CascadePSP: Toward Class-Agnostic and Very High-Resolution Segmentation via Global and Local Refinement

Ho Kei Cheng*  Jihoon Chung*  
HKUST  
{hkchengad,jchungaa}@cs.ust.hk

Yu-Wing Tai  
Tencent  
yuwingtai@tencent.com

Chi-Keung Tang  
HKUST  
cktang@cs.ust.hk

Abstract

State-of-the-art semantic segmentation methods were almost exclusively trained on images within a fixed resolution range. These segmentations are inaccurate for very high-resolution images since using bicubic upsampling of low-resolution segmentation does not adequately capture high-resolution details along object boundaries. In this paper, we propose a novel approach to address the high-resolution segmentation problem without using any high-resolution training data. The key insight is our CascadePSP network which refines and corrects local boundaries whenever possible. Although our network is trained with low-resolution segmentation data, our method is applicable to any resolution even for very high-resolution images larger than 4K. We present quantitative and qualitative studies on different datasets to show that CascadePSP can reveal pixel-accurate segmentation boundaries using our novel refinement module without any finetuning. Thus, our method can be regarded as class-agnostic. Finally, we demonstrate the application of our model to scene parsing in multi-class segmentation.

1. Introduction

Resolution of commodity cameras and displays has significantly increased with 4K UHD (3840 × 2160) being the high industry standard. Despite the demand for high-resolution media, many state-of-the-art computer vision algorithms face various challenges with images with high pixel count. Image semantic segmentation is one of these computer vision tasks. Models for semantic segmentation in deep learning designed for low-resolution images (e.g. PASCAL or COCO dataset) often fail to generalize to higher resolution scenarios. Specifically, these models typically use GPU memory linear to the number of pixels, making it practically impossible to directly train a 4K UHD segmentation. High-resolution training data for semantic segmentation is difficult to obtain because pixel-accurate annotation is required, much less that even such high-resolution training data are available, to train a model on very high-resolution images, a much larger receptive field is required to capture sufficient semantics. Plausible workarounds include downsampling and cropping, but the former removes details while the latter destroys image context.

This paper proposes CascadePSP1, a general segmentation refinement model that refines any given segmentation from low to high resolution. Our model is trained independently and can be easily appended to any existing methods to improve their segmentation, a finer and more accurate segmentation mask of an object can be produced. Our model takes as input an initial mask that can be an output of any algorithm to provide a rough object location. Then our CascadePSP will output a refined mask. Our model is designed in a cascade fashion that generates refined segmentation in a coarse-to-fine manner. Coarse outputs from the early levels predict object structure which will be used as input to the latter levels to refine boundary details.

1Source code, pretrained models and dataset are available at https://github.com/hkchengrex/CascadePSP.

Figure 1. Segmentation of a high-resolution image (3492 × 2328). Left: Produced by Deeplab V3+ [6]. Right: Refined by our algorithm.
tion in very high-resolution but also refines and corrects erroneous boundary to produce more accurate result.

To evaluate on very high-resolution images, we have annotated a high-resolution dataset with 50 validation and 100 test objects with the same semantic classes as in PASCAL, dubbed the BIG dataset. We test our model on PASCAL VOC 2012, BIG, and ADE20K. With a single model without using the dataset itself for finetuning, we have achieved consistent improvement over the state-of-the-art methods across these datasets and models. We show that our model does not have to be trained with respect to a specific dataset, or with outputs of a specific model. Rather, performing data augmentation by perturbing the ground truth is sufficient. We also show that our model can be extended to scene parsing for dense multi-class semantic segmentation with straightforward adaptation. Our main contributions can be summarized as:

- We propose CascadePSP, a general cascade segmentation refinement model that can refine any given input segmentations, boosting the performance of state-of-the-art segmentation models without finetuning.
- We further show that our method can be used to produce high-quality and very high-resolution segmentations which has never been achieved by previous deep learning based methods.
- We introduce the BIG dataset that can be used as an accurate evaluation dataset for very high resolution semantic image segmentation task.

