# Look Closer to Segment Better: Boundary Patch Refinement for Instance Segmentation

Chufeng Tang[1*] Hang Chen[1*] Xiao Li[1] Jianmin Li[1] Zhaoxiang Zhang[2] Xiaolin Hu[1†]
[1]State Key Laboratory of Intelligent Technology and Systems, THU-Bosch JCML Center,
BNRist, Institute for AI, Department of Computer Science and Technology, Tsinghua University
[2]Institute of Automation, CAS & University of Chinese Academy of Sciences &
Centre for Artificial Intelligence and Robotics, HKISI_CAS
{tcf18, chenhang20, lixiao20}@mails.tsinghua.edu.cn zhaoxiang.zhang@ia.ac.cn
{lijianmin, xlhu}@mail.tsinghua.edu.cn

## Abstract

*Tremendous efforts have been made on instance segmentation but the mask quality is still not satisfactory. The boundaries of predicted instance masks are usually imprecise due to the low spatial resolution of feature maps and the imbalance problem caused by the extremely low proportion of boundary pixels. To address these issues, we propose a conceptually simple yet effective post-processing refinement framework to improve the boundary quality based on the results of any instance segmentation model, termed BPR. Following the idea of looking closer to segment boundaries better, we extract and refine a series of small boundary patches along the predicted instance boundaries. The refinement is accomplished by a boundary patch refinement network at higher resolution. The proposed BPR framework yields significant improvements over the Mask R-CNN baseline on Cityscapes benchmark, especially on the boundary-aware metrics. Moreover, by applying the BPR framework to the "PolyTransform + SegFix" baseline, we reached* $1^{st}$ *place on the Cityscapes leaderboard. Code is available at* https://github.com/tinyalpha/BPR.

## 1. Introduction

Instance segmentation, which aims to assign a pixel-wise instance mask with a category label to each object in an image, has great potential in various computer vision applications, such as autonomous driving and robotics. Mask R-CNN [13] is a prevailing two-stage instance segmentation framework, which first employs a Faster R-CNN [32] detector to detect objects in an image and further performs binary segmentation within each detected bounding box. Other methods [14, 25] built upon Mask R-CNN consistently achieve superior performance. Driven by the recent development of one-stage detectors [22, 37, 53], a number of one-stage instance segmentation frameworks [2, 3, 6, 19, 36, 40, 41, 42, 46, 51] have been proposed.

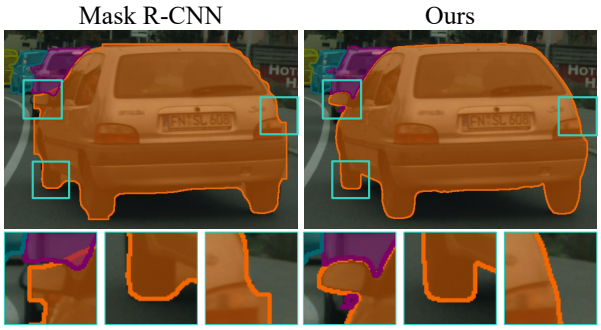


Figure 1: **Left**: Instance segmentation results and the extracted boundary patches of Mask R-CNN. **Right**: After the refinement of our BPR framework, the instance mask aligns better with object boundaries. Best viewed with zoom-in.

However, the quality of the predicted instance mask is still not satisfactory. One of the most important problems is the imprecise segmentation around instance boundaries. As shown in Figure 1(left), the predicted instance masks of Mask R-CNN are coarse and not well-aligned with the real object boundaries. Empirically, correcting the error pixels near object boundaries can improve the mask quality a lot. We conducted an upper bound analysis in Table 1. A large gain (9.4/14.2/17.8 in AP) can be obtained by simply replacing the predictions with ground-truth labels for pixels within a certain Euclidean distance (1px/2px/3px) to the predicted boundaries, especially for small objects.

*Equal contribution.
†Corresponding author.

We argue that there are two critical issues leading to low-quality boundary segmentation. (1) The low spatial resolution of the output, *e.g*. 28×28 in Mask R-CNN or at most 1/4 input resolution in some one-stage frameworks [36, 41], makes finer details around object boundaries disappear. The predicted boundaries are always coarse and imprecise (see Figure 1 and 4). (2) Pixels around object boundaries only make up a small fraction of the whole image (less than 1% [16]), and are inherently hard to classify. Treating all pixels equally may leads to an optimization bias towards smooth interior areas, while underestimating the boundary pixels.

As a long-standing challenge in dense prediction tasks, many studies have attempted to improve the boundary quality, while the above issues are still not well solved. For example, BMask R-CNN [7] and Gated-SCNN [35] employ an extra branch to enhance the boundary awareness of mask features, which can fix the optimization bias to some extent, while the low resolution issue remains unsolved. PolyTransform [21] and SegFix [48] act as a post-processing scheme to improve the boundary quality. PolyTransform [21] employs a deforming network with the cropped instance patch to predict the offsets of polygon vertices, while suffering from a large computational overhead. SegFix [48] replaces the coarse predictions of boundary pixels with interior predictions, but it relies on precise boundary predictions. We argue that the instance boundary prediction task shares a similar complexity with instance segmentation.

Considering the human annotation behavior for instance segmentation, the annotators usually first localize and categorize each object in the given image, and then explicitly or implicitly segment some coarse instance masks at a low resolution. Afterwards, to obtain a high-quality mask, the annotators need to repeatedly zoom into the local boundary regions and explore the sharper boundary segmentation at higher resolution. Intuitively, high-level semantics are required to localize and roughly segment objects, while low-level details (*e.g*. colour consistency and contrast) are more critical for segmenting the local boundary regions.

