Joint COCO and Mapillary Workshop at ICCV 2019: COCO Instance Segmentation Challenge Track

Technical Report: Cascade RPN and Feature+Task Cascade for Instance Segmentation

Thang Vu[1]* Keundong Lee[2]* Hyunjun Jang[1] Junyeong Kim[1] Trung X. Pham[1]
Jong Gook Ko[2] Chang D. Yoo[1]
[1]Korea Advanced Institute of Science and Technology (KAIST), Korea
[2]Electronics and Telecommunications Research Institute (ETRI), Korea

Abstract

This report presents a region proposal network referred to as Cascade RPN and an instance segmentation network referred to as Feature+Task Cascade for improving instance segmentation performance. Cascade RPN improves region proposal performance by (1) systematically ensuring alignment through RPN stages and (2) adaptively selecting the metric to define positive samples. A simple implementation of two-stage Cascade RPN achieves 13.4 points AR higher than conventional RPN. Meanwhile, Feature+Task Cascade improve segmentation performance by (1) performing feature cascade that explicitly constructs information flow between the box and mask branches, (2) performing task cascade that eliminates the mask predictions on intermediate noisy boxes and the requirements of multi-stage mask predictions, and (3) relying on a global context branch to provide individual objects with context prior for final predictions. Experiments on COCO dataset show that while running 38% faster, the FTC improves the AP of the box and mask predictions by respectively 1.3 and 2.3 points compared to the strong Cascade Mask R-CNN baseline. The overall system achieves 49.6 mask AP without using large mini-batch size and synchronized batch normalization. Codes for Cascade RPN and FTC are available respectively at https://github.com/thangvubk/Cascade-RPN *and* https://github.com/thangvubk/FTC.

1. Method Details

The proposed system consists of a new region proposal network Cascade RPN and a new instance segmentation network FTC, which are in turn presented in this section.

*Equal contribution

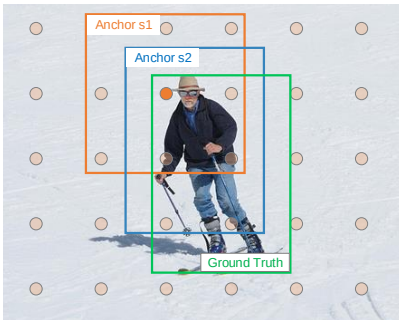


Figure 1: The misalignment problem in iterative RPN. At stage 2, the anchor is refined to be closer to the ground truth; however, it does not align with the image feature (represented by orange dot).

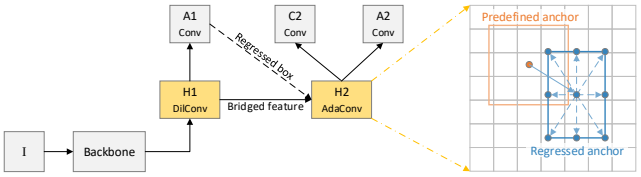
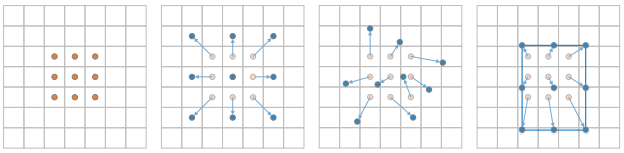


Figure 2: Cascade RPN relies on the proposed Adaptive Convolution (AdaConv) to address the misalignment problem in Iterative RPN.

1.1. Cascade RPN

Region proposals have become the *de-facto* paradigm for high-quality object detectors. Region proposals serve as the attention mechanism that enables the detector to produce accurate bounding boxes while maintaining computation tractability. Recently, Region Proposal Network (RPN) [5] has been proposed that shares full-image convolutional features with the detection network, enabling nearly cost-free region proposals. There have been a number of studies attempting to improve the performance of RPN by iterative



(a) (b) (c) (d)

Figure 3: Illustrations of the sampling locations in different convolutional layers with 3×3 kernel of (a) convolution, (b) dilated convolution, (c) deformable convolution, and (d) the proposed adaptive convolution.

refinement. However, this approach ignores the problem that the regressed boxes are misaligned to the image features, breaking the alignment rule required for object detection, as shown in Figure 1.

Cascade RPN [6] is proposed to systematically address the problem arising from heuristically defining the anchors and aligning the features to the anchors. First, instead of using multiple anchors with different scales and aspect ratios, Cascade RPN relies on a single anchor and incorporates both anchor-based and anchor-free criteria in defining positive boxes to achieve high performance. Second, to benefit from multi-stage refinement while maintaining the alignment between anchor boxes and features, Cascade RPN relies on the proposed adaptive convolution that adapts to the refined anchors after each stage.

