PolarMask: Single Shot Instance Segmentation with Polar Representation

Enze Xie[1,2]*, Peize Sun[3]*, Xiaoge Song[4]*, Wenhai Wang[4],
Xuebo Liu[2], Ding Liang[2], Chunhua Shen[5], Ping Luo[1]

[1]The University of Hong Kong [2]Sensetime Group Ltd
[3]Xi'an Jiaotong University [4]Nanjing University [5]The University of Adelaide

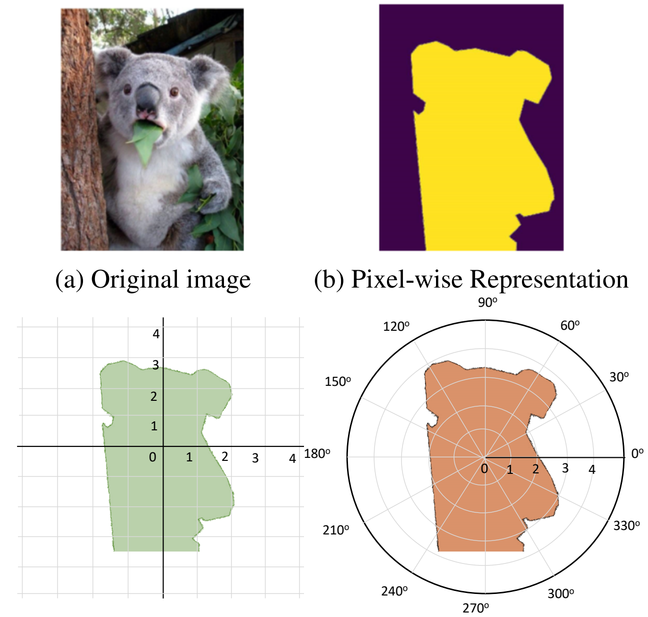
E-mail: xieenze@hku.hk

Abstract

In this paper, we introduce an anchor-box free and single shot instance segmentation method, which is conceptually simple, fully convolutional and can be used as a mask prediction module for instance segmentation, by easily embedding it into most off-the-shelf detection methods. Our method, termed PolarMask, formulates the instance segmentation problem as instance center classification and dense distance regression in a polar coordinate. Moreover, we propose two effective approaches to deal with sampling high-quality center examples and optimization for dense distance regression, respectively, which can significantly improve the performance and simplify the training process. Without any bells and whistles, PolarMask achieves 32.9% in mask mAP with single-model and single-scale training/testing on challenging COCO dataset. For the first time, we demonstrate a much simpler and flexible instance segmentation framework achieving competitive accuracy. We hope that the proposed PolarMask framework can serve as a fundamental and strong baseline for single shot instance segmentation tasks. Code is available at: github.com/xieenze/PolarMask.

1. Introduction

Instance segmentation is one of the fundamental tasks in computer vision, which enables numerous downstream vision applications. It is challenging as it requires to predict both the location and the semantic mask of each instance in an image. Therefore intuitively instance segmentation can be solved by bounding box detection then semantic segmentation within each box, adopted by two-stage methods, such as Mask R-CNN [12]. Recent trends in the vision community have spent more effort in designing simpler pipelines of bounding box detectors [14, 18, 25, 26, 28] and subsequent instance-wise recognition tasks including instance segmentation [2, 4, 29], which is also the main focus of our work here. *Thus, our aim is to design a conceptually simple mask prediction module that can be easily plugged into many off-the-shelf detectors, enabling instance segmentation.*



(a) Original image (b) Pixel-wise Representation

(c) Cartesian Representation (d) Polar Representation

Figure 1 – Instance segmentation of different mask representation. (a) is the original image. (b) is the pixel-wise mask representation. (c) and (d) represent a mask by its contour, in the Cartesian and Polar coordinates, respectively.