In this paper, motivated by the human segmentation behavior, we propose a conceptually simple yet effective post-processing framework to improve the boundary quality through a *crop-then-refine* strategy. Specifically, given a coarse instance mask produced by any instance segmentation model, we first extract a series of small image patches along the predicted instance boundaries. After concatenated with mask patches, the boundary patches are fed into a refinement network, which performs binary segmentation to refine the coarse boundaries. The refined mask patches are then reassembled into a compact and high-quality instance mask, shown in Figure 1(right). We termed the proposed framework as **BPR** (**B**oundary **P**atch **R**efinement). The proposed framework can alleviate the aforementioned issues and improve the mask quality without any modification or fine-tuning to the segmentation models. Since we only crop around object boundaries, the patches are allowed to be processed with the much higher resolution than previous methods, so that low-level details can be retained better. Concurrently, the fraction of boundary pixels in the small patch is naturally increased, which can alleviate the optimization bias. The proposed BPR framework significantly improves the results of Mask R-CNN baseline (+4.3% AP on Cityscapes dataset), and produces substantially better masks with finer boundaries. We found that the model trained on the results of Mask R-CNN can be easily transferred to refine the results of other instance segmentation models as well, without the need for re-training. We outperform some boundary refinement methods [17, 48] and show that these methods are complementary by successfully transferring our model to improve their results. Furthermore, by applying our BPR framework to the "PolyTransform + SegFix" baseline [48], we established a new state-of-the-art on the Cityscapes test set with AP of 42.7%, and ranked $1^{st}$ place on the Cityscapes leaderboard by the CVPR 2021 submission deadline.

| Dist. | AP | $AP_{50}$ | $AP_{75}$ | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|
| - | 36.4 | 60.8 | 36.9 | 11.1 | 32.4 | 57.3 |
| 1px | 45.8 | 64.8 | 49.3 | 21.1 | 42.6 | 63.5 |
| 2px | 50.6 | 66.5 | 54.6 | 26.3 | 47.0 | 66.8 |
| 3px | 54.2 | 67.5 | 58.5 | 30.4 | 50.7 | 69.3 |
| $\infty$ | 70.4 | 70.4 | 70.4 | 41.5 | 66.7 | 88.3 |

Table 1: A large gain can be obtained by replacing the predictions for pixels within a certain Euclidean distance to the predicted boundaries with their group-truth labels. $\infty$ means all error pixels are corrected. Experiments were conducted with Mask R-CNN as baseline on Cityscapes val set.

## 2. Related Work

**Instance Segmentation**. Recent studies on instance segmentation can be divided into two categories: two-stage and one-stage methods, as briefly reviewed below.

Two-stage methods usually follow the classical *detect-then-segment* strategy. The dominant method is still Mask R-CNN [13], which inherits from the two-stage detector Faster R-CNN [32] to first detect objects in an image and further performs binary segmentation within each detected bounding box. Following Mask R-CNN, PANet [25] enhances feature representation through bottom-up path augmentation. Mask Scoring R-CNN [14] adds an additional mask-IoU head to re-score the mask predictions. These methods consistently achieve superior performance.

One-stage methods recently attracts more attention due to the rapid development of one-stage detectors [22, 37, 53]. Some methods [2, 3, 19, 46, 51] continue to adapt the

*detect-then-segment* strategy but replace the detectors with the one-stage alternatives. YOLACT [2] achieves real-time speed by learning a set of prototypes and the prototypes are assembled with the learned linear coefficients. BlendMask [3] further improves this idea by assembling with attention maps. Some recent proposed methods [6, 36, 40, 41] eliminate the need for detection by directly segmenting objects in a location-wise manner. CondInst [36] and SOLOv2 [41] achieve remarkable performance with high efficiency. In addition, there are some approaches [9, 11, 30] built upon the semantic segmentation models, which usually learn the pixel-wise embeddings and then cluster them into instances. Several works [1, 31, 42, 45] replace the pixel-wise instance representation into the contour-based representation.

Our proposed framework is agnostic to the instance segmentation methods, thus it can be applied to refine the results of any instance segmentation model, both one-stage and two-stage methods.

**Semantic Segmentation**. Modern semantic segmentation approaches are pioneered by fully convolutional networks (FCNs) [27]. Many studies have been proposed on this foundation to improve the segmentation results, such as increasing the resolution of feature maps with dilated/atrous convolutions [4, 5], enriching context information [10, 47, 50, 52], using an encoder-decoder architecture [5, 15, 28, 33], or some refinement schemes [18, 20, 48]. Minaee *et al*. [29] provided a comprehensive review of these approaches. In this paper, we adopt the prevailing HRNet [39] in our framework, which can maintain high-resolution representation throughout the whole network.

**Boundary Refinement for Segmentation**. Most recent studies focused on boundary refinement aim at designing a boundary-aware segmentation model by integrating an extra and specialized module to process boundaries. For example, BMask R-CNN [7] and Gated-SCNN [35] employ an extra branch to enhance the boundary awareness of mask features by estimating boundaries directly. PointRend [17] iteratively samples the feature points with unreliable predictions and refines them with a shared MLP. Another line of work attempts to refine the boundaries based on the results of existing segmentation models with a post-processing scheme. SegFix [48] is a general refinement mechanism, which replaces the unreliable predictions of boundary pixels with the predictions of interior pixels. The effectiveness of SegFix highly depends on the accuracy of boundary prediction. However, it is very challenging to directly estimate precise instance boundaries. Intuitively, the instance segmentation task could easily be settled if the precise boundaries are already given. Our method shares more similarities with PolyTransform [21], which transforms the contour of instance into a set of polygon vertices. A Transformer [38] based network is applied to predict the offsets of vertices towards object boundaries. It achieves superior performance while suffering from a large computational overhead due to the large instance patch and the heavy Transformer architecture. Our proposed method is also a post-processing scheme. Different from these methods, we focus on refining the boundary patches to improve the mask quality.

