Global-residual and Local-boundary Refinement Networks for Rectifying Scene Parsing Predictions *

Rui Zhang[1,3], **Sheng Tang**[1], **Min Lin**[2], **Jintao Li**[1], **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; linmin@360.cn; jtli@ict.ac.cn; yanshuicheng@360.cn.

Abstract

Most of the existing scene parsing methods suffer from the serious problems of both inconsistent parsing results and object boundary shift. To tackle these issues, we first propose a Global-residual Refinement Network (GRN) through exploiting global contextual information to predict the parsing residuals and iteratively smoothen the inconsistent parsing labels. Furthermore, we propose a Local-boundary Refinement Network (LRN) to learn the position-adaptive propagation coefficients so that local contextual information from neighbors can be optimally captured for refining object boundaries. Finally, we cascade the proposed two refinement networks after a fully residual convolutional neural network within a uniform framework. Extensive experiments on ADE20K and Cityscapes datasets well demonstrate the effectiveness of the two refinement methods for refining scene parsing predictions.

1 Introduction

Semantic scene parsing aims to associate one of the semantic classes to each pixel in a scene image. As an important step towards better scene understanding, accurate scene parsing is a significant and challenging task in computer vision. Recently, the most successful approaches of scene parsing are based on Convolutional Neural Networks (CNNs) [Krizhevsky *et al.*, 2012], especially the variants of Fully Convolution Networks (FCNs) [Long *et al.*, 2015] including [Chen *et al.*, 2015a] [Noh *et al.*, 2015] [Badrinarayanan *et al.*, 2015]. Nevertheless, the predictions of existing methods have two critical drawbacks: (1) the typical inconsistent parsing results on stuff (*e.g.* sky, wall) and large objects as shown in Figure 1 (b); (2) imprecise and discontinuous object boundaries as presented in Figure 1 (e).

To mitigate these issues, in this paper we propose two refinement networks to rectify scene parsing predictions. Rectification is usually regarded as a crucial or even indispensable step in many practical frameworks. For example, in object detection, bounding-box refinement [Gidaris and Komodakis, 2015] is widely used in [He *et al.*, 2016] [Bell *et al.*, 2016] [Shrivastava *et al.*, 2016], bringing significant improvement of bounding-box localization and scoring. Inspired by its success, we design two new refinement networks particularly for rectifying the parsing predictions, from both global and local views respectively. Each of the two networks can be employed after the existing parsing frameworks individually. Moreover, cascading them together for refinement can gain more precise parsing results.

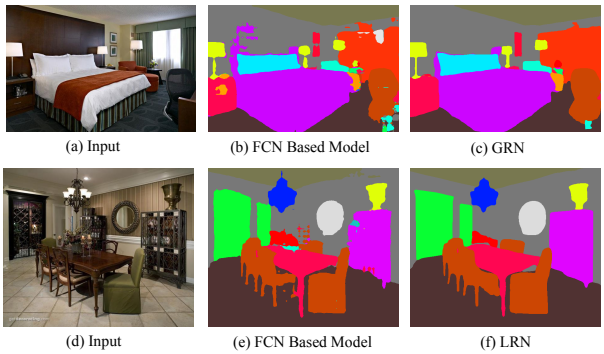

Figure 1: Result of FCN based model (b) has inconsistent labels in wall, curtain and bedside table, which can be refined by the proposed GRN (c). Result of FCN based model (e) has imprecise and discontinuous object boundaries of cabinet, table and chairs, which can be refined by the proposed LRN (f).

Firstly, we consider performing refinement from the global view. Inconsistent parsing results are very common in predictions of existing scene parsing frameworks, as shown in Figure 1 (b). To address this problem, we design the Global-residual Refinement Network (GRN) through exploiting global contextual information and spatial layout relationships during refining. This network takes the original images and the K confidence maps (*i.e.*, the output of the last layer before SoftMax layer, each for one of the K semantic classes) as input. Then outputs the global parsing residual, which will be added to the input confidence maps to obtain the global rectifying results. This network effectively captures global contextual information by iteratively using a deep neural network with large receptive fields. After global refinement by GRN, some mislabeling can be corrected and some inconsis-

*Corresponding author: Sheng Tang. This work was performed when Rui Zhang was an intern at AI Institute, 360 company. This work was supported by National Nature Science Foundation of China (61572472), Beijing Natural Science Foundation (4152050).

tent parsing results can be smoothed, as shown in Figure 1 (c).