Instance segmentation is usually solved by binary classification in a spatial layout surrounded by bounding boxes, shown in Figure 1(b). Such pixel-to-pixel correspondence prediction is luxurious, especially in the single-shot methods. Instead, we point out that masks can be recovered successfully and effectively if the contour is obtained. An intuitive method to locate contours is shown in Figure 1(c), which predicts the Cartesian coordinates of the point com-

*indicates equal contribution.

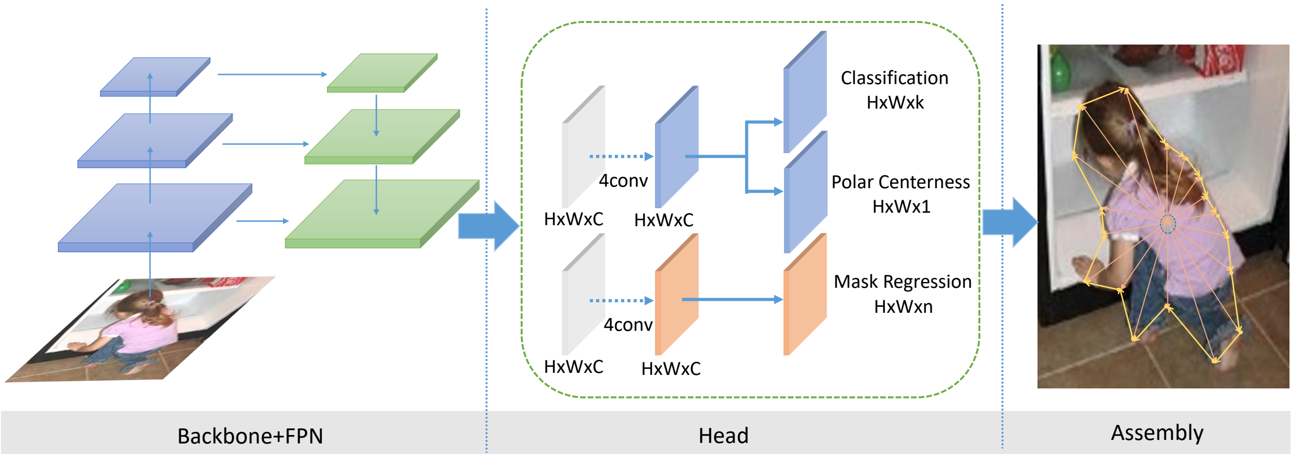


Figure 2 – The overall pipeline of PolarMask. The left part contains the backbone and feature pyramid to extract features of different levels. The middle part is the two heads for classification and polar mask regression. H, W, C are the height, width, channels of feature maps, respectively, and k is the number of categories (e.g., $k = 80$ on the COCO dataset), n is the number of rays (e.g., $n = 36$)

.

posing the contour. Here we term it as Cartesian Representation. The second approach is Polar Representation, which applies the angle and the distance as the coordinate to locate points, shown in Figure 1(d).

In this work, we design instance segmentation methods based on the Polar Representation since its inherent advantages are as follows: (1) The origin point of the polar coordinate can be seen as the center of object. (2) Starting from the origin point, the point in contour is determined by the distance and angle. (3) The angle is naturally directional and makes it very convenient to connect the points into a whole contour. We claim that Cartesian Representation may exhibit first two properties similarly. However, it lacks the advantage of the third property.

We instantiate such an instance segmentation method by using the recent object detector FCOS [25], mainly for its simplicity. Note that, it is possible to use other detectors such as RetinaNet [18], YOLO [23] with minimal modification to our framework. Specifically, we propose PolarMask, formulating instance segmentation as instance center classification and dense distance regression in a polar coordinate, shown in Figure 2. The model takes an input image and predicts the distance from a sampled positive location (candidates of the instance center) to the instance contour at each angle, and after assembling, outputs the final mask. The overall pipeline of PolarMask is almost as simple and clean as FCOS. It introduces *negligible* computation overhead. Simplicity and efficiency are the two key factors to single shot instance segmentation, and PolarMask achieves them successfully.