## 3. Framework

An overview of the proposed framework is illustrated in Figure 2. As a post-processing mechanism, the proposed framework can be applied to refine the results of any prevailing instance segmentation model, without any modification or fine-tuning to the segmentation models themselves.

### 3.1. Boundary Patch Extraction

Given an instance mask produced by an instance segmentation model, we first need to determine which part of the mask should be refined. Based on the findings of previous works [7, 48] and our verification experiments in Table 1, we propose an effective *sliding-window* style algorithm to extract a series of patches along the predicted instance boundaries. Specifically, we densely assign a group of squared bounding boxes where the central areas of the box should cover the boundary pixels, shown in Figure 2(b). The obtained boxes still contain large overlaps and redundancies, thus we further apply a Non-Maximum Suppression (NMS) algorithm to filter out a subset of patches (Figure 2c). Empirically, with the larger overlaps, the segmentation performance can be boosted, while simultaneously suffering from the larger computational cost. We can adjust the NMS threshold to control the amount of overlap to achieve a better speed/accuracy trade-off. In addition to image patches, we also extract the corresponding binary mask patches from the given instance mask. The concatenated image and mask patches (Figures 2d and 2e) are resized and fed into the following boundary patch refinement network.

### 3.2. Boundary Patch Refinement

**Mask Patch**. The benefit of the binary mask patch is that it accelerates training convergence and provides location guidance for the instance to be segmented. As discussed in the previous works on semantic segmentation [39, 47], context information plays a vital role for pixel-wise classification. Therefore, the cropped image patches are hard to be classified independently due to the limited context information. With the help of location and semantic information provided by the mask patches, the refinement network can eliminate the need for learning instance-level semantics from scratch. Instead, the refinement network only needs to learn how to locate the hard pixels around the decision boundary and push them to the correct side. We believe this goal can be achieved by exploring low-level image properties (*e.g*. colour consistency and contrast) provided in the local and high-resolution image patches. More importantly,

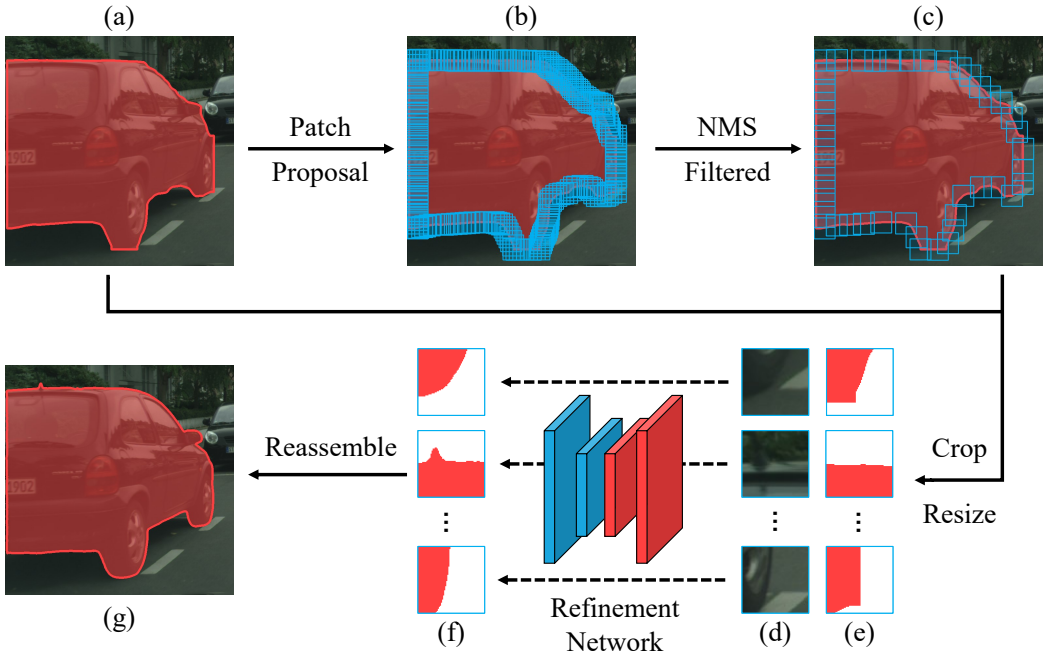


Figure 2: Overview of the proposed boundary patch refinement framework. Given a coarse mask (a) produced by an instance segmentation model, we first densely assign a series of squared bounding boxes along the predicted boundaries (b), and filter out a subset of boundary patches (c) using NMS. NMS threshold of 0.25 is used here. The extracted image patches (d) and mask patches (e) are resized and fed into the boundary patch refinement network. Mask patches after refinement (f) are reassembled into a compact and precise instance mask (g). Best viewed digitally and in colour.

the adjacent instances are likely to share an identical boundary patch, while the learning goals are totally different and ambiguous. Together with different mask patches for each instance, these issues can be avoided. As compared in Table 2, the model has trouble to converge without using the mask patches, examples of which are shown in Figure 3.

**Boundary Patch Refinement Network**. The role of this refinement network is to perform binary segmentation for each extracted boundary patch individually. Any semantic segmentation model can be employed for this task by simply modifying the input channels to 4 (3 for the RGB image patch and 1 for the binary mask patch) and output classes to 2. For the sake of convenience, we adopt the state-of-the-art HRNetV2 [39] as the refinement network in our implementation, which can maintain high-resolution representation throughout the whole network. By increasing the input size appropriately, the boundary patches can be processed with much higher resolution than in previous methods.

**Reassembling**. The refined boundary patches are reassembled into a compact instance-level mask by replacing their previous predictions. Predictions are unchanged for those pixels without refinement. For the overlapping areas of adjacent patches, the results are aggregated by simply averaging the output logits and applying a threshold of 0.5 to distinguish the foreground and background.