2. Related Works

Semantic Segmentation Fully Convolutional Neural Networks (FCN) was first introduced in semantic segmentation in [31] which achieved remarkable progress at the time of introduction. While FCNs capture information from bottom-up, contextual information with wide field-of-view is also important for pixel labeling tasks and is exploited by many segmentation models [3, 5, 14, 17, 32, 39, 51], including image pyramid methods that use multi-scale inputs [5, 9, 14, 22, 23, 36], or feature pyramid methods that use feature maps of different receptive field sizes by spatial pooling [29, 53] or dilated convolutions with different rates [3, 4, 6, 21, 42, 49]. We choose PSPNet [53] for pyramid pooling in our network because the pertinent module is independent of input resolution, thus providing a simple yet effective method to capture contextual information even when the training and testing resolution significantly differ as in our case.

Encoder-decoder models have also been widely used in segmentation methods [1, 6, 21, 25, 27, 33, 37, 42]. They first reduce the spatial dimension to capture high-level semantics and then recover the spatial extent using a decoder. Skip connections [12, 37, 40] can be added to produce sharper boundaries which we have also employed.

Semantic segmentation models typically have a large output stride such as 4 or 8 [2, 3, 4] due to memory and computational limitation. Outputs with stride are usually bilinearly upsampled to the target size, leading to inaccurate boundary labels. Recently, the authors of [7] have proposed Global-Local Networks (GLNet) to solve this problem using a global information branch with a local fine structure network. However, they still require high-resolution training images which are not available for most tasks.

Our method adopts the encoder-decoder model to obtain better semantic and boundary information with a refinement cascade, which also helps to efficiently generate high-resolution segmentations. This formulation also makes our method highly robust and can generalize to high-resolution data without finetuning.

Refining Segmentation FCN based methods typically do not generate very high-quality segmentation. Researchers have addressed this issue with graphical models such as CRF [2, 3, 23, 20, 30, 54] or region growing [10]. They often adhere to low-level color boundaries without fully leveraging high-level semantic information and cannot fix large error regions. Propagation-based approaches [26] cannot handle very high-resolution data due to computational and memory constraints. Separate refinement modules are also used to increase boundary accuracy [35, 46, 50]. They are trained in an end-to-end fashion. Large models are prone to overfitting [50] while shallow refinement networks [35, 47] have limited refinement capability. Contrary, our method has a high model capacity and can be trained independently to repair segmentation using only objectness. Finetuning with the specific model is not required so our training is not hindered by overfitting.

Cascade Network Multi-scale analysis leverages both large and small scale features in many computer vision tasks, such as edge detection [15, 45], detection [24, 28, 41], and segmentation [7, 22, 52]. In particular, a number of methods [22, 45, 52] predict independent results at each stage and merge them to obtain multi-scale information. Our method not only fuses features from coarse scales but uses them as one of the inputs for the next finer level. We will show that adding coarse outputs as input for the next level does not change our formulation and thus the same network can be used recursively for higher resolution refinement.

3. CascadePSP

In this section, we first describe our single refinement module and then our cascade method which makes use of multiple refinement modules for high-resolution segmentation.

3.1. Refinement Module

As illustrated in Figure 2, our refinement module takes an image and multiple imperfect segmentation masks at different scales to produce a refined segmentation. Multi-scale inputs allow the model to capture different levels of structural and boundary information, which allow the network to learn to adaptively fuse the mask features from different
scales to refine the segmentation at the finest level.

All the input segmentations at lower resolution are bilinearly upsampled to the same size and concatenated with the RGB image. We extract stride 8 feature maps from the inputs using PSPNet [53] with ResNet-50 [16] as the backbone. We follow the pyramid pooling sizes of \([1, 2, 3, 6]\) as in [53] which helps to capture global context. Besides the final stride 1 output, our model also generates intermediate stride 8 and stride 4 segmentations which focus on fixing the overall structure of the input segmentation. We skip stride 2 to provide flexibility to correct local error boundary.

To reconstruct pixel-level image details that are lost in the extraction process, we employ skip-connection from the backbone network and fuse the features using an upsampling block. We concatenate the skip connected features and the bilinearly upsampled features from the main branch, and process them with two ResNet blocks. A segmentation output is generated using a 2-layer \(1 \times 1\) conv followed by a sigmoid activation.