Furthermore, PolarMask can be viewed as a generalization of FCOS, or FCOS is a special case of PolarMask since bounding boxes can be viewed as the simplest mask with only 4 direction. Thus, one is suggested to use PolarMask over FCOS for instance recognition wherever mask annotation is available [5, 19].

In order to maximize the advantages of Polar Representation, we propose Polar Centerness and Polar IoU Loss to deal with sampling high-quality center examples and optimization for dense distance regression, respectively. They improve mask accuracy by about 15% relatively, showing considerable gains under stricter localization metrics. Without bells and whistles, PolarMask achieves 32.9% in mask mAP with single-model and single-scale training/testing on the challenging COCO dataset [19].

The main contributions of this work are three-fold:

- We introduce a new method for instance segmentation, termed PolarMask, to model instance masks in the polar coordinate, which converts instance segmentation to two parallel tasks: instance center classification and dense distance regression. The main desirable characteristics of PolarMask is being simple and effective.
- We propose the Polar IoU Loss and Polar Centerness, tailored for our framework. We show that the proposed Polar IoU loss can largely ease the optimization and considerably improve the accuracy, compared with standard loss such as the smooth-l_1 loss. In parallel, Polar Centerness improves the original idea of 'centreness' in FCOS, leading to further performance boost.
- For the first time, we demonstrate a much simpler and flexible instance segmentation framework achieving competitive performance compared with more complex one-stage methods, which typically involve multi-scale train and longer training time. We hope that PolarMask can serve as a fundamental and strong baseline for single shot instance segmentation.

2. Related Work

Two-Stage Instance Segmentation. Two-stage instance segmentation often formulates this task as the paradigm of 'Detect then Segment' [16, 12, 20, 15]. They often detect bounding boxes then perform segmentation in the area of each bounding box. The main idea of FCIS [16] is to predict a set of position-sensitive output channels fully convolutionally. These channels simultaneously address object classes, boxes, and masks, making the system fast. Mask R-CNN [12], built upon Faster R-CNN, simply adds an additional mask branch add use RoI-Align to replace RoI-Pooling [9] for improved accuracy. Following Mask R-CNN, PANet [20] introduces bottom-up path augmentation, adaptive feature pooling and fully-connected fusion to boost up the performance of instance segmentation. Mask Scoring R-CNN [15] re-scores the confidence of mask from classification score by adding a mask-IoU branch, which makes the network to predict the IoU of mask and ground-truth.

In summary, the above methods typically consist of two steps, first detecting bounding box and then segmenting in each bounding box. They can achieve state-of-the-art performance but are often slow.

One Stage Instance Segmentation. Deep Watershed Transform [1] uses fully convolutional networks to predict the energy map of the whole image and use the watershed algorithm to yield connected components corresponding to object instances. InstanceFCN [6] uses instance-sensitive score maps for generating proposals. It first produces a set of instance-sensitive score maps, then an assembling module is used to generate object instances in a sliding window. The recent YOLACT [2] first generates a set of prototype masks, the linear combination coefficients for each instance, and bounding boxes, then linearly combines the prototypes using the corresponding predicted coefficients and then crops with a predicted bounding box. TensorMask [4] investigates the paradigm of dense sliding-window instance segmentation, using structured 4D tensors to represent masks over a spatial domain. ExtremeNet [29] uses keypoint detection to predict 8 extreme points of one instance and generates an octagon mask, achieving relatively reasonable object mask prediction. The backbone of ExtremeNet is HourGlass [21], which is very heavy and often needs longer training time. It also needs several steps for post-processing including grouping. In contrast, our method is simpler than ExtremeNet while much achieving better results than ExtremeNet.

