Patch Proposal Network for Fast Semantic Segmentation of High-Resolution Images

Tong Wu,[1*] **Zhenzhen Lei,**[1*] **Bingqian Lin,**[3] **Cuihua Li,**[1] **Yanyun Qu,**[1†] **Yuan Xie**[2†]
[1]Fujian Key Laboratory of Sensing and Computing for Smart City, School of Informatics, Xiamen University, Fujian, China
[2]School of Computer Science and Technology, East China Normal University, Shanghai, China
[3]School of Biomedical Engineering, Sun Yat-sen University, Guangzhou, China
{tongwu, zzlei}@stu.xmu.edu.cn, linbq6@mail2.sysu.edu.cn, {chli, yyqu}@xmu.edu.cn, yxie@cs.ecnu.edu.cn

Abstract

Despite recent progress on the segmentation of high-resolution images, there exist an unsolved problem, *i.e.,* the trade-off among the segmentation accuracy, memory resources and inference speed. So far, GLNet is introduced for high or ultra-resolution image segmentation, which has reduced the computational memory of the segmentation network. However, it ignores the importances of different cropped patches, and treats tiled patches equally for fusion with the whole image, resulting in high computational cost. To solve this problem, we introduce a patch proposal network (PPN) in this paper, which adaptively distinguishes the critical patches from the trivial ones to fuse with the whole image for refining segmentation. PPN is a classification network which alleviates network training burden and improves segmentation accuracy. We further embed PPN in a global-local segmentation network, instructing global branch and refinement branch to work collaboratively. We implement our method on four image datasets:DeepGlobe, ISIC, CRAG and Cityscapes, the first two are ultra-resolution image datasets and the last two are high-resolution image datasets. The experimental results show that our method achieves almost the best segmentation performance compared with the state-of-the-art segmentation methods and the inference speed is 12.9 fps on DeepGlobe and 10 fps on ISIC. Moreover, we embed PPN with the general semantic segmentation network and the experimental results on Cityscapes which contains more object classes demonstrate the generalization ability on general semantic segmentation.

Introduction

With the rising up of deep learning, semantic segmentation achieves prominent progress. Recently, more focuses are shifted on solving the semantic segmentation problem of high-resolution or ultra-high resolution images (HRI or URI) by implementing the existing deep segmentation models for special applications, such as medical diagnosis (Tschandl, Rosendahl, and Kittler 2018; Codella et al. 2018; Graham et al. 2019), urban planning and road extraction (Demir et al. 2018). However, semantic segmentation of HRI or URI containing more than 2M or 4M pixels, respectively, requires large computational resources and consumes a lot of time. The most popular semantic segmentation methods, such as FCN (Long, Shelhamer, and Darrell 2015), PSPNet (Zhao et al. 2017), are difficult to be applied directly, because the standard GPU devices are hard to afford such huge computational burden, causing computational difficulties as well as even poor segmentation performance.

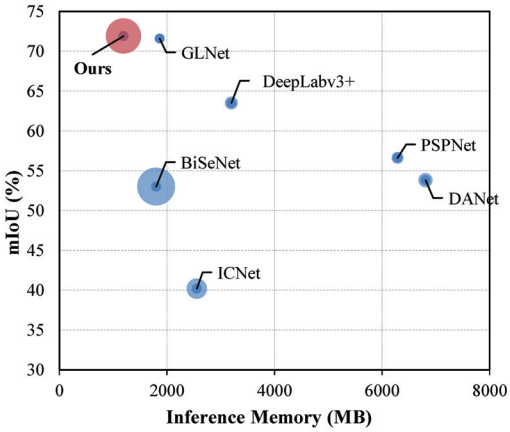

Figure 1: Performance comparison in inference memory usage, mIoU and inference speed on DeepGlobe. The bigger the circle is, the faster the speed.

To solve the semantic segmentation problem of URI or HRI, there exist three classes of segmentation methods: global approaches, local approaches and collaborative global-local approaches. The global methods firstly down sample an input image to a middle or small resolution one, and then implement a deep segmentation model to solve the segmentation problem. Meanwhile, the local methods conduct a deep segmentation model on the patches of HRI after the input image is divided into several patches. It has been observed that global methods, only using down-sampled images, lose details, causing inaccurate edge segmentation. And the local methods may destroy the unity of an object which makes it difficult to classify correctly the object only

[*]Equal contribution.
[†]Corresponding author.

depending on a patch. A more promising way, *i.e.,* the collaborative global-local methods, which combine global and local methods in a collaborative manner, apply effective fusion mechanisms to fuse features from both global and local information for better segmentation.

Despite the recent progress in semantic segmentation of HRI, a problem remains unsolved: How to balance among the computational resource, the inference speed as well as the segmentation accuracy. The latest method GLNet (Chen et al. 2019), belonging to the global-local methods, conducts a bidirectional combination of feature maps at each layer with global context and local fine structures. Although reasonably effective, it is still time-consuming, because it treats every image patch segmentation equally and fuses them with the whole image segmentation. So in order to effectively and efficiently handle the fusion of global context and local details, we consider the importance of image patches, that is, that important patches should be allocated more computing resources to get more critical detail structures while trivial patches are not processed.