**Loss** We produce the best result using cross-entropy loss for the coarser stride 8 output, \(L_{1+L2}\) loss for the finer stride 1 output, and the average of cross-entropy and \(L_{1+L2}\) loss for the intermediate stride 4 output. Different loss functions are applied for different strides because the coarse refinement focuses on the global structure while ignoring local details, while the finest refinement aims to achieve pixel-wise accuracy by relying on local cues. To encourage better boundary refinement, \(L1\) loss on segmentation gradient magnitude is also employed on the stride 1 output. The segmentation gradient is estimated by a \(3 \times 3\) mean filter followed by a Sobel operator [18]. The gradient loss makes outputs adhere better to the object boundary at the pixel level. As gradient is sparser compared to pixel level loss, we weigh it with \(\alpha\), which is set to 5 in our experiments. The gradient loss can be written as:

\[
\mathcal{L}_{\text{grad}} = \alpha \cdot \frac{1}{n} \sum_{i} \| \nabla(f_{m}(x_{i})) - \nabla(f_{m}(y_{i})) \|_{1}
\]

where \(f_{m}(\cdot)\) denotes the \(3 \times 3\) mean filter, \(\nabla\) denotes the gradient operator approximated by a Sobel operator, \(n\) is the total number of pixels, \(x_{i}\) and \(y_{i}\) are the \(i\)th pixel of the ground truth segmentation and output segmentation respectively. Our final loss can be written as:

\[
\mathcal{L} = \mathcal{L}_{CE}^{8} + \frac{1}{2}(\mathcal{L}_{1+L2}^{4} + \mathcal{L}_{CE}^{4}) + \mathcal{L}_{L1+L2}^{1} + \mathcal{L}_{\text{grad}}^{1}
\]

where \(\mathcal{L}_{CE}^{8}, \mathcal{L}_{1+L2}^{4}\), and \(\mathcal{L}_{\text{grad}}^{1}\) denote cross-entropy loss, \(L1+L2\) loss, and gradient loss for output stride \(s\) respectively.

**Ablation Study of Refinement Module** We evaluate our method using standard segmentation metric IoU. To highlight the perceptual importance of boundary accuracy, we propose a new **mean Boundary Accuracy** measure (mBA). For a robust estimation for images of different sizes, we sample 5 radii in \([3, \frac{W}{80}]\) with uniform intervals, compute the segmentation accuracy within each radius from the ground truth boundary, then average these values. Here we perform ablation studies to show the efficacy of our cascade design and loss function. Table 1 shows that our model produces the most significant improvement in IoU and even more significantly in boundary accuracy.

With a multi-level cascade, the module can delegate different stages of refinement to different scales. As shown in Figure 3, the 3-level model uses intermediate small-scale segmentations (will be detailed in Section 3.2) to better capture object structure. Although both models have the same
Figure 4. **Global step** refines the whole image using the same refinement module (RM) to perform a 3-level cascade with output strides (OS) of 8, 4, and 1. The cascade is jointly optimized, capturing object structure at large output strides and accurate boundary at small output strides (i.e., with a higher resolution). The receptive field, the 3-level model can better leverage structural cues to produce a more detailed segmentation than the 1-level model.

### 3.2. Global and Local Cascade Refinement

In testing, we use the **Global step** and the **Local step** to perform high-resolution segmentation refinement by employing the same trained refinement module. Specifically, the Global step considers the whole resized image to repair structure while the Local step refines details in full resolution using image crops. The same refinement module can be used recursively for higher resolution refinement.

#### 3.2.1 Global Step

Figure 4 details the design of the Global step which refines the whole image with a 3-level cascade. As the full-resolution image during testing often cannot be fit into the GPU for processing, we downsample the input such that the long-axis has length $L$ while maintaining the same aspect ratio. Inputs to the cascade are initialized with the input segmentation, which is replicated to keep the input channel dimension constant. After the first level of the cascade, one of the input channels will be replaced with the bilinearly upsampled coarse output. This is repeated until the last level, where the input consists of both the initial segmentation and all outputs from previous levels.