Note that these methods do not model instances directly and they can sometime be hard to optimize (e.g., longer training time, more data augmentation and extra labels). Our PolarMask directly models instance segmentation with a much simpler and flexible way of two paralleled branches: classifying each pixel of mass-center of instance and regressing the dense distance of rays between mass-center and contours. The most significant advantage of PolarMask is being simple and efficient compared with all above methods. In the experiments, we have not adopted many training tricks, such as data augmentation and longer training time, since our goal is to design a conceptually simple and flexible mask prediction module.

3. Our Method

In this section, we first briefly introduce the overallarchitecture of the proposed PolarMask. Then, we reformulate instance segmentation with the proposed Polar Representation. Next, we introduce a novel concept of Polar Centerness to ease the procedure of choosing high-quality center samples. Finally, we introduce a new Polar IoU Loss to optimize dense regression problem.

3.1. Architecture

PolarMask is a simple, unified network composed of a backbone network [13], a feature pyramid network [17], and two or three task-specific heads, depending on whether predicting bounding boxes.[1] The settings of the backbone and feature pyramid network are same as FCOS [25]. While there exist many stronger candidates for those components, we align these setting with FCOS to show the simplicity and effectiveness of our instance modeling method.

3.2. Polar Mask Segmentation

In this section, we will describe how to model instances in the polar coordinate in detail.

Polar Representation Given an instance mask, we firstly sample a candidate center (x_c, y_c) of the instance and the point located on the contour (x_i, y_i), $i = 1, 2, ..., N$. Then, starting from the center, n rays are emitted uniformly with the same angle interval $\Delta\theta$ (e.g., $n = 36$, $\Delta\theta = 10°$), whose length is determined from the center to the contour.

Thus we model the instance mask in the polar coordinate as center and n rays. Since the angle interval is pre-defined, only the length of the ray needs to be predicted. In this way, we formulate the instance segmentation as instance center classification and dense distance regression in a polar coordinate.

Mass Center There are many choices for the center of the instance, such as box center or mass-center. How to choose a better center depends on its effect on the mask prediction performance. Here we verify the upper bound of box center and mass-center, and conclude that mass-center is more advantageous. Details are in Figure 7. We explain that the mass-center has a greater probability of falling inside the instance, compared with the box center. Although for some extreme cases, such as a donut, neither mass-center nor box

[1] It is optional to have the box prediction branch or not. As we empirically show, the box prediction branch has little impact on mask prediction.

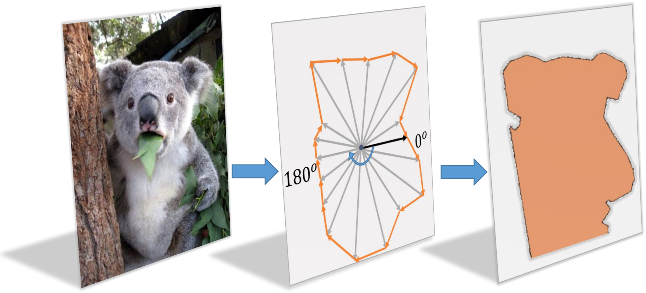


Figure 3 – **Mask Assembling**. Polar Representation provides a directional angle. The contour points are connected one by one start from 0° (bold line) and assembles the whole contour and mask.

center lies inside the instance. We leave it for further research.

Center Samples Location (x, y) is considered as a center sample if it falls into areas around the mass-center of any instance. Otherwise it is a negative sample. We define the region for sampling positive pixels to be 1.5× strides [25] of the feature map from the mass-center to left, top, right and bottom. Thus each instance has about 9∼16 pixels near the mass-center as center examples. It has two advantages: (1) Increasing the number of positive samples from 1 to 9∼16 can largely avoid imbalance of positive and negative samples. Nevertheless, focal loss [18] is still needed when training the classification branch. (2) Mass-center may not be the best center sample of an instance. More candidate points make it possible to automatically find the best center of one instance. We will discuss it in details in Section 3.3.