Adaptive Convolution. Given a feature map $\boldsymbol{x}$, in the standard 2D convolution, the feature map is first sampled using a regular grid $\mathbb{R} = \{(r_x, r_y)\}$, and the samples are summed up with the weight $\boldsymbol{w}$. Here, the grid $\mathbb{R}$ is defined by the kernel size and dilation. For example, $\mathbb{R} = \{(-1,-1),(-1,0),\ldots,(0,1),(1,1)\}$ corresponds to kernel size 3×3 and dilation 1. For each location $\boldsymbol{p}$ on the output feature $\boldsymbol{y}$, we have:

$$\boldsymbol{y}[\boldsymbol{p}] = \sum_{\boldsymbol{r} \in \mathbb{R}} \boldsymbol{w}[\boldsymbol{p}] \cdot \boldsymbol{x}[\boldsymbol{p} + \boldsymbol{r}]. \tag{1}$$

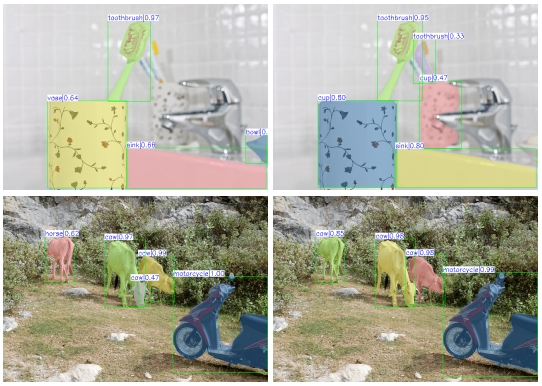
In adaptive convolution, the regular grid $\mathbb{R}$ is replaced by the offset field $\mathbb{O}$ that is directly inferred from the input anchor.

$$\boldsymbol{y}[\boldsymbol{p}] = \sum_{\boldsymbol{o} \in \mathbb{O}} \boldsymbol{w}[\boldsymbol{p}] \cdot \boldsymbol{x}[\boldsymbol{p} + \boldsymbol{o}]. \tag{2}$$

Let $\bar{\boldsymbol{a}}$ denote the projection of anchor $\boldsymbol{a}$ onto the feature map. The offset $\boldsymbol{o}$ can be decoupled into center offset and shape offset (shown in Figure 2):

$$\boldsymbol{o} = \boldsymbol{o}_{\text{ctr}} + \boldsymbol{o}_{\text{shp}}, \tag{3}$$

where $\boldsymbol{o}_{\text{ctr}} = (\bar{a}_x - p_x, \bar{a}_y - p_y)$ and $\boldsymbol{o}_{\text{shp}}$ is defined by the anchor shape and kernel size. For example, if kernel size is 3×3, then $\boldsymbol{o}_{\text{shp}} \in \{(-\frac{\bar{a}_w}{2}, -\frac{\bar{a}_h}{2}), (-\frac{\bar{a}_w}{2}, 0), \ldots, (0, \frac{\bar{a}_h}{2}), (\frac{\bar{a}_w}{2}, \frac{\bar{a}_h}{2})\}$. Illustrations of sampling locations in adaptive convolution and common convolutions are shown in Figure 3.



(a) FTC w/o ctx. (b) FTC w/ ctx.

Figure 4: Examples from COCO dataset show that global context information provides helpful clues for inferring individual objects. Here, "w/ ctx." and "w/o ctx." denote the proposed FTC with and without global context information, respectively. In the first row, without global context information, a cup is confused as a vase. With global context information (right figures), the detector produces more reliable predictions based on context prior.