### 3.3. Learning and Inference

The refinement network is trained based on the boundary patches extracted from training images and tested on validation or testing images. We do not directly train or fine-tune the instance segmentation models. During training, we only extract boundary patches from instances whose predicted masks have an Intersection over Union (IoU) overlap larger than 0.5 with the ground-truth masks, while all predicted instances are retained during inference. The model outputs are supervised with the corresponding ground-truth mask patches using the pixel-wise binary cross-entropy loss. We simply fix the NMS eliminating threshold to 0.25 during training, while adopting different thresholds during inference based on the speed requirements. See *Supplementary Materials* for more implementation details.

## 4. Experiments

### 4.1. Datasets and Metrics

**Datasets**. We mainly report the results on Cityscapes [8], a real-world dataset with high-quality instance segmentation annotations. We only used the `fine` data, containing $2,975/500/1,525$ images for train/val/test, which are collected from 27 cities, with a high resolution of $1024\times2048$.

Eight instance categories are involved, including bicycle, bus, person, train, truck, motorcycle, car, and rider.

**Metrics**. The COCO-style [23] mask AP (averaged over 10 IoU thresholds ranging from 0.5 to 0.95 in the step of 0.05), $AP_{50}$ (AP at an IoU of 0.5) and $AP_S$/$AP_M$/$AP_L$ (for small/medium/large instances) were reported in most of our experiments. The official Cityscapes-style AP [8] was only used to report the final results for a fair comparison, which is slightly higher than the COCO-style AP. Similar to [21, 35, 48], we also used a boundary F-score to evaluate the quality of the predicted boundaries. A mask was considered correct if the boundary is within a certain distance threshold from the ground-truth. We used a threshold of 1px and only compute for true positives that are determined on the same 10 IoU thresholds ranging from 0.5 to 0.95. The boundary F-score was computed in an instance-wise manner and then averaged over them, termed AF.

### 4.2. Ablation Study

We investigated the effectiveness of the proposed framework through extensive ablation experiments on the configurable design choices. We started the refinement with the results of Mask R-CNN ResNet-FPN-50 baseline trained on Cityscapes `fine` data (with COCO pre-training). We adopted the lightweight HRNetV2-W18-Small as the refinement network in the default setting, with input size equal to 128×128. The boundary patches were extracted with patch size equal to 64×64 without padding, and the inference NMS threshold was set to 0.25 by default.

**Effects of Mask Patch**. To validate the effect of mask patch for boundary refinement, we made a comparison by eliminating the mask patches while keeping other settings unchanged. As indicated in Table 2, the model trained with image patches solely yielded a terrible result, even worse than the segmentation results before refinement. However, together with mask patches, we achieved a significant improvement (+3.4% in AP, +11.9% in AF) by refining the Mask R-CNN segmentation results. We also show some patch-wise examples in Figure 3. For a simple case with one dominant instance in the image patch (first row), both of the models (w/ or w/o mask patch) produced reasonable results. However, as for cases with multiple instances crowded in the image patch, the model without mask patch (last column) failed to distinguish which object should be segmented, leading to coarse (4th row) or completely wrong (2nd and 3rd rows) predictions. In contrast, with the help of mask patches, we produced high-quality predictions with accurate and distinct boundaries (3rd column).

**Patch Size**. We increased the boundary patch size by cropping with a larger box and/or with padding. Note that the padded areas were only used to enrich the context and not used for reassembling. As the patch size gets larger, the model becomes less focused but can access more context information. In Table 3, we compared various choices and found that the 64×64 patch without padding works better. We used this setting in all experiments.

| w/ mask | AP | $AP_{50}$ | AF | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|
| – | 36.4 | 60.8 | 54.9 | 11.1 | 32.4 | 57.3 |
| ✘ | 20.1 | 42.2 | 57.2 | 4.0 | 14.7 | 36.3 |
| ✔ | **39.8** | **62.0** | **66.8** | **12.7** | **35.9** | **62.2** |

Table 2: **Effects of mask patch**: A dramatic performance drop can be observed without the use of mask patch. "–" indicates the results of Mask R-CNN before refinement.


Figure 3: Boundary patch examples of (from left to right): ground-truth, predictions of Mask R-CNN, results refined by our proposed framework, results without the use of mask patch. The mask patch plays a crucial role in our framework, resulting in high-quality boundaries (the 3rd column).

| scale/pad | AP | $AP_{50}$ | AF | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|
| – | 36.4 | 60.8 | 54.9 | 11.1 | 32.4 | 57.3 |
| 32 / 0 | 39.4 | 62.0 | 66.8 | 12.6 | 35.6 | 61.4 |
| 32 / 5 | 39.7 | **62.2** | **67.6** | **12.9** | **35.9** | 61.6 |
| 64 / 0 | **39.8** | 62.0 | 66.8 | 12.7 | **35.9** | 62.2 |
| 64 / 5 | 39.7 | 61.7 | 66.5 | 12.5 | 35.8 | 62.1 |
| 96 / 0 | 39.6 | 62.0 | 65.7 | 12.2 | 35.4 | **62.3** |

Table 3: **Results of different patch size**. The patch size of 64×64 without padding works better.