In this paper, we propose an elegant and efficient solution with better segmentation performance and fast speed. Draw lesson from Region Proposal Network in Faster R-CNN (Ren et al. 2015), we design a Patch Proposal sub-Network (PPN), which is a binary classification network and aims to distinguish important patches from trivial patches. In other words, PPN chooses patches that contain the object edges or details that need to be refined, while patches only contain background or flat regions that tend to be ignored. We further embed PPN into a global-local network which contains a global branch and a refinement branch, named GRNet. Different from GLNet, we allocate more computational resources to more important patches, which avoids the time consumption of trivial patches. In addition, we fuse the global and local feature maps only once.

The contributions of this paper can be summarized as follows:

- PPN is designed to select the important patches from the trivial patches. PPN is a classification network with an elegant discriminant rule. And it alleviates network training burden and improves segmentation results.
- We embed PPN in a global-refinement network (GRNet) for semantic segmentation of HRI or URI and use PPN to instruct both the global branch and the refinement branch to work collaboratively.
- The proposed GRNet achieves the best performance compared with state-of-the-art methods on 3 public high-resolution datasets: DeepGlobe, ISIC and CRAG. Especially, our method achieves 12.9 fps on a GPU on DeepGlobe dataset, thus, it is a practical segmentation method in terms of both speed and accuracy.
- PPN has good generalization ability. It can be easily and directly integrated into other popular semantic segmentation frameworks. PPN improves the segmentation performance of the baseline semantic segmentation network.

Related Work

It is recognized that the multi-scale scheme which integrates multi-scale context information is very effective in segmentation. RefineNet (Lin et al. 2017a) adds a multi-path refinement block to recursively exploit multi-scale features at different levels. Feature Pyramid Network (FPN) (Lin et al. 2017b) utilizes multi-scale semantic information to achieve prediction by top-down fusion mechanism of different layers.

Moreover, context aggregation plays an important role in the segmentation method. DeepLab (Chen et al. 2017) adopts a dilated convolution and atrous spatial pyramid pooling module to help extend the receptive field, which is beneficial to the better aggregate global context into fine-grained features. PSPNet (Zhao et al. 2017) employs pyramid pooling module which aggregates context information in different regions to improve the ability of capturing global context.

In addition, for the purpose of real-time and low latency, the pursuit of a fast or real-time semantic segmentation model attracts more attention. ICNet (Zhao et al. 2018) adopts a cascade feature fusion mechanism that takes advantage of low-resolution information as well as details of high-resolution images to refine segmentation prediction. BiSeNet (Yu et al. 2018a) builds two paths: a spatial path which is responsible for obtaining spatial information and context path which achieves a larger receptive field, then it uses the feature fusion module to integrate the output of two paths. Although these real-time segmentation networks have low computational complexity and memory consumption, they get much less segmentation accuracy than others.

The Proposal Method

The Architecture of GRNet

In this section, we introduce the framework of GRNet. Fig. 2 shows the architecture of GRNet. GRNet contains three components: the global branch (G-branch), PPN and the refinement branch (R-branch). The down-sampling images are fed into G-branch and PPN. G-branch is used to generate the preliminary global-level segmentation feature of the down-sampling image. PPN selects the important patches. After that, G-branch and PPN guide R-branch to refine the segmentation of the selected patches, which is regarded as Feature extraction and Refinement. Subsequently, the global-level feature and the refined local feature are fused to generate the final segmentation. In the following, we will detail the operating mechanism of PPN, Feature extraction and Refinement as well as Feature fusion.

Patch Proposal Network. PPN is an independent network, acting as a selector. In the testing stage, PPN handles the selection without supervision. Thus, PPN must learn a selection rule in the training stage. Inspired by teachers in teaching, if the teacher wants to improve the average grade of the group, an easier way is to improve the grades of students that below the average grade. We therefore adopt a similar discriminant selection rule to instruct PPN to perform patches selection. When the segmentation score I_c of current feature patch is lower than the overall average score

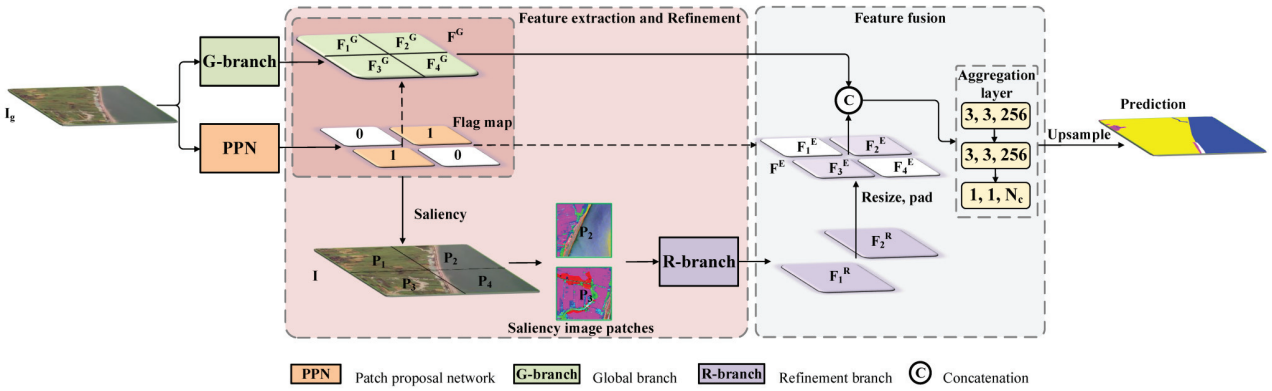

Figure 2: The architecture of GRNet. GRNet contains three components: G-branch, PPN, and R-branch. The downsampling image is firstly processed by G-branch and PPN. Then the results guide R-branch to refine the segmentation. After that, the refined segmentation results are fused with the output of the global branch in Feature fusion block. Finally, the segmentation results are generated.