Secondly, we also perform refinement from the local view. Most of the existing scene parsing frameworks focus on learning semantic-level features and predict low-resolution parsing results, followed by the deconvolutional layers [Zeiler *et al.*, 2011] to upsample the predictions to original size. This implementation easily leads to losing many details, hence producing imprecise and discontinuous object boundaries, as shown in Figure 1 (e). To tackle this problem, we propose a Local-boundary Refinement Network (LRN) to rectify object boundaries and details by capturing local contextual information. The LRN also takes the original images and the K confidence maps as input, like GRN. It learns $m \times m$ coefficient maps which indicate how each of the $m \times m$ neighbors propagates to the center point. The propagation coefficients are position-aware for each pixel, so as to capture the local contextual information of $m \times m$ neighbors adaptively. LRN works in a similar spirit as bounding-box refinement in object detection task, but it learns local coefficients to refine the object boundaries instead of the bounding-box coordinates. After local refinement by LRN, the object boundaries will be more precise and smooth, as shown in Figure 1 (f).

To verify the effectiveness of our methods, we employ a fully convolutional neural network as the front model and cascade the two refinement networks accordingly. We report experimental results on two scene parsing benchmarks including ADE20K dataset [Zhou *et al.*, 2016] and Cityscapes dataset [Cordts *et al.*, 2016]. Our main contributions can be summarized as follows:

- We propose an iterative Global-residual Refinement Network to predict the global parsing residuals and iteratively boost the parsing results by exploiting global contextual information.
- We design a Local-boundary Refinement Network to refine object boundaries. This network learns position-adaptive local propagation coefficients to exploit local contextual information from neighbors.
- We cascade the proposed two refinement networks after a fully residual convolutional neural network within a uniform framework for scene parsing. We achieve state-of-the-art performance on two challenging scene parsing benchmarks, including ADE20K dataset and Cityscapes dataset.

2 Related Work

Recently, many approaches for scene parsing are based on CNNs [Krizhevsky *et al.*, 2012] and achieve remarkable success. Among them, FCNs [Long *et al.*, 2015] are the most popular framework. Following this framework, lots of approaches are proposed for better parsing results by employing the hole algorithm [Chen *et al.*, 2015a], multiple deconvolution layers and uppooling operators [Noh *et al.*, 2015] [Badrinarayanan *et al.*, 2015], and middle layer features [Hariharan *et al.*, 2015] [Liang *et al.*, 2015] [Mostajabi *et al.*, 2015]. Most of these methods mainly design reasonable networks to predict the parsing results from original images. Their predictions are confronted with the problems of both inconsistent parsing results and object boundary shift.

Different from above approaches, some methods that focus on refinement are proposed to improve the parsing results. Fully connected CRFs [Krähenbühl and Koltun, 2011] [Chen *et al.*, 2015a] is an effective refinement method which precisely localizes segment boundaries. It is based on energy functions to integrate score maps automatically. However, it only considers low-level information to optimize the energy functions. MS-Dilation [Yu and Koltun, 2016] is another refinement method which employs dilated convolutional operators to capture contextual information and aggregate parsing predictions. However, this method only considers global contextual information.

Some recent approaches are also proposed by taking advantage of contextual information and spatial dependencies for scene parsing. Different topological structures of multi-dimensional RNNs are proposed to model the contextual dependencies of image units, such as diagonal structure [Shuai *et al.*, 2016], eight-neighboring structure [Byeon *et al.*, 2015][Liang *et al.*, 2016b] and arbitrary graph topological structure [Liang *et al.*, 2016a]. In order to reduce the length of RNN sequences, most of these methods perform RNN layers on the low-resolution predictions, which may lose lots of details. In addition, graphical models are also widely used in scene parsing to capture the spatial relationships between semantic patches in images [Liu *et al.*, 2015] [Zheng *et al.*, 2015] [Lin *et al.*, 2016] [Vemulapalli *et al.*, 2016]. Graphical models are formulated as special layers and then jointly trained with CNNs. These approaches only employ semantic-level features from CNNs to learn unary and pairwise potential functions and model semantic dependencies between image units.

Unlike these above methods, the proposed two refinement networks exploit both global and local contextual information. Both pixel-level and semantic-level information can be captured from the input which concatenated by the raw RGB values and the confidence maps from the front model. We focus on modeling contextual dependency on high-resolution predictions (the confidence maps have the same size with those of original images) to preserve more details. The two networks are synergistic and complementary to FCNs based methods thus can be employed together.

3 Proposed Refinement Networks

In this section, we first detailedly introduce the proposed two refinement networks that rectify parsing predictions globally and locally. Then we describe how to cascade them together for refinement.

3.1 Global-residual Refinement Network

For stuff and large objects, predictions may be affected by their patterns and textures, and confused with the visually similar categories, leading to misclassification and inconsistent parsing results. To tackle this issue, we propose the Global-residual Refinement Network (GRN) to rectify the confidence maps of the front model globally.