Distance Regression Given a center point (x_c, y_c) and the intersection points located on the contour (x_i, y_i), $i = 1$, 2, ..., N, the angle θ_i and the distance d_i between the center point and each contour point can be computed easily, from which the required n rays can be picked up in most cases. However, there are some corner cases:

- If one ray has multiple intersection points with the contour of instance, we directly choose the one with the maximum length.
- If one ray, which starts from the center outside of the mask, does not have intersection points with the contour of an instance at some certain angles, we set its regression target as the minimum value ϵ (e.g., $\epsilon = 10^{-6}$).
- If the intersection point between a ray and the contour happens to be a sub-pixel (i.e., pixel coordinates are not integers), we can always use an interpolation method, such as linear interpolation, to estimate its regression target.

We argue that these corner cases are the main obstacles of restricting the upper bound of Polar Representation from reaching 100% AP. However, it is not supposed to be seen as Polar Representation being inferior to the *non-parametric* Pixel-wise Representation. The evidence is two-folds. First, even the Pixel-wise Representation is far away from the upper bound of 100% AP in practice, since some operation, such as down-sampling, is indispensable. Second, current performance is far away from the upper bound regardless of the Pixel-wise Representation or Polar Representation. Therefore, the research effort is suggested to better spend on improving the practical performance of models, rather than the theoretical upper bound.

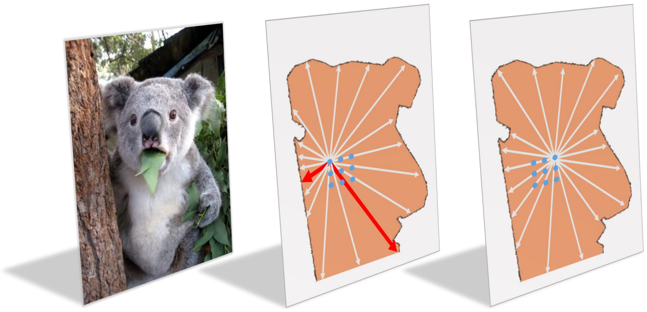


Figure 4 – **Polar Centerness**. Polar centerness is used to down-weight such regression task as the high diversity of rays' lengths as shown in red lines in the middle plot. These example are always hard to optimize and produce low-quality masks. During inference, the polar centerness predicted by the network is multiplied to the classification score, thus can down-weight the low-quality masks.

The training of regression branch is non-trivial. First, the mask branch in PolarMask is actually a dense distance regression task since every training example has n rays (e.g., $n = 36$). It may cause the imbalance between the regression loss and classification loss. Second, for one instance, its n rays are relevant and should be trained as a whole, rather than being seen as a set of independent regression examples. Therefore, we put forward the Polar IoU Loss, discussed in details in Section 3.4.

Mask Assembling During inference, the network outputs the classification and centerness, we multiply centerness with classification and obtain final confidence scores. We only assemble masks from at most 1k top-scoring predictions per FPN level, after thresholding the confidence scores at 0.05. The top predictions from all levels are merged and non-maximum suppression (NMS) with a threshold of 0.5 is applied to yield the final results. Here we introduce the mask assembling process and a fast NMS process.

Given a center sample (x_c, y_c) and the ray's length d_i, $i = 1$, 2, ..., n, we can calculate the position of each corresponding contour point with the following formula:

$$x_i = \cos\theta_i \times d_i + x_c \tag{1}$$

$$y_i = \sin\theta_i \times d_i + y_c. \tag{2}$$

Starting from 0°, the contour points are connected one by one, shown in Figure 3 and finally assembles a whole contour as well as the mask.

We apply NMS to remove redundant masks. To fasten the process, We calculate the smallest bounding boxes of masks and then apply NMS based on the IoU of boxes. We verify that such a simplified post-processing do not negatively effect the final mask performance.

3.3. Polar Centerness

Centerness [25] is introduced to suppress these low-quality detected objects without introducing any hyper-parameters and it is proven to be effective in object bounding box detection. However, directly transferring it to our system can be sub-optimal since its centerness is designed for bounding boxes and we care about mask prediction.