I_t, this feature patch would be selected and recorded as 1, which is formulated as Eq. (1). Here, we take the mean intersection over union (mIoU) which is obtained in G-branch as the measurement for segmentation score. About the label generation for PPN, we will elaborate on training details. In the following, we detail the structure of PPN.

$$K = \begin{cases} 1 & I_c < I_t \\ 0 & otherwise \end{cases} \tag{1}$$

We down sample a URI or HRI $I \in \mathbb{R}^{H \times W}$ to $I_g \in \mathbb{R}^{H_g \times W_g}$, and equally split I into patches $\{P_i\}_{i=1}^{N} \in \mathbb{R}^{H_p \times W_p}$ without overlap, where N represents the amount of patches. Then, I_g is fed into G-branch to obtain global-level segmentation feature F^G. Meanwhile, we uniformly divide F^G into some feature patches $\{F_i^G\}_{i=1}^{N}$ in the same way as we divide image I_g, where F_i^G is the i-th feature patch in F^G.

PPN is a classification network, and the architecture of PPN is shown in Fig. 3. Downsampling image I_g is fed into the backbone of PPN to extract the deep features F^B, which is tiled into feature patches $\{F_i^B\}_{i=1}^{N}$. Then F^B and F_i^B are further handled by the average pooling layer and then go through the fully connected layer to obtain the global score G_{score} and the patch score P_{score}. Subsequently, the difference between G_{score} and each P_{score} is input into the sigmoid function for prediction and the prediction results form the flag map, where the value 1 represents the predicted probability ≥ 0.5, and the value 0 represents the predicted probability < 0.5. We apply Binary Cross Entropy (BCE) Loss for PPN, which is a standard practice for training a binary classification network.

Feature Extraction and Refinement. According to the flag map of PPN, we adaptively select the feature patch F_i^G from G-branch and its corresponding patch P_i of the original image I at the counterpart location to make G-branch and R-branch work collaboratively. To refine the patch segmentation, we firstly magnify F_i^G to the size of P_i. After that, we implement the saliency operation similar to (Yu et al. 2018b) on the magnified F_i^G and conduct the element-wise multiplication between the result and P_i, after that, we get the saliency image patches which are fed into R-branch. R-branch outputs the refined patch segmentation F_i^R.

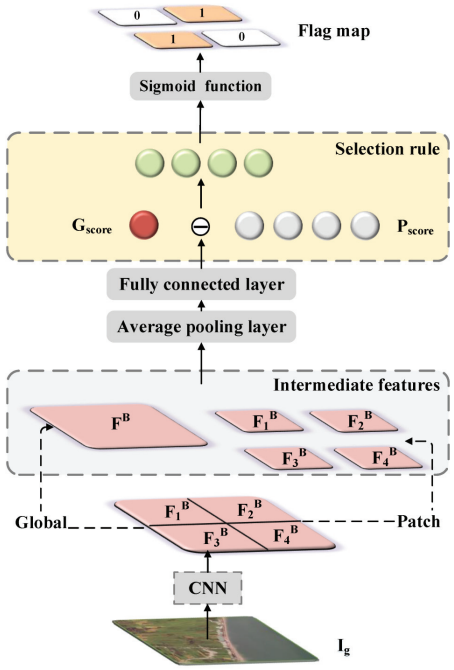

Figure 3: The framework of Patch Proposal Network. The whole preliminary features and feature patches are fed sequentially into the average pooling layer, fully connection layer and sigmoid function to obtain the flag map. The minus sign indicates the element-wise subtraction between G_{score} and P_{score}.

Feature Fusion. In order to prepare a simple and effective fusion mechanism that better fuses the global-level feature F^G from G-branch and the selected local features F_i^R

from R-branch, we first reconstruct a feature map F^E (initialized to 0) with the same size as F^G and uniformly divided into $\{F_1^E, F_2^E, ..., F_i^E, ..., F_N^E\}$. If the i-th patch is selected by PPN, then the refined feature F_i^R will replace the counterpart F_i^E in F^E. In particular, the selected patches should be fused with the counterpart in F^G, while the unselected ones are unchanged. Finally, the reconstructed F^E is concatenated with F^G and then the result is fed into the aggregation layer which contains three convolutional layers: $(3, 3, 256)$, $(3, 3, 256)$, $(1, 1, N_c)$ where the triple (k, k, s) means that the convolution kernel is $k \times k$ with the stride of 1 and s channels, and N_c is the number of classes.