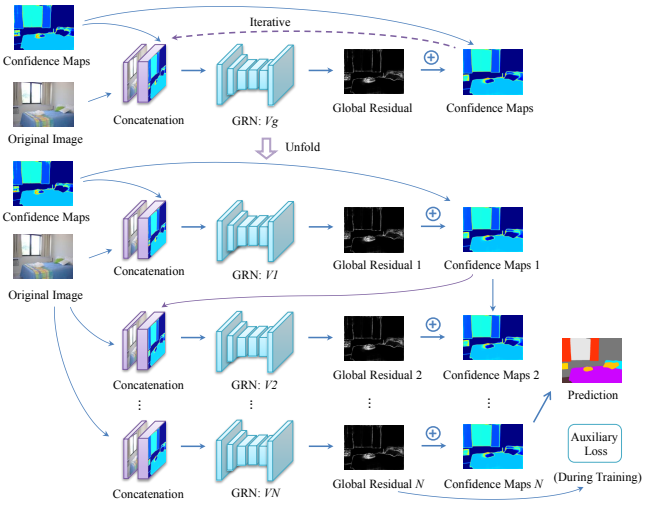

Figure 2: Architecture of the proposed GRN

In order to capture the global contextual information from confidence maps and spatial dependencies from original images together, the GRN takes the confidence maps from the front model and the original images as input. Since the input RGB values of original images are initialized by dividing 255, the input confidence scores should also be normalized to match the same amplitudes with the initialized RGB values, according to the equation:

$$v_i^k = \frac{\exp(u_i^k)}{\sum_{t=1}^{K} \exp(u_i^t)}, k \in 1, 2, \ldots, K, \tag{1}$$

where u_i^k is the confidence value produced by the front model at position i for category k, and K is the number of categories in the dataset. GRN outputs global parsing residuals, which will be added to the confidence maps of the front model to obtain the global refined results.

The GRN exploits deep CNNs to incorporate global contextual information. During the forward propagation procedure, each position is influenced by the hidden cells in the previous layer at neighboring positions, so that the receptive fields expand with the increasing depth of CNNs, which can be formulated as:

$$T_l = [T_{l-1} + (k_l - 1)] \times s_l, l = 1, 2, \ldots, L, \tag{2}$$

where T_l is the size of receptive fields of layer l, k_l is the kernel size of layer l, s_l is the stride of layer l. In the beginning, $T_0 = 1$. With a deep CNN with many 3×3 convolutional layers and several downsampling with stride= 2, GRN can obtain very large receptive fields and capture global contextual information. In this work, we implement GRN in the same architecture with the front model (described in Section 3.3). Thus, the parameters of pre-trained classification model can be adopted for initialization. Moreover, it is worth noting that the global refined results can be concatenated with the original images and fed into GRN again, thereby forming the iterative refinement. The receptive fields will linearly expand with the number of iteration increases, formulated as:

$$T^n = (T_L - 1) \times n + 1, n = 1, 2, \ldots, N, \tag{3}$$

where T^n is the size of receptive fields after n iteration. As shown in Figure 2, suppose for N iterations, the GRN model V_g of N iterations can be unfolded to N models $V_1, V_2, \cdots, V_N$, which share the same parameters. With the iterative processing, the receptive fields can expand quickly to cover the whole image and incorporate global contextual information. Different from the iterative methods [Pinheiro and Collobert, 2014] [Li *et al.*, 2016], GRN focuses on how to obtain large receptive fields to capture global contextual information for refinement. Consequently, GRN generates parsing residuals instead of direct results.

Specifically, GRN is initialized with the parameters from the pre-trained classification model. Particularly, the pre-trained model only takes original images as input. Thus, the parameters associated with confidence maps in the first convolution layer are initialized randomly. During training stage, the loss function is the cross-entropy terms for the summation of the front confidences and global residuals. However, most of the loss values are often small since the parsing results from the front model are good enough to approximate the groundtruth, which inevitably causes small gradients for updating GRN. Therefore, we add an auxiliary loss function of the cross-entropy terms for global residuals, in order to obtain larger gradients and speed up convergence. This auxiliary loss branch will be ignored during test stage.

3.2 Local-boundary Refinement Network

Most existing scene parsing approaches based on CNNs focus on learning semantic-level information while ignoring low-level information, resulting in discontinuous and imprecise parsing boundaries. To solve this problem, we propose a Local-boundary Refinement Network (LRN) to refine the parsing predictions locally and adaptively.