**Different Patch Extraction Schemes**. The most important contribution of this work is the idea of looking closer at instance boundaries to achieve better segmentation results.

| scheme | size | AP | $AP_{50}$ | AF |
|---|---|---|---|---|
| – | – | 36.4 | 60.8 | 54.9 |
| dense sampling + NMS | 64 | **39.8** | **62.0** | **66.8** |
| pre-defined grid | 32 | 39.3 | 61.8 | 65.8 |
| pre-defined grid | 64 | 39.1 | 61.9 | 65.6 |
| pre-defined grid | 96 | 38.8 | 61.6 | 63.7 |
| instance-level patch | 256 | 37.5 | 61.1 | 61.5 |
| instance-level patch | 512 | 38.7 | 61.6 | 63.8 |

Table 4: **Different patch extraction schemes**: The "dense sampling + NMS filtering" scheme works better.

| size | FPS | AP | AF | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|
| – | – | 36.4 | 54.9 | 11.1 | 32.4 | 57.3 |
| 64 | 17.5 | 39.1 | 64.9 | 11.8 | 35.1 | 61.6 |
| 128 | 9.4 | 39.8 | 66.8 | 12.7 | **35.9** | 62.2 |
| 256 | 4.1 | **40.0** | **67.0** | **12.8** | **35.9** | **62.5** |
| 512 | <2 | 39.7 | 66.9 | 12.7 | 35.7 | 61.9 |

Table 5: **Input size of the refinement network**: Better performance is achieved with input size of 256×256.

| Net | FPS | AP | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|
| – | – | 36.4 | 11.1 | 32.4 | 57.3 |
| HRNet-W18s | 9.4 | 39.8 | 12.7 | 35.9 | 62.2 |
| HRNet-W18 | 5.8 | 39.8 | 12.6 | 35.8 | 62.1 |
| HRNet-W48 | 2.5 | 40.1 | **12.9** | 36.2 | 62.1 |

Table 6: **Alternatives of the refinement network**: Stronger segmentation backbones lead to better results.

| thr. | #patch/img | AP | $AP_{50}$ | AF |
|---|---|---|---|---|
| – | – | 36.4 | 60.8 | 54.9 |
| 0 | 32 | 37.7 | 61.5 | 58.7 |
| 0.15 | 103 | 39.6 | 61.9 | 66.0 |
| 0.25 | 135 | 39.8 | 62.0 | 66.8 |
| 0.35 | 178 | 39.9 | 62.0 | 67.0 |
| 0.45 | 241 | 40.0 | 62.0 | 67.0 |
| 0.55 | 332 | **40.1** | 62.0 | 67.1 |
| 0.65 | 485 | **40.1** | 62.0 | **67.2** |

Table 7: **NMS eliminating threshold**: We achieved consistent gains with the larger thresholds, saturating around 0.55. The average number of patches per image is also listed.

There are multiple choices about how to extract the boundary patches for refinement. We compared three extraction schemes and listed the results in Table 4. The most straightforward scheme is dividing the input image into a group of patches according to the pre-defined grids, and then picking only the patches that covering the predicted boundaries. We varied the patch size and found the results were consistently worse than our proposed "dense sampling + NMS filtering" scheme. One of the most important reasons is the imbalanced foreground/background ratio. We observed that some extracted patches are almost entirely filled with either foreground or background pixels. These patches are hard to refine due to the lack of context, thus leading to sub-optimal results. In contrast, by restricting the center of patches to cover the boundary pixels, the imbalance problem can be alleviated. Another scheme, similar to some previous works [21, 26], is cropping the *whole* instance based on the detected bounding box and further re-segmenting the instance patch. As shown, even though the input size was increased to 512×512, the results are still sub-optimal, which demonstrated the effectiveness of our *local* boundary patches. See *Supplementary Materials* for detailed descriptions.

**Input Size of the Refinement Network**. The extracted patches are resized into a larger scale before refinement. Table 5 shows the impact of input size. We also report the approximate inference speed of the refinement network, with a fixed batch size of 135 (on average 135 patches per image). As the input size increases, the AP and AF scores increase accordingly, and slightly drop after 256. The boundaries can be processed with the higher resolution with the larger input size, thus more details can be retained.

**Alternatives of refinement network**. We compared different backbones for our refinement network in Table 6. As shown, a stronger backbone usually lead to higher performance, but at the expense of lower speed. Since the model essentially performs binary segmentation for patches, it can further benefit from the advances in semantic segmentation, such as increasing the resolution of feature maps [4, 5, 39] and more effective backbones [43, 49].

**NMS Eliminating Threshold**. We studied the impact of different NMS eliminating thresholds during inference, shown in Table 7. As the threshold gets larger, the number of boundary patches increases rapidly. The overlap of adjacent patches provides a chance to correct unreliable predictions of the inferior patches. As shown, the resulting boundary quality was consistently improved with a larger threshold, and reached saturation around 0.55. We believe a better speed/accuracy trade-off can be achieved by setting a proper threshold.

### 4.3. Transferability

What the BPR model learned is a general ability to correct error pixels around instance boundaries. We can easily transfer this **ability of boundary refinement** to refine the results of any instance segmentation model. Specifically, once we get a model trained on the boundary patches extracted from the train-set predictions of Mask R-CNN on Cityscapes, we can make inference to refine any predictions (on Cityscapes train/val/test sets) produced by any models (not only Mask R-CNN), without the need for training from scratch. After training, the BPR model becomes model-agnostic, similar to SegFix [48]. We validated the transferability by applying the model trained on Mask R-CNN