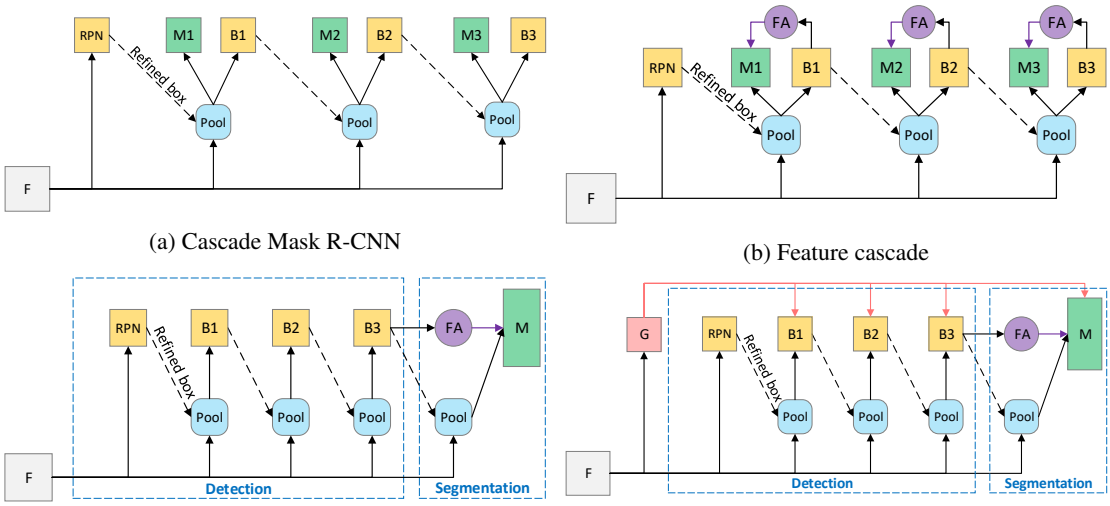
1.2. Feature and Task Cascade

The proposed FTC introduces three enhancements: *feature cascade*, *task cascade*, and *global context*.

Feature Cascade. Motivated by the observation that "implicit" mutual information between the box and mask branches improves the overall performance, the *feature cascade* "explicitly" constructs the information flow between two branches at the feature space as presented in Figure 5b.

Task Cascade. To address the limitations of previous cascade models that perform mask predictions on intermediate noisy boxes and require multiple upsamplers and predictors, the proposed FTC introduces *task cascade* that performs segmentation after the accurate boxes are obtained at the final stage of the detector, referred to Figure 5c. Task cascade not only improves the accuracy of instance masks but also is computationally friendly since most of the mask computation is performed in a low-resolution space before being upsampled (commonly by deconvolution) only once at the end of the mask branch.

Global Context. The *global context* branch provides objects with context prior for final predictions, explained in Figure 4. The global context branch takes as input the backbone features and outputs global context features (shown in Figure 5d), which is guided by multi-label classification.



(a) Cascade Mask R-CNN

(b) Feature cascade

(c) Task cascade

(d) The proposed FTC

Figure 5: Architecture of cascade approaches: (a) Cascade Mask R-CNN. (b-d) Evolution of the proposed FTC. Here, "F", "RPN", "Pool", "B", "M", "FA" and "G" denote image features, Region Proposal Network, region-wise pooling, box branch, mask branch, feature adaptation, and global context branch, respectively.

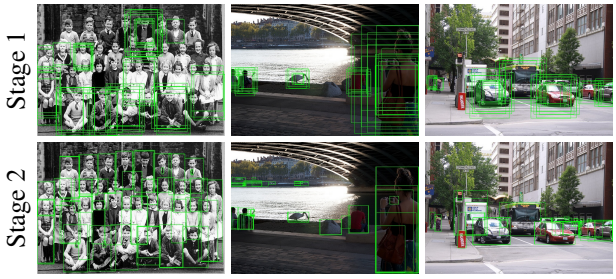


Figure 6: Examples of region proposal results at stage 1 (first row) and stage 2 (second row) of Cascade RPN.

Table 1: Region proposal results on COCO `val`.

Method	AR_{100}	AR_{300}	AR_{1000}	Time (s)
RPN	44.6	52.9	58.3	**0.04**
Iterative RPN	48.5	55.4	58.8	0.05
GA-RPN	59.1	65.1	68.5	0.06
Cascade RPN	**61.1**	**67.6**	**71.7**	0.06

2. Experiment Results

The experiments are performed on COCO 2017 segmentation dataset [4]. Additionally, the coco stuff annotations [1] are used for semantic branch analogous to [3].

2.1. Cascade RPN

Region Proposal Performance Cascade RPN is compared to recent region proposal methods including RPN [5], Iterative RPN, GA-RPN [7]. Table 1 shows that Cascade RPN outperforms other region proposal methods by significant margins.

Table 2: Detection results on COCO `test-dev`

Method	Proposal method	AP	AP_{50}	AP_{75}
Fast R-CNN	RPN	37.0	**59.5**	39.9
	Cascade RPN	**40.1**	59.4	**43.8**
Faster R-CNN	RPN	37.1	**59.3**	40.1
	Cascade RPN	**40.6**	58.9	**44.5**

Detection Performance. Cascade RPN and the baselines are integrated into common two-stage object detectors, including Fast R-CNN and Faster R-CNN. Table 2 shows that the detectors using Cascade RPN achieve better detection performance.

Qualitative Evaluation. The examples of region proposal results at the first and second stages are illustrated in the first and second row of Figure 6, respectively. The results show that the output proposals at the second stage are more accurate and cover a larger number of objects.