Figure 3 shows the details of LRN. It concatenates the confidence maps from the front model and original images as input, in order to take advantage of local contextual information of confidence maps and the low-level information of original images simultaneously. The confidence maps should also be initialized as Equation (1). This network outputs the normalized $m \times m$ local propagation coefficient maps for all the positions, formulated as:

$$w_i^p = \frac{\exp(h_i^p)}{\sum_{t=1}^{m \times m} \exp(h_i^t)}, p \in 1, 2, \ldots, m \times m, \tag{4}$$

where h_i^p is the confidence value learned by LRN at position i for its neighbor p, and $m \times m$ is the size of propagation neighbors. The propagation coefficient vector at position i will flat to a square at first, then multiply by the confidence maps of its $m \times m$ neighbors, and finally aggregate to the center point to generate the refinement results, denoted as:

$$\mathbf{g}_i = \sum_{p=1}^{m \times m} w_i^p \cdot \mathbf{f}_i^p, \tag{5}$$

where $\mathbf{f}_i^p$ is the confidence vector of the neighbor p of position i from the front model, and $\mathbf{g}_i$ is the refined vector of position i. Note that w_i^p is shared across all the channels of $\mathbf{f}_i^p$. Most importantly, since the propagation coefficients

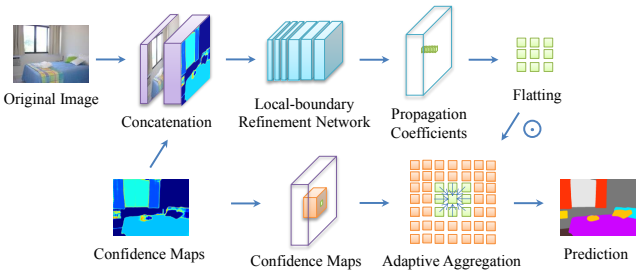

Figure 3: Architecture of the proposed LRN

Layer	Channel	Kernel size	Bias size
1	256	$(K+3) \times 256 \times 3 \times 3$	256
2	256	$256 \times 256 \times 3 \times 3$	256
3	256	$256 \times 256 \times 3 \times 3$	256
4	512	$256 \times 512 \times 3 \times 3$	512
5	512	$512 \times 512 \times 3 \times 3$	512
6	512	$512 \times 512 \times 3 \times 3$	512
7	$m \times m$	$512 \times (m \times m) \times 3 \times 3$	$m \times m$

Table 1: The detailed configuration of LRN. K is the number of semantic classes; $m \times m$ is the size of propagation neighbors.

are automatically learned from LRN and each position has its own refinement coefficients, the propagation coefficients are position-adaptive to capture the local contextual information optimally.

In order to learn the unknown ideal propagation coefficients, we propose an implicit learning method through multiplying the propagation coefficients by confidence maps from the previous model so that we can end-to-end learn the coefficients implicitly with the loss between the products and parsing groundtruth. Therefore, we can avoid explicit supervised learning of the propagation coefficients whose groundtruth cannot be easily acquired. As for implementation, we design a small network including 7 convolutional layers with 3×3 kernels for LRN, since large reception fields are unnecessary for LRN to capture local contextual information. The detailed configuration of LRN is designed as shown in Table 1. We adopt batch normalization (BN) [Ioffe and Szegedy, 2015] right after each convolution and before activation. Pooling and large stride are not used in LRN in order to keep the same resolution between input and output. As the number of layers increases, the receptive field expands and the contextual information increases, which requires more channels for storage in the following layers.

In particular, it is crucial to train a network with appropriate initialization. According to the purpose of LRN, the confidence maps before and after local refinement should be similar. Thus, we propose an intuitive and reasonable method for initialization, formulated as:

$$\begin{cases} k_l(a,c) = \varepsilon, \\ b_l(c) = \begin{cases} 1 & l = L, c = (m \times m + 1)/2 \\ 0 & \text{others} \end{cases} \\ (l = 1, 2, \ldots, L) \end{cases}, \qquad (6)$$

where L is the number of layers in LRN, k_l is the initialized convolutional kernels of layer l, b_l is the initialized bias of layer l, c is the channel of a layer, a is the position in kernels, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ and $\sigma \ll 1$. In this initialization method, kernels are set to small values while all the biases in the former $L-1$ layers are set to 0. In the last layer, biases are set to 0 except the value at position $(m \times m + 1)/2$ (*i.e.* the center position in the neighborhood) is set to 1. By this initialization, the confidence at the center position has a large influence while the other neighbors have little influence during propagation, thus the confidence maps after refinement can approximate identity mapping results. Consequently, the local contextual information will be captured from neighbors during back-propagation learning.