Overall Loss Function

We utilize the focal loss (Lin et al. 2017c) for all output of network. In detail, we define $\mathcal{L}_\mathcal{G}, \mathcal{L}_\mathcal{R}, \mathcal{L}_\mathcal{A}$ as the G-branch, R-branch and aggregation loss, respectively. $\mathcal{L}_\mathcal{G}$,$\mathcal{L}_\mathcal{R}$ and $\mathcal{L}_\mathcal{A}$ are formulated as:

$$\begin{aligned}
\mathcal{L}_\mathcal{G} &= \begin{cases} -(1-y_g{}')^\gamma log y_g{}', & y_g = 1 \\ -y_g{}'^\gamma log(1-y_g{}'), & y_g = 0 \end{cases} \\
\mathcal{L}_\mathcal{R} &= \begin{cases} -(1-y_r{}')^\gamma log y_r{}', & y_r = 1 \\ -y_r{}'^\gamma log(1-y_r{}'), & y_r = 0 \end{cases} \\
\mathcal{L}_\mathcal{A} &= \begin{cases} -(1-y_a{}')^\gamma log y_a{}', & y_a = 1 \\ -y_a{}'^\gamma log(1-y_a{}'), & y_a = 0 \end{cases}
\end{aligned} \tag{2}$$

where $y_g{}', y_r{}'$ and $y_a{}'$ represent the pixel-wise label classification of G-branch, R-branch and Aggregation layer, respectively, and y_g, y_r, y_a are the ground truth. The overall loss function $\mathcal{L}$ is formulated as:

$$\mathcal{L} = \mathcal{L}_\mathcal{G} + \mathcal{L}_\mathcal{R} + \mathcal{L}_\mathcal{A}. \tag{3}$$

Training

In this paper, we adopt a 3-step training scheme via alternating optimization. Here, we define the network except PPN as Mainbody which includes G-branch, R-branch and Aggregation layer, and PPN is treated as a selector. We use FPN (Lin et al. 2017b) with ResNet50 (He et al. 2016) as the backbone for G-branch and use FPN with ResNet18 (He et al. 2016) as the backbone for R-branch. PPN uses ResNet18 (He et al. 2016) without the last residual block as the backbone. Firstly, we train G-branch, which is fine-tuned end-to-end for the global segmentation task. Secondly, we train PPN under the guidance of the G-branch output. G-branch and PPN are two separate networks. Finally, we refine the segmentation in R-branch. The refining results together with the output of G-branch are fused under the guidance of PPN in Aggregation layer. Mainbody is tuned with PPN fixed. The alternating training is conducted iteratively until the setting iteration number reaches.

Training G-branch. We train G-branch independently for a few epochs. Specifically, we feed the downsampled image I_g into G-branch, and obtain the global segmentation feature F^G, and then the weights of G-branch is updated with the loss function $\mathcal{L}_\mathcal{G}$.

Training PPN. For training PPN, we must prepare the label for PPN, thus, we input I_g to the fixed G-branch, and obtain the output F^G, then we calculate mIoU of the overall F^G and that of the patch F_i^G. According to the select strategy used in PPN, if the mIoU of F_i^G is less than that of F^G, the i-th patch should be selected and the corresponding label set to 1, otherwise set to 0. For PPN, we adopt BCE Loss for training.

Training Mainbody. After training PPN for one epoch, we make PPN fixed and utilize its output as the flag map which guides the refinement and fusion in Mainbody. We use the overall loss function $\mathcal{L}$ in Eq. (3) to tune the weights of Mainbody. We alternately train PPN and Mainbody with one epoch per iteration. This training algorithm is run until reaching the maximum epoch or the algorithm converges.

Experiments

In this section, we implement our method on four datasets: CRAG, ISIC, DeepGlobe, and Cityscapes to evaluate the effectiveness and generalization of our proposed network. The first two datasets containing medical images are HRI and URI datasets, respectively, and DeepGlobe is a URI dataset containing satellite images. We evaluate the effectiveness and efficiency of our method on the first three datasets where HRI or URI segmentation are demanded increasingly and hardly explored. Cityscapes is a benchmark semantic segmentation dataset with HRI images. We evaluate the generalization ability of PPN on Cityscapes. We further conduct an ablation study to explore how each component of our method influences the segmentation performance. We take three criteria to evaluate the performance of our method: the segmentation accuracy, the memory usage, and the inference speed.

Implementation Details

In our model, an image with the size of 512×512 is fed into G-branch and then it is uniformly partitioned into 4×4 block to generate 16 patches. We use Adam (Kingma and Ba 2014) with initial learning rate as 1×10^{-4} to optimize G-branch, R-branch and PPN. The weight decay coefficient and momentum are set to 5×10^{-4} and 0.9, respectively. The parameter γ in Focal loss functions of G-branch and R-branch is set to 3. The epochs of pre-training G-branch and maximum number of alternate training are set to 10 and 120, respectively. We train our model with 10 batches on a single 1080Ti GPU in a PyTorch framework (Ketkar 2017). We use the terminal tool "gpustat" to measure the GPU memory usage. For the fairness of comparison with the state-of-the-art methods, we set the batch size to 1 during inference. Because some comparison methods (e.g. PSPNet, DANet) can not process the whole images without downsampling during inference, similar to (Chen et al. 2019), we adopt an appropriate downsampling rate to avoid over-down sampling and reduce the loss of resolution in training [1].