This design enables our network to fix segmentation errors progressively while keeping details present in the initial segmentation. With multiple levels, we can roughly delineate the object and fix larger error in coarse levels, and focus on boundary accuracy in fine levels using more robust features provided by the coarse levels.

Figure 5 illustrates the details of the Local step. Very high-resolution images cannot be processed in a single pass even with modern GPUs due to the memory constraint. Also, the drastic change of scale between training and testing data will cause poor segmentation quality. We leverage our cascade model to first perform global refinement using a downsampled image, and then perform local refinement using image crops from a higher resolution image. These crops enable the Local step to handle high-resolution images without high-resolution training data while taking image context into account due to the Global step.

During the Local step, the model takes the two outputs of the last level of the Global step, denoted as $S^2_4$ and $S^2_1$. Both outputs are bilinearly resized to the original size of the image $W \times H$. The model takes image crops of size $L \times L$ and 16 pixels will be chipped away from each side of the crop output to avoid boundary artifacts, with exceptions at the image border. The crops are taken uniformly with a stride of $L/2 - 32$ such that most pixels are covered by four crops, and invalid crops that go beyond image borders are shifted to align with the last row/column of the image. The image crops are then fed into a 2-level cascade with output stride of 4 and 1 respectively. In fusion, the outputs from different patches might disagree with each other due to different image context, and we resolve this by averaging all the output values. For images with even higher resolution, we can apply the local step recursively in a coarse-to-fine manner.
3.2.3 Choosing L

Figure 6 presents the relationship between GPU memory usage and refinement quality (mBA) during testing when different L is chosen. We have chosen \( L = 900 \) with 3.16 GB of GPU memory usage in our experiments to balance the tradeoff between increasing GPU memory usage and diminishing performance gain. Using an even higher L is unnecessary and occupies extra memory in our experiments on the BIG validation set. In low-memory settings, a smaller L such as 500 can be used to produce a slightly worse \((-0.6\%\) mBA\) refinement with a much lower memory footprint (1.16 GB). Note that the GPU memory usage only relates to L but not the image resolution as the fusion step can be easily performed on the CPU.

3.2.4 Ablation Study for Global and Local Refinement

Table 2 shows that both the Global step and the Local step are essential to high-resolution segmentation refinement. Note that the IoU drop is much more significant when we remove the Global step, indicating that the Global step is mainly responsible for fixing the overall structure contributing more to IoU boost while the Local step alone cannot achieve due to insufficient image context. Without the Local step, although IoU only decreases slightly, we note that boundary accuracy decreases more significantly since the Global step cannot extract high-resolution details.

Figure 7 studies the importance of the Local step for different resolution inputs: we evaluate our method with and without the Local step in various-sized segmentations generated by resizing the BIG validation set. While the Global step is sufficient for low-resolution inputs, the Local step is crucial for accurate high-resolution refinement with size higher than the switching point 900. We therefore use both the Global and Local step for inputs with \( \text{max}(H, W) \geq 900 \), and only the Global step for lower resolution inputs.

![Figure 6. Relationship between the choice of L and mBA in the BIG validation set. With increasing L, GPU memory usage increases with diminishing performance gain.](image)

![Figure 7. We evaluated our method across different input resolutions. The Global step does not benefit from higher resolution inputs because it is bounded by \( L = 900 \). The Local step is crucial for high-resolution refinement to handle inputs with a larger size than L with bounded memory cost.](image)

### Table 2

<table>
<thead>
<tr>
<th>Configuration</th>
<th>BIG</th>
<th>mBA (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deeplab V3+</td>
<td>89.65</td>
<td>60.94</td>
</tr>
<tr>
<td>Global step only</td>
<td>91.86</td>
<td>73.10</td>
</tr>
<tr>
<td>Local step only</td>
<td>91.35</td>
<td>73.06</td>
</tr>
<tr>
<td>Both steps (Ours)</td>
<td>92.01</td>
<td>75.59</td>
</tr>
</tbody>
</table>

Table 2. Ablation experiments for the Global and Local step. Input segmentations are taken from Deeplab V3+ on the BIG validation set. Using both steps show the best results.