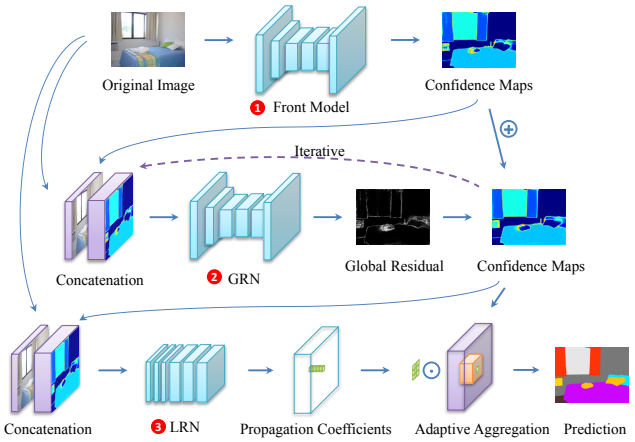

Figure 4: The cascaded architecture: (1) fully residual convolutional neural network as the front model; (2) iterative Global-residual Refinement Network to rectify globally; (3) Local-boundary Refinement Network to rectify locally.

3.3 Cascaded Architecture

As shown in Figure 4, we cascade the GRN and LRN following a front model. The whole pipeline consists of three indispensable stages. First, we implement the widely used fully residual convolutional neural network as the front model, which adapts the ResNet [He *et al.*, 2016] to the convolution-deconvolution framework [Long *et al.*, 2015]. Second, we utilize the GRN to improve the confidence maps from the front model globally. Finally, we perform the LRN to rectify the object boundaries locally. We adopt the cascaded pipeline in order to rectify the predictions step by step, from global to local. If the GRN is applied after LRN, the boundary may be shifted and over-smoothed during global refinement, damaging the improvement of LRN. Thus we implement LRN after GRN to gain more precise boundaries. Additionally, since the GRN and LRN are complementary, cascading them together will collaboratively refine the predictions, outperforming the parallel pipeline which combines the two refinement results by average fusion.

In this architecture, the front model employs a fully convolutional residual network following the design of the FCN framework [Long *et al.*, 2015] but with separate implementations. The convolutional parameters of residual models [He *et al.*, 2016] pre-trained on large scale classification datasets [Deng *et al.*, 2009] are utilized to obtain the low-resolution predictions, followed by the deconvolution layers [Zeiler *et al.*, 2011] to upsample the predictions to the original size. Note that there is a global average pooling of 7×7 before

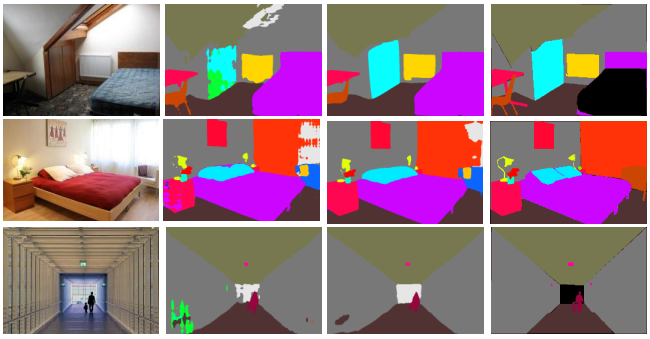

Figure 5: Results of GRN on ADE20K validation set

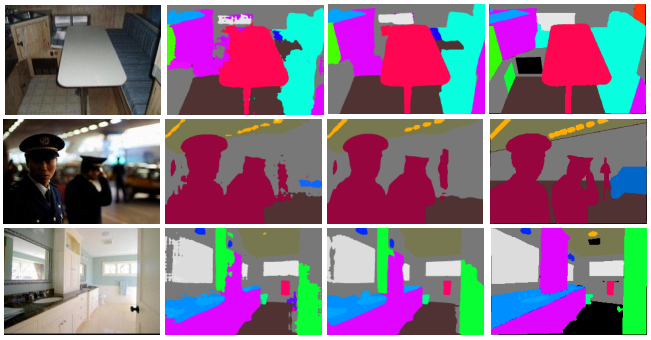

Figure 6: Results of LRN on ADE20K validation set

Method	ADE20K	Cityscapes
Front	38.45%	72.93%
Front+GRN	40.02%	74.65%
Front+LRN	39.79%	74.36%
Front+GRN+LRN	41.12%	75.63%
Front+GRN+LRN+MS	42.60%	77.16%
Front+FC-CRF[Chen *et al.*, 2015a]	38.78%	73.47%
Front+MSD[Yu and Koltun, 2016]	39.43%	73.96%

Table 2: Mean IOU of the two refinement networks and other popular refinement methods on ADE20K and Cityscapes validation set. Front: front model; MS: multi-scale fusion; FC-CRF: Fully connected CRF; MSD: MS-Dilation.

the fully connected layer, which smoothens the features and causes loss of many details. To keep more details, we replace this global average pooling layer and the fully connected layer with a 3×3 convolutional layer by dilation of 3 to maintain the same size of field-of-view and reduce the complexity of this layer, similarly with [Chen *et al.*, 2015a]. Moreover, there are 5 downsampling operators in the original residual network, so that the direct predictions before deconvolutional layers have a low resolution of 1/32, missing lots of details. We remove the last two stride operators and employ the hole algorithm [Chen *et al.*, 2015a] in the 4^{th} and 5^{th} residual blocks, so that we can obtain a higher resolution of 1/8 and maintain more details.