| | training data | $AP_{val}$ | AP | $AP_{50}$ | person | rider | car | truck | bus | train | mcycle | bicycle |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SGN [24] | fine + coarse | 29.2 | 25.0 | 44.9 | 21.8 | 20.1 | 39.4 | 24.8 | 33.2 | 30.8 | 17.7 | 12.4 |
| Mask R-CNN [13] | fine | 31.5 | 26.2 | 49.9 | 30.5 | 23.7 | 46.9 | 22.8 | 32.2 | 18.6 | 19.1 | 16.0 |
| BMask R-CNN [7] | fine | 35.0 | 29.4 | 54.7 | 34.3 | 25.6 | 52.6 | 24.2 | 35.1 | 24.5 | 21.4 | 17.1 |
| AdaptIS [34] | fine | 36.3 | 32.5 | 52.5 | 31.4 | 29.1 | 50.0 | 31.6 | 41.7 | 39.4 | 24.7 | 12.1 |
| PANet [25] | fine | 36.5 | 31.8 | 57.1 | 36.8 | 30.4 | 54.8 | 27.0 | 36.3 | 25.5 | 22.6 | 20.8 |
| SSAP [11] | fine | 37.3 | 32.7 | 51.8 | 35.4 | 25.5 | 55.9 | 33.2 | 43.9 | 31.9 | 19.5 | 16.2 |
| UPSNet [44] | fine + COCO | 37.8 | 33.0 | 59.7 | 35.9 | 27.4 | 51.9 | 31.8 | 43.1 | 31.4 | 23.8 | 19.1 |
| PANet [25] | fine + COCO | 41.4 | 36.4 | 63.1 | 41.5 | 33.6 | 58.2 | 31.8 | 45.3 | 28.7 | 28.2 | 24.1 |
| Mask R-CNN* [13] | fine + COCO | 36.8 | 32.6 | 59.2 | 36.7 | 29.2 | 52.8 | 30.0 | 40.3 | 27.9 | 25.0 | 19.0 |
| + SegFix* [48] | | 38.2 | 33.3 | 57.8 | 37.9 | 30.3 | 54.1 | 31.0 | 40.0 | 27.9 | 25.1 | 20.5 |
| + BPR | | 41.1 | 36.9 | 61.0 | 42.0 | 33.3 | 59.9 | 32.9 | 44.4 | 32.6 | 28.0 | 22.3 |
| + SegFix + BPR | | 40.9 | 36.8 | 59.8 | 41.0 | 32.8 | 58.7 | 32.9 | 43.1 | 36.8 | 26.5 | 22.2 |
| PolyTransform [21] | fine + COCO | 44.6 | 40.1 | 65.9 | 42.4 | 34.8 | 58.5 | 39.8 | 50.0 | 41.3 | 30.9 | 23.4 |
| + SegFix [48] | | - | 41.2 | 66.1 | 44.3 | 35.9 | 60.5 | 40.5 | 51.2 | 41.6 | 31.7 | 24.1 |
| + BPR† | | **46.9** | 42.4 | **66.6** | 45.6 | 36.7 | 62.4 | 41.2 | 52.3 | 43.4 | **32.7** | **25.2** |
| + SegFix + BPR† | | - | **42.7** | 66.5 | **46.0** | **37.1** | **62.8** | **41.3** | **52.7** | **43.7** | 32.6 | 25.1 |

Table 8: **Results on Cityscapes val ($AP_{val}$ column) and test (remaining columns) sets.** We used BPR to denote our framework. BPR† indicates that the BPR trained on the results of Mask R-CNN* was transferred to another model. Mask R-CNN* is on our implementation, slightly higher than [13]. SegFix* used their own Mask R-CNN baseline (36.5/32.0 in AP val/test), slightly lower than ours. We established the new state-of-the-art results on Cityscapes `val` and `test` sets.

| | AP | $AP_{50}$ | AF |
|---|---|---|---|
| PointRend [17] | 35.6 | 60.6 | 58.0 |
| w/ BPR† | 38.6 | 62.4 | 66.5 |
| Mask R-CNN + SegFix [48] | 38.2 | 63.4 | 63.2 |
| w/ BPR† | 40.0 | 63.4 | 67.0 |

Table 9: **Transfer to Other Models**: BPR† was trained on the results of Mask R-CNN. It can be successfully transferred to refine the results of PointRend and SegFix.

| w/ BPR | AP | AP* | $AP^*_S$ | $AP^*_M$ | $AP^*_L$ | AF |
|---|---|---|---|---|---|---|
| | 38.4 | 40.4 | 24.5 | 48.3 | 57.2 | 54.5 |
| ✔ | **39.2** | **42.1** | **24.8** | **50.3** | **60.4** | **58.4** |

Table 10: **Results on COCO**. AP* is measured on the higher-quality LVIS [12] annotations. We improved based on the results of Mask R-CNN ResNeXt-FPN-101 baseline.

results to refine the predictions of PointRend [17] and SegFix [48]. Note that these two methods are also designed to improve boundary quality in segmentation. As shown in Table 9, the transferred model still improved the results of PointRend and SegFix by a large margin, suggesting that our method is compatible with them.

### 4.4. Overall Results

**Comparison with State-of-the-art Methods**. We integrated the optimal design choices and hyperparameters found in above ablation experiments into a stronger BPR model. Specifically, we adopted the HRNetV2-W48 as our refinement network, with 256×256 input patches resized from 64×64, and a NMS threshold of 0.55 during inference. We evaluated the framework on Cityscapes `val` and `test` sets and compared the performance against some state-of-the-art methods in Table 8. (1) Compared with the Mask R-CNN baseline, we achieved a significant improvement (+**4.3**% AP in both `val` and `test`). We outperformed SegFix [48] by a large margin, which is also a boundary refinement module applied to the same baseline with ours. Furthermore, by applying our BPR model to the results already refined by SegFix, we can still improved a lot (slightly lower than applying BPR only). (2) We transferred the above BPR model to refine the results of the stronger PolyTransform [21] baseline ($1^{st}$ place at CVPR 2020). Our "PolyTransform + BPR" consistently improved 2.3% AP on Cityscapes `test` set and also outperformed "PolyTransform + SegFix" ($2^{nd}$ place at ECCV 2020) by a large margin (+1.2%). By applying BPR to "PolyTransform + SegFix", we established a new state-of-the-art on Cityscapes `test` with AP of **42.7%**, reaching **1st** place on the Cityscapes leaderboard by the CVPR 2021 submission deadline.

**Qualitative Results**. We show some qualitative results on Cityscapes `val` in Figure 4. Compared with the coarse predictions of Mask R-CNN, our BPR framework generated substantially better instance segmentation results with precise and distinct boundaries. It largely alleviated the over-smoothing issues [17] in previous methods caused by the low resolution feature maps. More results are included in *Supplementary Materials*. In addition, we also provided a detailed limitation analysis in *Supplementary Materials*.

