantly outperforms the baseline on both SKLARGE and BMAX500 datasets.

## Acknowledgement

This work is partially supported by NSFC under Grant 61671427 and Grant 61771447.

## References


[1] Detecting symmetry in the wild. https://sites.google.com/view/symcomp17/challenges/2d-symmetry. 1

[2] P. Arbelaez, M. Maire, C. C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011. 3

[3] P. A. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marqués, and J. Malik. Multiscale combinatorial grouping. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 328–335, 2014. 3

[4] P. Dollár and C. L. Zitnick. Fast edge detection using structured forests. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 37(8):1558–1570, 2015. 3

[5] W. Ke, J. Chen, J. Jiao, G. Zhao, and Q. Ye. SRN: side-output residual network for object symmetry detection in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1068–1076, 2017. 1, 3, 4

[6] J. Liu, G. Slota, G. Zheng, Z. Wu, M. Park, S. Lee, I. Rauschert, and Y. Liu. Symmetry detection from real-world images competition 2013: Summary and results. In *International Conference on Computer Vision and Pattern Recognition Workshops*, pages 200–205, 2013. 1

[7] Y. Liu, X. H. Ming-Ming Cheng, K. Wang, and X. Bai. Richer convolutional features for edge detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3000–3009, 2017. 1, 3, 4

[8] I. Rauschert, K. Brocklehurst, S. Kashyap, J. Liu, and Y. Liu. First symmetry detection competition: Summary and results. *The Pennsylvania State University, PA, Tech. Rep. CSE11-012*, 2011. 1

[9] W. Shen, K. Zhao, Y. Jiang, Y. Wang, X. Bai, and A. L. Yuille. Deepskeleton: Learning multi-task scale-associated deep side outputs for object skeleton extraction in natural images. *arXiv: 1609.03659*, 2016. 3

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 1, 2, 3

[11] S. Tsogkas and S. J. Dickinson. AMAT: medial axis transform for natural images. *arXiv: 1703.08628*, 2017. 3

[12] S. Tsogkas and I. Kokkinos. Learning-based symmetry detection in natural images. In *European Conference on Computer Vision*, pages 41–54, 2012. 3

[13] S. Xie and Z. Tu. Holistically-nested edge detection. In *International Conference on Computer Vision*, pages 1395–1403, 2015. 3, 4