4 Experiments

4.1 Experimental Settings

We evaluate the proposed two refinement networks on two challenging scene parsing datasets, *i.e.* ADE20K dataset [Zhou *et al.*, 2016] and Cityscapes dataset [Cordts *et al.*, 2016].

ADE20K Dataset: The ADE20K dataset [Zhou *et al.*, 2016] is a new large-scale dataset released by ImageNet Large Scale Visual Recognition Challenge 2016 (ILSVRC2016)[1]. This dataset contains 150 semantic classes for scene parsing, 20,210 images for training, 2,000 images for validation and 3,351 images for testing. Pixel-level annotations are provided for whole images. The performance of the proposed two refinement networks is evaluated based on both pixel-wise accuracy and the Intersection over Union (IoU) averaged over all the semantic categories.

[1]http://image-net.org/challenges/LSVRC/2016/index

Method	Mean IoU
SegNet [Badrinarayanan *et al.*, 2015]	21.64%
Cascade-SegNet [Zhou *et al.*, 2016]	27.51%
Cascade-DilatedNet [Zhou *et al.*, 2016]	34.90%
FCN-8s(VGG-Net) [Long *et al.*, 2015]	29.30%
FCN-8s(ResNet) [Long *et al.*, 2015]	32.51%
DeepLab(VGG-Net) [Chen *et al.*, 2015a]	32.31%
DeepLab(ResNet) [Chen *et al.*, 2015a]	36.64%
MPF-RNN [Jin *et al.*, 2016]	34.63%
Ours (single model)	42.60%
Ours (ensemble 3 models)	44.54%

Table 3: Comparison with other state-of-the-art methods on ADE20K validation set

Cityscape Dataset: The Cityscapes dataset [Cordts *et al.*, 2016] contains 5,000 images collected in street scenes from 50 different cities, with high quality pixel-level annotations of 19 semantic classes. There are 2,979 images in training set, 500 images in validation set and 1,525 images in test set. The images in this dataset have a high resolution of 2048×1024. Intersection over Union (IoU) averaged over all the categories is adopted for evaluation. We didn't use coarse data in our experiments.

Implementation Details: The front model and the iterative GRN has the same architecture described in Section 3.3 (based on the ResNet-101 structure [He *et al.*, 2016]), while the LRN utilizes the architecture shown in Table 1. Both the front model and GRN are initialized with the parameters pre-trained on the ImageNet classification dataset [Deng *et al.*, 2009]. We replace the 1000-way ImageNet classifier in the last layer with a classifier that has the same number of semantic classes of the scene parsing datasets. The LRN is initialized with the scheme illustrated in Section 3.2. We set $m = 7$ as the size of refinement neighbors for LRN. We decouple the three components and optimize them one-by-one. The loss function is the sum of cross-entropy terms for each spatial position in the output, with the unlabeled pixels ignored. Standard stochastic gradient descent (SGD) with mini-batch of 4 samples is adopted for training. We use the momentum of 0.9 and weight decay of 0.0001, the same with settings during pre-training the classification model [He *et al.*, 2016]. For training the front model, the learning rate is initialized at 0.001 for 30 epochs and then divided by 10 for another 10 epochs. After that, GRN is trained for 20 epochs in total, including 10 epochs with the learning rate of 0.001 and 10