**Speed**. Only the speed of refinement network was considered in Table 5 and 6, excluding the patch extraction

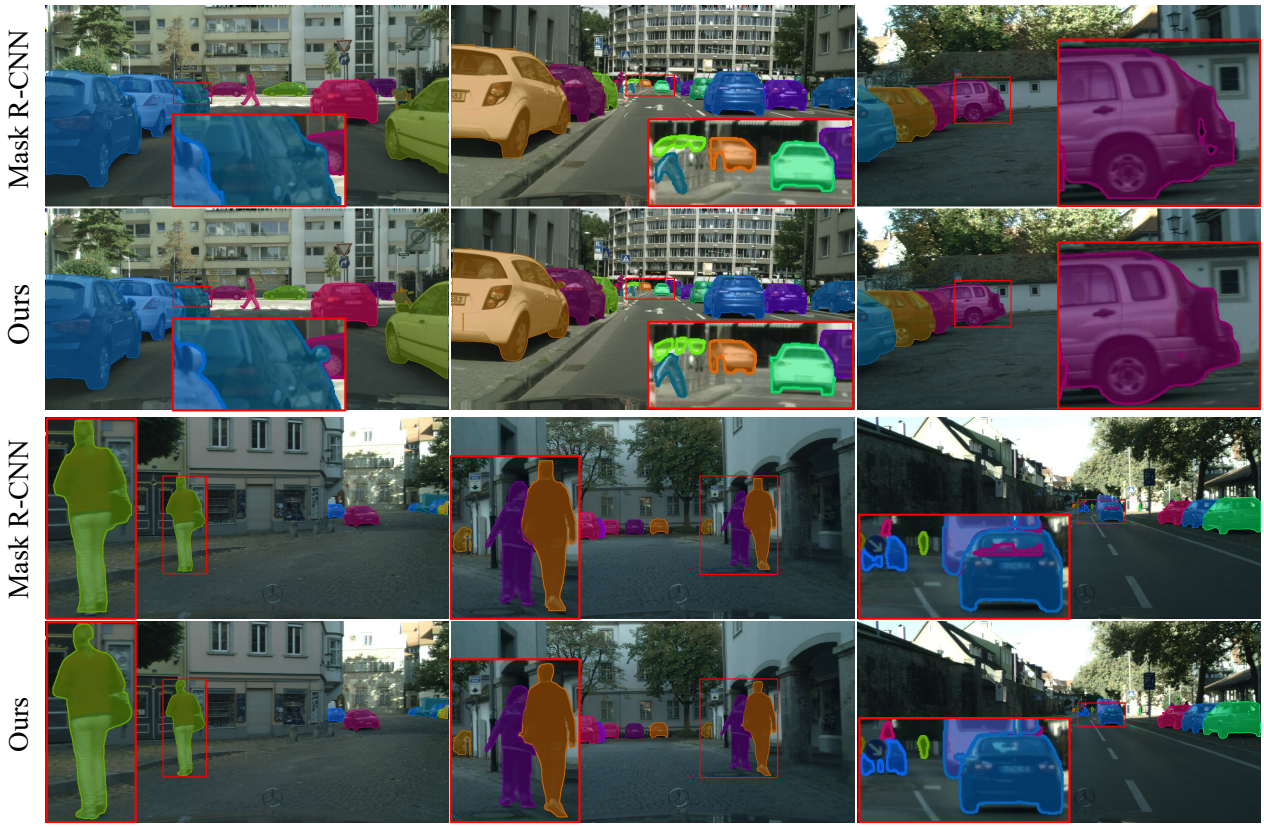


Figure 4: Qualitative results on Cityscapes `val`. The proposed framework (2nd and 4th rows) produces substantially better masks with more precise boundaries than Mask R-CNN (1st and 3rd rows). Best viewed digitally and in colour.

and reassembling time. As a whole pipeline, it takes about 211ms to process a single Cityscapes image (1024×2048) on a single RTX 2080Ti GPU under the default setting of ablation experiments, which is still much faster than PolyTransform [21]. The detailed speed calculation and more speed analysis are included in *Supplementary Materials*.

**Results on COCO Dataset**. To demonstrate the generality of our framework, we also report the results on the more challenging COCO dataset [23], which contains 80 categories and more images (118k/5k for train/val). It is important to note that the coarse annotations in COCO may not fully reflect the improvements in mask quality [12]. Following PointRend [17], we further report the AP$^\star$ measured using the higher quality LVIS [12] annotations. We randomly sampled about 8% of instances for fast training. As shown in Table 10, we improved the powerful Mask R-CNN ResNeXt-FPN-101 baseline by 0.8% AP and 1.7% AP$^\star$ on `val2017`. The coarse annotations on COCO `train2017` may provide ambiguous optimization objectives, especially for our local boundary patches. It may mislead the learning of our BPR model, leading to suboptimal results. This issue was also observed in some contour-based instance segmentation methods [31, 42, 45]. We believe that training with more instances on higher quality annotations (*e.g.* LVIS) can further improve the results. More analysis on COCO dataset is included in *Supplementary Materials*.

## 5. Conclusion

In this paper, we propose a conceptually simple yet effective boundary refinement framework to improve the boundary quality for any instance segmentation model. Starting from a coarse instance mask, we extract and refine a series of boundary patches along the predicted instance boundaries through an effective refinement network. The proposed framework achieved consistent and impressive improvements based on different baselines. Qualitative results show that our approach produced high-quality masks with precise and distinct boundaries.

**Acknowledgements** This work was supported by the National Key Research and Development Program of China (No. 2017YFA0700904), the National Science and Technology Major Project (No. 2018ZX01028-102), the National Natural Science Foundation of China (Nos. 61836014, U19B2034, 62061136001 and 61620106010) and THU-Bosch JCML center.