Given a set $\{d_1, d_2, \ldots, d_n\}$ for the length of n rays of one instance, where $d_{\max}$ and $d_{\min}$ are the maximum and minimum of the set. We propose Polar Centerness:

$$\text{Polar Centerness} = \sqrt{\frac{\min(\{d_1, d_2, \ldots, d_n\})}{\max(\{d_1, d_2, \ldots, d_n\})}} \tag{3}$$

Specifically, we add a single layer branch, in parallel with the classification branch to predict Polar Centerness of a location, as shown in Figure 2. It is a simple yet effective strategy to re-weight the points so that the closer d_{min} and d_{max} are, higher weight the point is assigned. Experiments show that Polar Centerness improves the accuracy especially under stricter localization metrics, such as AP$_{75}$.

3.4. Polar IoU Loss

As discussed above, the method of polar segmentation converts the task of instance segmentation into a set of regression problems. In most cases in the field of object detection and segmentation, smooth-l_1 loss [10] and IoU loss [27] are the two effective ways to supervise the regression problems. However, smooth-l_1 loss overlooks the correlation between samples of the same objects, thus, resulting in less accurate localization. IoU loss, however, the training procedure considers the optimization as a whole, and directly optimizes the metric of interest. Nevertheless, computing the IoU of the predicted mask and its ground-truth is tricky and very difficult to implement parallel computations. In this work, we derive an easy and effective algorithm to compute mask IoU based on the polar vector representation and achieve competitive performance, as shown in Figure 5.

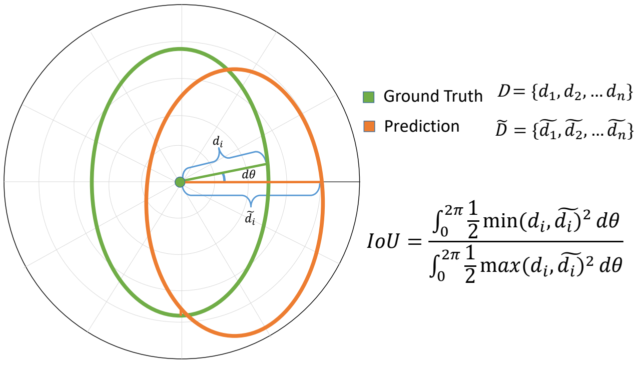

🟩 Ground Truth
$D = \{d_1, d_2, \ldots d_n\}$
🟧 Prediction
$\widetilde{D} = \{\widetilde{d_1}, \widetilde{d_2}, \ldots \widetilde{d_n}\}$
d_i
$d\theta$
$\widetilde{d_i}$
$IoU = \frac{\int_0^{2\pi} \frac{1}{2} \min(d_i, \widetilde{d_i})^2 \, d\theta}{\int_0^{2\pi} \frac{1}{2} max(d_i, \widetilde{d_i})^2 \, d\theta}$

Figure 5 – Mask IoU in Polar Representation. Mask IoU (interaction area over union area) in the polar coordinate can be calculated by integrating the differential IoU area in terms of differential angles.

We introduce Polar IoU Loss starting from the definition of IoU, which is the ratio of interaction area over union area between the predicted mask and ground-truth. In the polar coordinate system, for an instance, mask IoU is calculated as follows:

$$\text{IoU} = \frac{\int_0^{2\pi} \frac{1}{2} \min(d, d^*)^2 d\theta}{\int_0^{2\pi} \frac{1}{2} \max(d, d^*)^2 d\theta} \tag{4}$$

where regression target d and predicted d^* are length of the ray, angle is θ. Then we transform it to the discrete form[2]

$$\text{IoU} = \lim_{N \to \infty} \frac{\sum_{i=1}^{N} \frac{1}{2} d_{\min}^2 \Delta\theta_i}{\sum_{i=1}^{N} \frac{1}{2} d_{\max}^2 \Delta\theta_i} \tag{6}$$