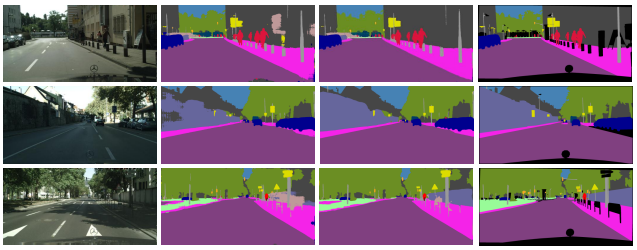

Figure 7: Results of GRN on Cityscapes validation set

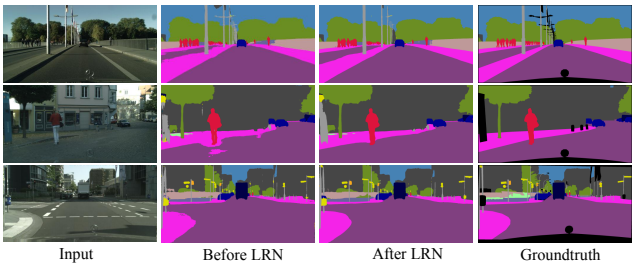

Figure 8: Results of LRN on Cityscapes validation set

epochs with the learning rate of 0.0001. Finally, the learning rate of 0.0001 is implemented to train LRN for 20 epochs. During training, we randomly crop samples of only 500×500 for ADE20K dataset and 700×700 for Cityscapes dataset due to the limitation of GPU memory. At test stage, the whole images are fed into the model to maintain contextual information. Data augmentation through horizontal flip and multi-scale input are also applied in both training and test stages. In particular, 5 scales ($\{0.5, 0.75, 1.0, 1.25, 1.5\}$) are employed for ADE20K dataset and 4 scales ($\{0.5, 0.75, 1.0, 1.25\}$) are employed for Cityscapes dataset.

Our experiments are implemented based on MXNet platform [Chen *et al.*, 2015b], which is efficient concerning GPU memory utilization. All of our networks are trained and tested on four parallel NVIDIA Tesla K40 GPUs.

4.2 Results and Analysis

Results on ADE20K Dataset: We report the evaluation results of the two refinement networks on ADE20K validation set in Table 2. In terms of mean IoU, the front model (based on ResNet-101) achieves 38.45%. Employing GRN individually with three iterations brings 1.57% improvement, while employing LRN individually with the proposed initialization scheme gives 1.34% gain. Besides, employing the GRN and LRN cascaded architecture yields 2.67% improvement. We also implement other two popular refinement methods: fully connected CRF[Chen *et al.*, 2015a] and MS-Dilation [Yu and Koltun, 2016]. Fully connected CRF only brings 0.33% improvement, while the MS-Dilation yields 0.98% gain, slightly poorer than the proposed two refinement networks. Finally, multi-scale fusion during testing improves the performance to 42.60%. Table 3 shows the comparison results with other state-of-the-art methods. Note that the ADE20K dataset is released very recently, thus many methods have not reported their results on this dataset. Compared with other methods, our single cascaded refinement model (with the front model based on ResNet-101) achieves 42.60%. We also train two other cascaded refinement models with ResNet-152 and ResNet-200 based architecture as the front model. The ensemble of the three models improves the performance to 44.54%, which outperforms previous methods by a substantial margin. We visualize the effect of the two networks. As shown in Figure 5, the discontinuous areas in stuff and large objects can be smoothed after GRN. Figure 6 provides the predictions after employing LRN, which can refine the object boundaries.

Method	Mean IoU
FCN-8s [Long *et al.*, 2015]	65.3%
CRF-RNN [Zheng *et al.*, 2015]	62.5%
Dilation10 [Yu and Koltun, 2016]	67.1%
DPN [Liu *et al.*, 2015]	66.8%
LRR-4x [Ghiasi and Fowlkes, 2016]	69.7%
DeepLab [Chen *et al.*, 2015a]	70.4%
Adelaide_Context [Lin *et al.*, 2016]	71.6%
Ours (single model)	76.15%
Ours (ensemble 3 models)	77.27%

Table 4: Comparison with other state-of-the-art methods on Cityscapes test set

Results on Cityscape Dataset: Table 2 reports the performance of the two refinement networks on Cityscapes validation set. In mean IoU, the front model (based on ResNet-101) attains 72.93%. Incorporating GRN and LRN individually yields 1.72% and 1.43% improvement respectively, while employing the cascaded architecture brings 2.70% improvement. Moreover, multi-scale fusion during testing improves the performance to 77.16%. By contrast, fully connected CRF only brings 0.54% improvement, while the MS-Dilation yields 1.03% gain, slightly poorer than the proposed two refinement networks. We give the comparison results with other state-of-the-art methods on Cityscapes test set in Table 4. Our single cascaded refinement model (with the front model based on ResNet-101) achieves 76.15% in terms of mean IoU, significantly higher than 71.6% by the published state-of-the-art algorithms of [Lin *et al.*, 2016]. We also train two other cascaded refinement models with ResNet-152 and ResNet-200 based architecture as the front model. The ensemble of the three models further improves the performance to 77.27%. As shown in Figure 7, GRN smoothens the discontinuous and inconsistent areas in stuff. Meanwhile, the LRN refines the object boundaries and details, as shown in Figure 8.