[1]In CRAG, The downsampling rate for PSPNet and DeepLabv3+ is 0.8, and DANet is 0.7, GLNet maintains original settings, the other methods do not perform downsampling operation. In DeepGlobe, the downsampling rate for BiSeNet

Datasets

DeepGlobe (Demir et al. 2018) is a high-quality satellite dataset focusing on rural areas, which provides 803 images in 7 classes with 2448×2448 pixels. We randomly divide the dataset into training, validation and testing sets with 455, 206 and 142 images, respectively. In particular, the objects out of seven class named as "unknown" which are not discussed in our experiments.

ISIC (Tschandl, Rosendahl, and Kittler 2018; Codella et al. 2018) is an ultra-resolution medical dataset for pigmented skin lesions, whose training set contains 2077 images, validation set contains 260 images and testing set contains 259 images. Average resolution of ISIC is up to 9 M. The largest image is up to the size of 6748×4499. The dense annotations contain two classes: lesion region and background. However, due to the large proportion of the background, we only select the lesion region for evaluation.

CRAG (Graham et al. 2019; Awan et al. 2017) is a HRI dataset that includes two classes and exhibits different differentiated glandular morphology. The CRAG dataset is split into training set and testing set which contain 173 and 40 images. Their average size is 1512×1516.

Cityscapes (Cordts et al. 2016) is a street scene dataset which usually used by general semantic segmentation methods for evaluation. It contains 3475 fine annotated images with the size of 2048×1024 and 2975 images are used for training and the rest is used for validation. There are 19 pre-defined semantic classes, and objects outside of these 19 classes will be ignored in both training and validation phase.

Ablation Studies

Effect of PPN. We implement our method on DeepGlobe and ISIC to investigate the effect of PPN. Fig. 4 shows some examples of selected patches by PPN. It is observed that G-branch only succeeds in part of the ground truth, and it cannot well deal with the detail structures. However, PPN selects most of the regions which are required for refining. The number of the selected regions is much less than the total number of tiled patches. Fig. 5 and Fig. 6 show the complete segmentation results of two instances from DeepGlobe and ISIC, respectively. As can be observed, the segmentation results of G-branch are not so accurate. In particular, the details of the object boundary are not well segmented. However, PPN selects the important patches containing rich details and ignores the trivial patches in the ground truth images, such as the yellow homogeneous patches in Fig. 5 (c) and the black homogeneous patches in Fig. 6 (c). The selected patches get more details after processed by R-branch, and the final fusion results of these patches make better segmentation on the boundaries and other detail structures.

Effect of the selected patch amount. Moreover, we perform an ablation study on the number of the tiled patches. In the experiments, we compare the segmentation results of GRNet with different number of patches: 4, 16, and 64. As shown in Table 1, it shows that with the increase of the number of selected patches, the segmentation result becomes better while the memory usage is larger. Considering the trade-off between the segmentation accuracy and memory usage, we select 16 proposal patches in our experiments.

Effect of our method. In order to investigate the effect of the architecture of our method, we construct three vari-

is 0.8, and DANet is 0.5, and other methods are consistent with (Chen et al. 2019). In ISIC, we downsample the URI to 1990×1990 for BiSeNet, and 1244×1244 for DANet, other methods are consistent with (Chen et al. 2019).

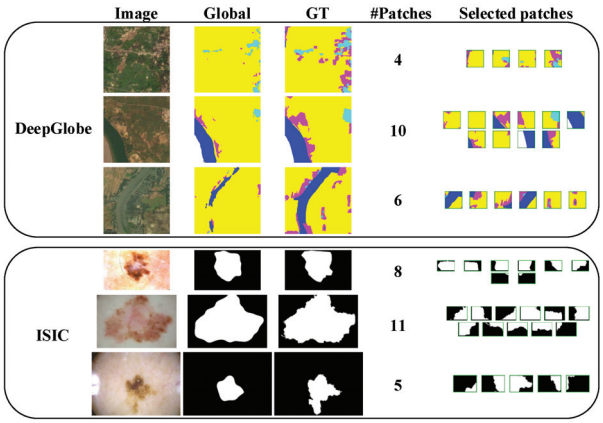

Figure 4: Example results for selecting patches. The column "Global" represents the global prediction from G-branch, "GT" represents the ground-truth, and "Selected patches" gives the selected patches in GT as a reference, which should be compared to the counterpart patches in the global prediction, and the comparison results show that in the global prediction need to be improved.

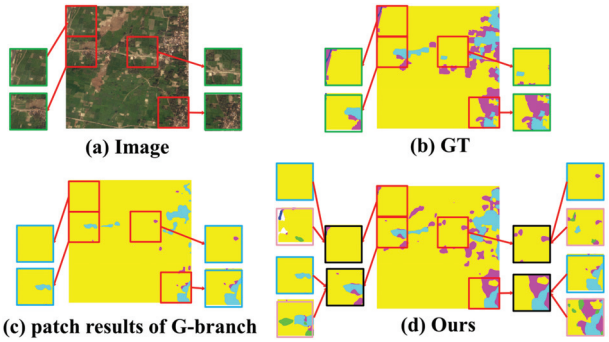

Figure 5: The selected patches by PPN and segmentation results of an image in DeepGlobe. The patches with green boundary represent the original image patches or ground truth. And the patches with blue, pink and black boundary represent results of G-branch, results of R-branch and ours final fusion results, respectively.