When N approaches infinity, the discrete form is equal to continuous form. We assume that the rays are uniformly emitted, so $\Delta\theta = \frac{2\pi}{N}$, which further simplifies the expression. We empirically observe that the power form has little impact on the performance if it is discarded and simplified into the following form:

$$\text{Polar IoU} = \frac{\sum_{i=1}^{n} d_{\min}}{\sum_{i=1}^{n} d_{\max}} \tag{7}$$

Polar IoU Loss is the binary cross entropy (BCE) loss of Polar IoU. Since the optimal IoU is always 1, the loss is actually is negative logarithm of Polar IoU:

$$\text{Polar IoU Loss} = \log \frac{\sum_{i=1}^{n} d_{\max}}{\sum_{i=1}^{n} d_{\min}} \tag{8}$$

Our proposed Polar IoU Loss exhibits two advantageous properties: (1) It is differentiable, enabling back propagation; and is very easy to implement parallel computations, thus facilitating a fast training process. (2) It predicts the regression targets as a whole. It improves the overall performance by a large margin compared with smooth-l_1 loss, shown in our experiments. (3) As a bonus, Polar IoU Loss is able to automatically keep the balance between classification loss and regression loss of dense distance prediction. We will discuss it in detail in our experiments.

[2] For notation convenience, we define:

$$d_{\min} = \min(d, d^*), d_{\max} = \max(d, d^*). \tag{5}$$

rays	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
18	26.2	48.7	25.4	11.8	28.2	38.0
24	27.3	49.5	26.9	12.4	29.5	40.1
36	**27.7**	49.6	27.4	12.6	30.2	39.7
72	27.6	49.7	27.2	12.9	30.0	39.7

(a) Number of Rays: More rays bring a large gain, while too many rays saturate since it already depicts the mask ground-truth well.

loss	α	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
Smooth-l_1	0.05	24.7	47.1	23.7	11.3	26.7	36.8
	0.30	25.1	46.4	24.5	10.6	27.3	37.3
	1.00	20.2	37.9	19.6	8.6	20.6	31.1
Polar IoU	1.00	**27.7**	49.6	27.4	12.6	30.2	39.7

(b) Polar IoU Loss *vs.* Smooth-L1 Loss: Polar IoU Loss outperforms Smooth-l_1 loss, even the best variants of balancing regression loss and classification loss.

centerness	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
Cartesian	27.7	49.6	27.4	12.6	30.2	39.7
Polar	**29.1**	49.5	29.7	12.6	31.8	42.3

(c) Polar Centerness *vs.* Cartesian Centerness: Polar Centerness bring a large gain, especially high IoU AP$_{75}$ and large instance AP$_L$.

box branch	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
w	27.7	49.6	27.4	12.6	30.2	39.7
w/o	27.5	49.8	27.0	13.0	30.0	40.0

(d) Box Branch: Box branch makes no difference to performance of mask prediction.

backbone	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
ResNet-50	29.1	49.5	29.7	12.6	31.8	42.3
ResNet-101	30.4	51.1	31.2	13.5	33.5	43.9
ResNeXt-101	32.6	54.4	33.7	15.0	36.0	47.1

(e) Backbone Architecture: All models are based on FPN. Better backbones bring expected gains: deeper networks do better, and ResNeXt improves on ResNet.

scale	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$	FPS
400	22.9	39.8	23.2	4.5	24.4	41.7	19.03
600	27.6	47.5	28.3	9.8	30.1	43.1	12.86
800	29.1	49.5	29.7	12.6	31.8	42.3	8.98

(f) Accuracy/speed trade-off on ResNet-50: PolarMask performance with different image scales. The FPS is reported on TitanX GPUs, for which the speed is about 50% slower than 1080Ti.

Table 1 – Ablation experiments for PolarMask. All models are trained on `trainval35k` and tested on `minival`, using ResNet50-FPN backbone unless otherwise noted.