5 Conclusion

Scene parsing is a significant and challenging task in computer vision. In this paper, we propose two refinement networks, which rectify scene parsing predictions globally and locally. By capturing the global contextual dependencies, the Global-residual Refinement Network predicts the global parsing residuals and iteratively refine the inconsistent parsing results. The Local-boundary Refinement Network learns position-adaptive local propagation coefficients and refines the object boundaries locally. The two networks can be employed individually or in a cascading architecture. Experiments show that the proposed two refinement networks significantly improve the parsing accuracy and achieve state-of-the-art on the challenging ADE20K and Cityscapes datasets.

References

[Badrinarayanan *et al.*, 2015] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv:1511.00561*, 2015.

[Bell *et al.*, 2016] Sean Bell, C. Lawrence Zitnick, Kavita Bala, and Ross B. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proc. of CVPR*, 2016.

[Byeon *et al.*, 2015] Wonmin Byeon, Thomas M. Breuel, Federico Raue, and Marcus Liwicki. Scene labeling with LSTM recurrent neural networks. In *Proc. of CVPR*, 2015.

[Chen *et al.*, 2015a] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.

[Chen *et al.*, 2015b] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. In *Neural Information Processing Systems, Workshop on Machine Learning Systems*, 2015.

[Cordts *et al.*, 2016] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *Proc. of CVPR*, 2009.

[Ghiasi and Fowlkes, 2016] Golnaz Ghiasi and Charless C. Fowlkes. Laplacian pyramid reconstruction and refinement for semantic segmentation. In *Proc. of ECCV*, 2016.

[Gidaris and Komodakis, 2015] Spyros Gidaris and Nikos Komodakis. Object detection via a multi-region and semantic segmentation-aware CNN model. In *Proc. of ICCV*, 2015.

[Hariharan *et al.*, 2015] Bharath Hariharan, Pablo Andrés Arbeláez, Ross B. Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proc. of CVPR*, 2015.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of CVPR*, 2016.

[Ioffe and Szegedy, 2015] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. of ICML*, 2015.

[Jin *et al.*, 2016] Xiaojie Jin, Yunpeng Chen, Jiashi Feng, Zequn Jie, and Shuicheng Yan. Multi-path feedback recurrent neural network for scene parsing. *CoRR*, abs/1608.07706, 2016.

[Krähenbühl and Koltun, 2011] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in Neural Information Processing Systems*, 2011.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[Li *et al.*, 2016] Ke Li, Bharath Hariharan, and Jitendra Malik. Iterative instance segmentation. In *Proc. of CVPR*, 2016.

[Liang *et al.*, 2015] Xiaodan Liang, Chunyan Xu, Xiaohui Shen, Jianchao Yang, Si Liu, Jinhui Tang, Liang Lin, and Shuicheng Yan. Human parsing with contextualized convolutional neural network. In *Proc. of ICCV*, 2015.

[Liang *et al.*, 2016a] Xiaodan Liang, Xiaohui Shen, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with graph LSTM. In *Proc. of ECCV*, 2016.

[Liang *et al.*, 2016b] Xiaodan Liang, Xiaohui Shen, Donglai Xiang, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with local-global long short-term memory. In *Proc. of CVPR*, 2016.

[Lin *et al.*, 2016] Guosheng Lin, Chunhua Shen, Ian D. Reid, and Anton van den Hengel. Efficient piecewise training of deep structured models for semantic segmentation. In *Proc. of CVPR*, 2016.

[Liu *et al.*, 2015] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *Proc. of ICCV*, 2015.

[Long *et al.*, 2015] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proc. of CVPR*, 2015.

[Mostajabi *et al.*, 2015] Mohammadreza Mostajabi, Payman Yadollahpour, and Gregory Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *Proc. of CVPR*, 2015.

[Noh *et al.*, 2015] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proc. of ICCV*, 2015.

[Pinheiro and Collobert, 2014] Pedro H. O. Pinheiro and Ronan Collobert. Recurrent convolutional neural networks for scene labeling. In *Proc. of ICML*, 2014.

[Shrivastava *et al.*, 2016] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proc. of CVPR*, 2016.

[Shuai *et al.*, 2016] Bing Shuai, Zhen Zuo, Gang Wang, and Bing Wang. Dag-recurrent neural networks for scene labeling. In *Proc. of CVPR*, 2016.

[Vemulapalli *et al.*, 2016] Raviteja Vemulapalli, Oncel Tuzel, Ming-Yu Liu, and Rama Chellapa. Gaussian conditional random field network for semantic segmentation. In *Proc. of CVPR*, 2016.

[Yu and Koltun, 2016] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *Proc. of ICLR*, 2016.

[Zeiler *et al.*, 2011] Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *Proc. of ICCV*, 2011.

[Zheng *et al.*, 2015] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *Proc. of ICCV*, 2015.

[Zhou *et al.*, 2016] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *CoRR*, abs/1608.05442, 2016.