Table 1: Ablation study on different amount of patches on DeepGlobe.

patch num	Memory(M)	Time(ms)	mIoU(%)
4	1239	12371	71.5
16	1193	10867	71.9
64	1131	14862	72.4

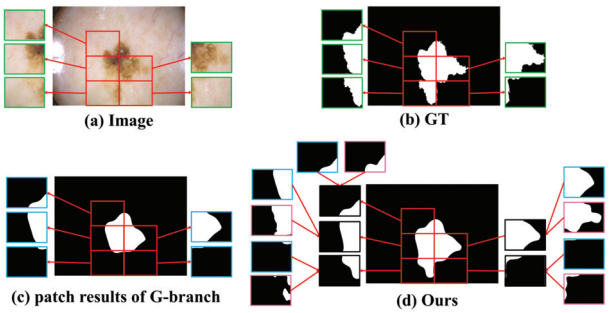

Figure 6: The selected patches by PPN and segmentation results of an image in ISIC. The patches with green boundary represent the original image patches or ground truth. And the patches with blue, pink and black boundary represent results of G-branch, results of R-branch and our final fusion results, respectively.

Table 2: Ablation experiments of our method on CRAG. Local and global represent using only patch-wise in image and only global context respectively for segmentation. Enhance indicates that utilize all patch-wise equally during parsing.

Model	Memory(M)	Time(ms)	mIoU(%)
LocalNet	853	2718	75.5
GlobalNet	865	3556	88.5
EnhanceNet	947	6217	87.7
Ours	945	5260	88.9

ants of our network: 1) GlobalNet: only G-branch is used. 2) LocalNet: only refinement branch is used and the input is the tiled patches. 3) EnhanceNet: Mainbody is used without PPN and the patches are equally refined and fused with the global feature F^G. We compare our method with the three variants. Table 2 gives the results on CRAG. We find that LocalNet is the worst variant with the segmentation accuracy 75.5%. It indicates that due to losing wholeness of the image, LocalNet performs not well. GlobalNet performs better than LocalNet with the gain 13%. EnhanceNet is a little worse than GlobalNet attribute to the erroneous fusion with local patches. GRNet is the best, which benefits from PPN to select important patches for effective fusion. It is worth noting that the result on CRAG of GRNet is higher by 1.2% than EnhanceNet without consuming too much memory.

Comparisons with state-of-the-art methods

Results on DeepGlobe dataset. We compare our method with six state-of-the-art methods: PSPNet (Zhao et al. 2017), ICNet (Zhao et al. 2018), BiSeNet (Yu et al. 2018a), DeepLabV3+ (Chen et al. 2018), DANet (Fu et al. 2019), and GLNet (Chen et al. 2019) on DeepGlobe. The comparison results are shown in Table 3. Our model outperforms other state-of-the-art approaches with the segmentation accuracy of 71.9% in mIoU and only uses 1193 MB GPU memory. Moreover, our model outperforms DeepLabv3+, with the gain of 7.7% in mIoU. Our model is slightly superior to GLNet in mIoU, with the reduction of 672 MB memory during testing. As for the inference speed, our model is almost the fastest method with 12.9 fps except BiSeNet. Note that our method yields a gain of 7.6 fps compared with ICNet which is a real-time segmentation method. It is obvious that our model achieves the best comprehensive performance. Fig. 7 gives the comparison results in visual effect on DeepGlobe. It is observed that our method achieves the closest segmentation results to the ground truth. PSPNet, ICNet, BiSeNet, DeepLabv3+ perform not well in HRI segmentation. The latest method GLNet yields a relative precise segmentation, but is inferior to our method in edge segmentation.

Table 3: Comparison with state-of-the-art approaches on DeepGlobe. DLv3+ is short for DeepLabv3+.

Model	Memory(M)	Time(ms)	FPS	mIoU(%)
PSPNet	6289	135964	1.0	56.6
ICNet	2557	26798	5.3	40.2
BiSeNet	1801	**9909**	**14.2**	53.0
DLv3+	3199	89557	1.6	63.5
DANet	6812	62902	2.3	53.8
GLNet	1865	276397	0.5	71.6
Ours	**1193**	10867	12.9	**71.9**

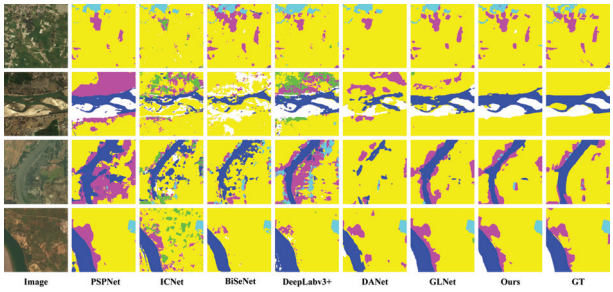

Figure 7: Comparison of the state-of-the-art segmentation methods on DeepGlobe.