## References


[1] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5221–5229, 2017. 3

[2] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Int. Conf. Comput. Vis.*, pages 9157–9166, 2019. 1, 2, 3

[3] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8573–8581, 2020. 1, 2, 3

[4] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 3, 6

[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Eur. Conf. Comput. Vis.*, pages 801–818, 2018. 3, 6

[6] Xinlei Chen, Ross Girshick, Kaiming He, and Piotr Dollár. Tensormask: A foundation for dense object segmentation. In *Int. Conf. Comput. Vis.*, pages 2061–2069, 2019. 1, 3

[7] Tianheng Cheng, Xinggang Wang, Lichao Huang, and Wenyu Liu. Boundary-preserving mask r-cnn. In *Eur. Conf. Comput. Vis.*, 2020. 2, 3, 7

[8] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3213–3223, 2016. 4, 5

[9] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. *arXiv preprint arXiv:1708.02551*, 2017. 3

[10] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3146–3154, 2019. 3

[11] Naiyu Gao, Yanhu Shan, Yupei Wang, Xin Zhao, Yinan Yu, Ming Yang, and Kaiqi Huang. Ssap: Single-shot instance segmentation with affinity pyramid. In *Int. Conf. Comput. Vis.*, pages 642–651, 2019. 3, 7

[12] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5356–5364, 2019. 7, 8

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Int. Conf. Comput. Vis.*, pages 2961–2969, 2017. 1, 2, 7

[14] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6409–6418, 2019. 1, 2

[15] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6399–6408, 2019. 3

[16] Alexander Kirillov, Evgeny Levinkov, Bjoern Andres, Bogdan Savchynskyy, and Carsten Rother. Instancecut: from edges to instances with multicut. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5008–5017, 2017. 2

[17] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9799–9808, 2020. 2, 3, 7, 8

[18] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Adv. Neural Inform. Process. Syst.*, pages 109–117, 2011. 3

[19] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13906–13915, 2020. 1, 2

[20] Ke Li, Bharath Hariharan, and Jitendra Malik. Iterative instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3659–3667, 2016. 3

[21] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9131–9140, 2020. 2, 3, 5, 6, 7, 8

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Int. Conf. Comput. Vis.*, pages 2980–2988, 2017. 1, 2

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Eur. Conf. Comput. Vis.*, pages 740–755, 2014. 5, 8

[24] Shu Liu, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. Sgn: Sequential grouping networks for instance segmentation. In *Int. Conf. Comput. Vis.*, pages 3496–3504, 2017. 7

[25] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8759–8768, 2018. 1, 2, 7

[26] Yu Liu, Guanglu Song, Yuhang Zang, Yan Gao, Enze Xie, Junjie Yan, Chen Change Loy, and Xiaogang Wang. 1st place solutions for openimage2019–object detection and instance segmentation. *arXiv preprint arXiv:2003.07557*, 2020. 6

[27] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3431–3440, 2015. 3

[28] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571, 2016. 3

[29] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *arXiv preprint arXiv:2001.05566*, 2020. 3

[30] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8837–8845, 2019. 3

[31] Sida Peng, Wen Jiang, Huaijin Pi, Xiuli Li, Hujun Bao, and Xiaowei Zhou. Deep snake for real-time instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8533–8542, 2020. 3, 8

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Adv. Neural Inform. Process. Syst.*, pages 91–99, 2015. 1, 2

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241, 2015. 3

[34] Konstantin Sofiiuk, Olga Barinova, and Anton Konushin. Adaptis: Adaptive instance selection network. In *Int. Conf. Comput. Vis.*, pages 7355–7363, 2019. 7

[35] Towaki Takikawa, David Acuna, Varun Jampani, and Sanja Fidler. Gated-scnn: Gated shape cnns for semantic segmentation. In *Int. Conf. Comput. Vis.*, pages 5229–5238, 2019. 2, 3, 5

[36] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *Eur. Conf. Comput. Vis.*, 2020. 1, 2, 3

[37] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Int. Conf. Comput. Vis.*, pages 9627–9636, 2019. 1, 2

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Adv. Neural Inform. Process. Syst.*, pages 5998–6008, 2017. 3

[39] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 3, 4, 6

[40] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. Solo: Segmenting objects by locations. In *Eur. Conf. Comput. Vis.*, 2020. 1, 3

[41] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. In *Adv. Neural Inform. Process. Syst.*, 2020. 1, 2, 3

[42] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12193–12202, 2020. 1, 3, 8

[43] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1492–1500, 2017. 6

[44] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. Upsnet: A unified panoptic segmentation network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8818–8826, 2019. 7

[45] Wenqiang Xu, Haiyang Wang, Fubo Qi, and Cewu Lu. Explicit shape encoding for real-time instance segmentation. In *Int. Conf. Comput. Vis.*, pages 5168–5177, 2019. 3, 8

[46] Hui Ying, Zhaojin Huang, Shu Liu, Tianjia Shao, and Kun Zhou. Embedmask: Embedding coupling for one-stage instance segmentation. *arXiv preprint arXiv:1912.01954*, 2019. 1, 2

[47] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. *arXiv preprint arXiv:1909.11065*, 2019. 3

[48] Yuhui Yuan, Jingyi Xie, Xilin Chen, and Jingdong Wang. Segfix: Model-agnostic boundary refinement for segmentation. In *Eur. Conf. Comput. Vis.*, pages 489–506, 2020. 2, 3, 5, 6, 7

[49] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi Zhang, Haibin Lin, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020. 6

[50] Hang Zhang, Han Zhang, Chenguang Wang, and Junyuan Xie. Co-occurrent features in semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 548–557, 2019. 3

[51] Rufeng Zhang, Zhi Tian, Chunhua Shen, Mingyu You, and Youliang Yan. Mask encoding for single shot instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 10226–10235, 2020. 1, 2

[52] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2881–2890, 2017. 3

[53] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 1, 2