3.3. Training

To learn objectness information, we train our model on a collection of datasets in a class-agnostic manner. We merge MSRA-10K [8], DUT-OMRON [48], ECSSD [38], and FSS-1000 [44] to generate a segmentation dataset of 36,572 with much more diverse semantic classes than common datasets such as PASCAL (20 classes) or COCO (80 classes). Using this dataset (> 1000 classes) makes our model more robust and generalizable to new classes.

During training, we take random \( 224 \times 224 \) image crops and generate input segmentations by perturbing the ground truth. The inputs go through a 3-level cascade as in the Global step with the loss computed in every level. Although the crop size is smaller than \( L \) which is used in testing, our model design helps bridging this gap. The fully convolutional feature extractor provides translational invariance while the pyramid pooling module provides important image context, allowing our model to be extended to higher resolution without significant performance loss. The use of smaller crop speeds up our training process and makes data preparation much easier as high-resolution training data for segmentation is expensive to obtain.

For generalizability, we avoid training using segmentation outputs generated by existing models which can lead to overfitting to that specific model. Instead, perturbed ground truth should portray various shapes and output of inaccurate segmentations produced by other methods, which helps our algorithm to be more robust to different initial segmentations. We generate such perturbed segmentations by subsampling the contour followed by random dilations and erosions. Examples of such perturbation are shown in Figure 8.

4. Experiments

In this section, we quantitatively evaluate our results with PASCAL VOC 2012 [13], BIG (our high-resolution data set), and ADE20K [55]. We evaluate our model without any finetuning in various settings and show the improvements made by our model.

4.1. Dataset and Evaluation Method

Although widely used in image segmentation tasks, PASCAL VOC 2012 dataset does not have pixel-perfect
4.2. Implementation Details

We implement our model with PyTorch [34]. We use PSPNet with ResNet-50 backbone [53] as our base network. Data augmentations, including perturbation of ground truths, image flipping and cropping are done on-the-fly to further increase data variety. We use Adam optimizer [19] with a weight decay of $10^{-4}$, learning rate of $3 \times 10^{-4}$ for 30K iterations followed by a learning rate of $3 \times 10^{-5}$ for another 30K iterations with a batch size of 9. The total training time is around 16h with two 1080Ti. The Local step is only performed in the region of interest, and the complete refinement process takes about 6.6s for Figure 1. Unless otherwise specified, we use the same trained model for all the experiments.

4.3. Segmentation Input

Our method can refine input segmentations using only objectness information. Note that our model has never seen any of the following datasets in training. In this section, we focus on the refinement effect of individual objects, meaning that class competition is not introduced.

Here, we compare and evaluate the effect of our refinement model on the output of various semantic segmentation models [6, 53] trained on the PASCAL VOC 2012 dataset. Our method is more effective than commonly used multi-scale testing, with experimental results further shown in the supplementary material.

PASCAL VOC 2012 As the input models are trained in the PASCAL VOC 2012 dataset, resizing is not needed to obtain their outputs which are then fed into our refinement model. These images are of low resolution, so we can refine them directly using the Global step only. We report the overall class-agnostic IoU and boundary accuracy in the upper half of Table 3. Results show that our method can improve segmentation quality in all cases, especially along the boundary region.

<table>
<thead>
<tr>
<th>Methods</th>
<th>IoU (%)</th>
<th>mBA (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCN-8s [31]</td>
<td>68.85</td>
<td>54.05</td>
</tr>
<tr>
<td>(+) Ours</td>
<td>72.70↑</td>
<td>65.36↑</td>
</tr>
<tr>
<td>RefineNet [22]</td>
<td>86.21</td>
<td>62.61</td>
</tr>
<tr>
<td>(+) Ours</td>
<td>87.48↑</td>
<td>71.34↑</td>
</tr>
<tr>
<td>DeepLabV3+ [6]</td>
<td>87.13</td>
<td>61.68</td>
</tr>
<tr>
<td>(+) Ours</td>
<td>89.01↑</td>
<td>72.10↑</td>
</tr>
<tr>
<td>PSPNet [53]</td>
<td>90.92</td>
<td>60.51</td>
</tr>
<tr>
<td>(+) Ours</td>
<td>92.86↑</td>
<td>72.24↑</td>
</tr>
</tbody>
</table>

Table 3. Comparison between different semantic segmentation methods with and without our refinement. Their results are produced using their respective official implementations with the best provided model. Low-resolution outputs from the original model are bicubic-upsampled to the original resolution for evaluation.