Results on ISIC [2] **dataset.** We further compare our method with the six state-of-the-art methods on ISIC. As shown in Table 4, GRNet beats almost the compared methods in segmentation accuracy except PSPNet with the smallest consuming memory. Especially, our model achieves $76.5\%/1389$ MB in terms of mIoU and memory and with the gain of $1.3\%/523$ MB compared to GLNet, which achieves the best balance between accuracy and memory usage before ours. Compared with mentioned real-time segmentation network BiSeNet, although our inference speed is slower than BiSeNet, our method outperforms BiSeNet in both segmentation accuracy and memory consumption with a large margin. PSPNet slight outperforms our model in mIoU, while their GPU memory usage is $2.5\times$ of that of ours and the inference speed is $5.2\times$ of that of ours.

Results on CRAG dataset. Table 5 gives the comparison results on CRAG. Our method outperforms other state-of-the-art approaches both in accuracy and memory usage.

[2]Consistent with (Chen et al. 2019), we take the metrics: score = 0 if IoU < 0.65; score = IoU, otherwise.

Table 4: Comparison with state-of-the-art approaches on ISIC. DLv3+ is short for DeepLabv3+.

Model	Memory(M)	Time(ms)	FPS	mIoU(%)
PSPNet	3679	127429	2.0	**77.0**
ICNet	1593	23879	11.0	33.8
BiSeNet	1575	**15741**	**16.3**	43.7
DLv3+	2033	85811	3.0	70.5
DANet	3888	67881	3.8	51.4
GLNet	1912	638854	0.4	75.2
Ours	**1389**	24371	10.8	**76.5**

Table 5: Comparison with state-of-the-art approaches on CRAG. DLv3+ is short for DeepLabv3+.

Model	Memory(M)	Time(ms)	FPS	mIoU(%)
PSPNet	3750	20397	2.0	88.6
ICNet	2580	9010	4.4	77.6
BiSeNet	1173	**3524**	**10.0**	88.1
DLv3+	3123	25949	1.5	**88.9**
DANet	4063	14092	2.9	82.3
GLNet	1763	42483	0.9	85.9
Ours	**945**	5260	8.0	**88.9**

Table 6: Evaluation of the generalization ability on Cityscapes. "BaseNet" refers to BiSeNet.

Model	Memory(M)	Time(ms)	FPS	mIoU(%)
BaseNet	1053	12417	40.3	74.7
Ours	1137	20793	24.0	**75.2**

In detail, DeepLabv3+ yields the equal accuracy to ours in mIoU. Nevertheless, our method runs $5.2\times$ faster in FPS, and use $3.3\times$ less in memory usage than DeepLabv3+. As for inference time and speed, our method is just slower than BiSeNet. In summary, compared with real-time segmentation networks, our method achieves the best accuracy and the comprehensive performance on CRAG.

The generalization of PPN

We verify the generalization ability of PPN. We want to know if PPN improves the segmentation performance of semantic segmentation if the baseline segmentation model can work. For quick implementation and validation, we chose BiSeNet (Yu et al. 2018a) as our baseline network, which is the state-of-the-art real-time framework and is excellent in both effect and efficiency. We choose the version in (Yu et al. 2018a) which use ResNet18 as backbone and evaluate whole image without any test strategies. PPN gets 74.7% mIoU on the validation set, which is basically the same as the best 74.8% reported in their paper.

Next we directly replace BiSeNet with G-branch, the structure of the other parts remains unchanged (any other refinement mechanism is also feasible). The input size of R-branch is consistent with G-branch for simplicity, which is 1536×768 as the same as (Yu et al. 2018a). We also set the number of the tiled patches as 16. As shown in Table 6, although the BaseNet has almost reached its best performance, under the guidance of PPN, the final performance

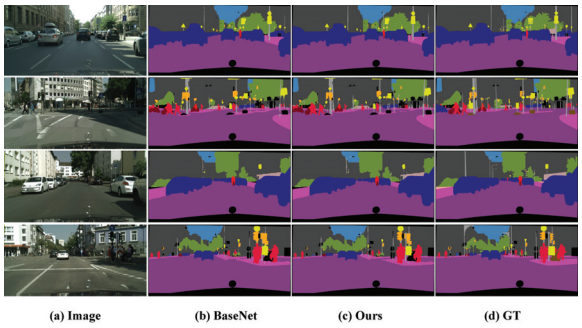

Figure 8: Segmentation results of BaseNet and our method on Cityscapes. "BaseNet" refers to BiSeNet. Note that the black area in the results belongs to the ignored classes.