2.2. FTC

Quantitative Comparison The proposed FTC is compared to other cascade approach for instance segmentation including Cascade Mask R-CNN [2] and HTC [3]. Table 3 shows that FTC outperforms the others in multiple evaluation metrics, including bbox AP, Mask AP and speed.

Qualitative Comparison. Figure 7 shows the visual com-

Figure 7: Qualitative comparison on COCO `val` between the proposed FTC and other methods.

Table 3: Benchmarking results between the proposed FTC and other state-of-the-art methods on COCO `test-dev`.

Method	Backbone	AP^{box}	AP	AP_{50}	AP_{75}	Fps
CM R-CNN	R-50	43.7	37.9	59.8	40.8	4.5
HTC		43.6	38.5	60.1	41.7	4.5
FTC (ours)		**45.0**	**40.2**	**62.3**	**43.4**	**6.2**
CM R-CNN	X-101	47.3	40.9	63.7	44.2	3.7
HTC		47.2	41.3	63.9	44.8	3.7
FTC (ours)		**48.3**	**42.7**	**65.7**	**46.4**	**4.6**

Table 4: Results of our final model on COCO `test-dev`

Settings	AP^{box}	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
2018 winner	-	49.0	**73.0**	53.9	**33.9**	**52.3**	61.2
Ours	55.3	**49.6**	72.7	**54.7**	29.5	51.9	**66.5**
FTC baseline	45.0	40.2	62.3	43.4	22.4	42.8	53.4
+ ms train	46.0	42.4	63.3	45.9	21.7	44.2	59.9
+ DCN	47.6	43.2	64.7	46.9	21.9	45.3	60.8
+ GN head	48.4	43.8	64.8	47.9	22.5	45.8	61.7
+ X-101	51.0	46.1	68.4	50.4	24.3	48.7	64.6
+ CRPN	51.9	46.3	68.0	50.7	24.3	48.6	64.8
+ ms test	53.7	48.0	70.6	52.9	28.0	50.1	64.7
+ ensemble	**55.3**	**49.6**	**72.7**	**54.7**	**29.5**	**51.9**	**66.5**

parison of the proposed FTC with other cascade models. It is clear that FTC achieves more accurate detected bounding boxes and instance masks.

2.3. COCO 2018 Instance Segmentation Challenge

Using Cascade RPN and FTC as the foundation, we achieve 49.6 mask AP on COCO `test-dev`, which is 0.6 point improvement compared to the winning entry of COCO 2018. It is noted that we do not adopt Synchronized Batch Normalization due to resource limitations. Instead, the normalization layers at the backbone are frozen and Group Normalization layers are used at the head. We expect better results when all the normalization layers are updated during training. Here, we report the detailed component-wise analysis of the overall system in Table 4.

FTC baseline. The FTC on ResNet-50 is the baseline.

Multi-scale Training. During training, the scale of the short edge is randomly sampled in the range [400, 1400] and the scale of the long edge is fixed to 1600.

DCN. The deformable convolution is applied in 3 last stages (C3-C5) of the backbone.

GN head. The Batch Normalization layers in the backbone are frozen and Group Normalization are used in FPN and network heads.

X-101. The backbones ResNet-50 is replaced by ResNext-101 64x4d.

CRPN. The trained detector is finetuned with the precomputed proposals of Cascade RPN.

Multi-scale testing. We use test time augmentation.

Ensemble. The ensemble uses 3 types of backbones: ResNeXt-101 64x4, ResNeXt-101 32x8, and SeNet-154. In our experiments, SENet-154 achieves comparable AP compared to that of ResNeXt-101. We hypothesize that it is because the backbone normalization layers are frozen.

3. Conclusion

We have presented Cascade RPN and FTC for instance segmentation. Cascade RPN improve the region proposal quality though multi-stage refinement while ensuring alignment between anchor and feature. FTC improves instance segmentation performance though task cascade, feature cascade and global context information. The final system achieves 49.6 mask AP on the COCO `test-dev` dataset.

References

[1] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. COCO-Stuff: Thing and stuff classes in context. In *CVPR*, 2018. 3

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation, 2019. 3

[3] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *CVPR*, 2019. 3

[4] 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 *ECCV*, 2014. 3

[5] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 3

[6] Thang Vu, Hyunjun Jang, Trung X Pham, and Chang D Yoo. Cascade rpn: Delving into high-quality region proposal network with adaptive convolution. In *NeurIPS*, 2019. 2

[7] Jiaqi Wang, Kai Chen, Shuo Yang, Chen Change Loy, and Dahua Lin. Region proposal by guided anchoring. In *CVPR*, 2019. 3