BIG dataset Most existing segmentation methods cannot be directly evaluated on the full-resolution BIG dataset,
due to the memory constraint. Therefore, we obtained initial segmentations by feeding resized images to the existing models. We downsampled the input image such that the long-axis is 512-pixel while maintaining the aspect ratio, and bicubic-upsampled the output segmentation to the original resolution.

In the lower half of Table 3, we show our results on the BIG test set with high-resolution segmentation. Note that even we have never seen any high-resolution training images, we are able to produce high-quality refinements at these scales. Figure 14 shows the visual improvement of our refinement. Although super-resolution models [11, 43] may seem plausible for upsampling segmentation masks, inputs with erroneous segmentation (e.g. the missing table leg in Figure 13 and the baby’s hand in Figure 14) cannot be corrected by super-resolution.

Our method relies on the input segmentation and low-level cues and does not have the specific semantic capability. Figure 11 shows one failure case where the input error is too large for our method to eliminate.

4.4. Scene parsing

To extend CascadePSP to scene parsing, in the presence of dense classes where class competition may be problematic, we propose a divide-and-conquer approach to independently refine each semantic object using our pretrained network, followed by integrating the results using a fusion function. Figure 12 overviews our strategy.

We refine sufficiently large connected components for each semantic object independently by taking ROIs with 25% padding. To handle overlapping regions, the naïve approach would be to use argmax on the output confidence which would lead to noisy results in regions where all the classes have low scores. Instead, our fusion function is a modified argmax where if all the input class confidence have values lower than 0.5, we fall back to the original segmentation.

Here, we evaluate our model on the validation set of ADE20K [55]. As the ADE20K dataset contains “stuff” background classes (see supplementary material) that are not strong in objectness and too different from our training data, we have attenuated their output scores to focus on foreground refinement. Note that refining the foreground objects can still help with background refinement since the argmax operation takes both confidence scores into consideration. Table 4 tabulates the results which show that our model produces higher quality segmentation. Figure 13 shows sample qualitative evaluation.

<table>
<thead>
<tr>
<th>Methods</th>
<th>mIoU (%)</th>
<th>mBA (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADE20K</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RefineNet [22] (+) Ours</td>
<td>42.20±0.73</td>
<td>56.07±1.07</td>
</tr>
<tr>
<td>EncNet [51]  (+) Ours</td>
<td>43.19±0.99</td>
<td>57.29±2.00</td>
</tr>
<tr>
<td>PSPNet [53]  (+) Ours</td>
<td>43.83±0.73</td>
<td>58.13±1.10</td>
</tr>
</tbody>
</table>

Table 4. Comparison between different methods with and without our refinement on the ADE20K validation set.

5. Conclusion

We propose CascadePSP, a general segmentation refinement framework for refining any input segmentations and achieve a higher accuracy without any finetuning afterward. CascadePSP performs high-resolution (up to 4K) segmentation refinement even our model has never seen any high-resolution training images. With a single refinement module trained on low-resolution data without any finetuning, the proposed Global step refines the entire image and provides sufficient image context for the subsequent Local step to perform full-resolution high-quality refinement. We hope this work can contribute to more high-resolution computer vision tasks in the future.

Acknowledgements We thank Gary Jing Yang Zhang for fruitful discussion during his exchange semester at HKUST.
Figure 14. Visual comparison on the BIG test set. Odd rows show the whole image and even rows show the zoomed-in crop. Inputs are from DeeplabV3+, RefineNet, PSPNet, and FCN-8s, top to bottom.
[38] Jianping Shi, Qiong Yan, Li Xu, and Jiaya Jia. Hierarchical image saliency detection on extended cssd. *TPAMI*, 2015.