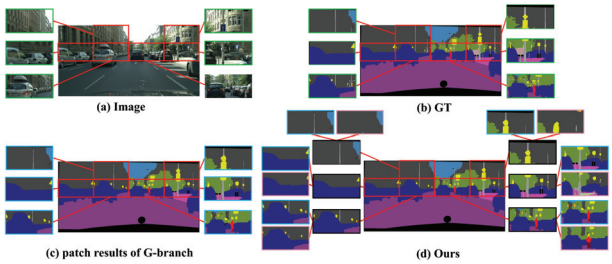

Figure 9: The selected patches by PPN and segmentation results of an image in Cityscapes. The patches with green boundary represent the original image patches or ground truth. And the patches with blue, pink and black boundary represent results of G-branch, results of R-branch and our final fusion results, respectively.

still can be further improved with slight memory and speed costs. The comparison results are shown in Table 7, it shows that in most cases our method is better than the BaseNet. It's worth noting that "pole" and "tlight (traffic light)" are challenging classes in Cityscapes, since their scale is relatively small, and it is difficult to fully capture for global segmentation. With the help of PPN, the difficulty is alleviated. The qualitative results and the patch proposal mechanism by PPN are show in Fig. 8 and Fig. 9, respectively. It further demonstrates that the segmentation network with PPN not only achieves good performance in terms of efficiency and accuracy in URI or HRI segmentation, but also can be directly embedded with other popular semantic segmentation frameworks to achieve better segmentation.

Conclusions

In this work, we propose PPN for better trade-off among segmentation accuracy, inference speed and the memory usage for semantic segmentation of HRI or URI. PPN is embedded in a global-local framework to select the important patches which are further refined. The experimental results on DeepGlobe, ISIC and CRAG demonstrate that our method achieves the best comprehensive performance. Moreover, PPN can also be embedded to other semantic segmentation frameworks and we implement it on Cityscapes

Table 7: Comparison in mIoU for each class on Cityscapes. "BaseNet" refers to BiSeNet.

Model	road	swalk	build	wall	fence	pole	tlight	sign	veg.	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU(%)
BaseNet	97.9	83.7	91.9	53.2	56.4	60.2	63.0	76.4	91.9	**62.3**	94.2	79.1	57.1	94.2	**74.9**	83.2	70.4	**56.6**	73.5	74.7
Ours	**98.0**	**83.8**	**92.0**	53.2	**56.6**	**63.2**	**67.0**	**77.1**	91.9	62.1	**94.5**	**79.4**	**57.2**	**94.6**	74.7	**83.5**	**71.0**	55.9	**73.7**	**75.2**

which is the benchmark semantic segmentation. The experimental results show that PPN has good generalization ability.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant 61876161, Grant 61772524, Grant U1065252 and partly by the Beijing Municipal Natural Science Foundation under Grant 4182067, and partly by the Fundamental Research Funds for the Central Universities associated with Shanghai Key Laboratory of Trustworthy Computing.

References

Awan, R.; Sirinukunwattana, K.; Epstein, D.; Jefferyes, S.; Qidwai, U.; Aftab, Z.; Mujeeb, I.; Snead, D.; and Rajpoot, N. 2017. Glandular morphometrics for objective grading of colorectal adenocarcinoma histology images. *Scientific reports* 7(1):16852.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence* 40(4):834–848.

Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; and Adam, H. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, 801–818.

Chen, W.; Jiang, Z.; Wang, Z.; Cui, K.; and Qian, X. 2019. Collaborative global-local networks for memory-efficient segmentation of ultra-high resolution images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8924–8933.

Codella, N. C.; Gutman, D.; Celebi, M. E.; Helba, B.; Marchetti, M. A.; Dusza, S. W.; Kalloo, A.; Liopyris, K.; Mishra, N.; Kittler, H.; et al. 2018. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 168–172. IEEE.

Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; and Schiele, B. 2016. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Demir, I.; Koperski, K.; Lindenbaum, D.; Pang, G.; Huang, J.; Basu, S.; Hughes, F.; Tuia, D.; and Raska, R. 2018. Deepglobe 2018: A challenge to parse the earth through satellite images. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 172–17209. IEEE.

Fu, J.; Liu, J.; Tian, H.; Li, Y.; Bao, Y.; Fang, Z.; and Lu, H. 2019. Dual attention network for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3146–3154.

Graham, S.; Chen, H.; Gamper, J.; Dou, Q.; Heng, P.-A.; Snead, D.; Tsang, Y. W.; and Rajpoot, N. 2019. Mild-net: Minimal information loss dilated network for gland instance segmentation in colon histology images. *Medical image analysis* 52:199–211.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Ketkar, N. 2017. Introduction to pytorch. In *Deep learning with python*. Springer. 195–208.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lin, G.; Milan, A.; Shen, C.; and Reid, I. 2017a. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1925–1934.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017b. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2117–2125.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017c. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3431–3440.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Tschandl, P.; Rosendahl, C.; and Kittler, H. 2018. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data* 5:180161.

Yu, C.; Wang, J.; Peng, C.; Gao, C.; Yu, G.; and Sang, N. 2018a. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 325–341.

Yu, Q.; Xie, L.; Wang, Y.; Zhou, Y.; Fishman, E. K.; and Yuille, A. L. 2018b. Recurrent saliency transformation network: Incorporating multi-stage visual cues for small organ segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8280–8289.

Zhao, H.; Shi, J.; Qi, X.; Wang, X.; and Jia, J. 2017. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2881–2890.

Zhao, H.; Qi, X.; Shen, X.; Shi, J.; and Jia, J. 2018. Icnet for real-time semantic segmentation on high-resolution images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 405–420.