4. Experiments

We present results of instance segmentation on the challenging COCO benchmark [19]. Following common practice [12, 4], we train using the union of 80K train images and a 35K subset of val images (`trainval35k`), and report ablations on the remaining 5K val. images (`minival`). We also compare results on `test-dev`. We adopt the 1× training strategy [11, 3], single scale training and testing of image short-edge as 800 unless otherwise noted.

Training Details In ablation study, ResNet-50-FPN [13, 17] is used as our backbone networks and the same hyper-parameters with FCOS [25] are used. Specifically, our network is trained with stochastic gradient descent (SGD) for 90K iterations with the initial learning rate being 0.01 and a mini-batch of 16 images. The learning rate is reduced by a factor of 10 at iteration 60K and 80K, respectively. Weight decay and momentum are set as 0.0001 and 0.9, respectively. We initialize our backbone networks with the weights pre-trained on ImageNet [8]. The input images are resized to have their shorter side being 800 and their longer side less or equal to 1333.

4.1. Ablation Study

Verification of Upper Bound The first concern about PolarMask is that it might not depict the mask precisely. In this section we prove that this concern may not be necessary. Here we verify the upper bound of PolarMask as the IoU of predicted mask and ground-truth when all of the rays regress to the distance equal to ground-truth. The verification results on different numbers of rays are shown in Figure 7. It can be seen that IoU is approaching to nearly perfect (above 90%) when the number of rays increases, which shows that Polar Segmentation is able to model the mask very well. Therefore, the concern about the upper bound of PolarMask is not necessary. Also, it is more reasonable to use mass-center than bounding box-center as the center of an instance because the bounding box center is more likely to fall out of the instance.

Number of Rays It plays the fundamental role in the whole system of PolarMask. From Table 1a and Figure 7, more rays show higher upper bound and better AP. For example, 36 rays improve by 1.5% AP compared to 18 rays. Also, too many rays, 72 rays, saturate the performance since it already depicts the mask contours well and the number of rays is no longer the main factor constraining the performance.

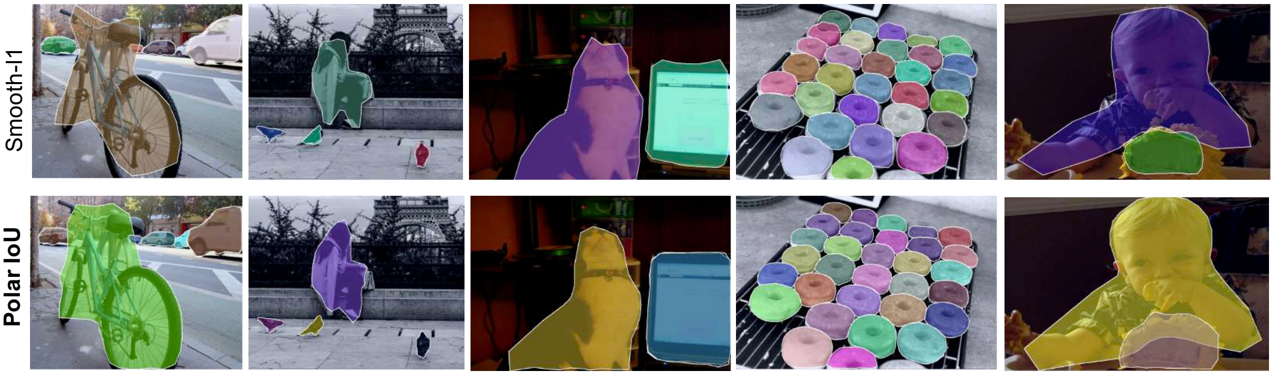


Figure 6 – Visualization of PolarMask with Smooth-l_1 loss and Polar IoU loss. Polar IoU Loss achieves to regress more accurate contour of instance while Smooth-l_1 Loss exhibits systematic artifacts.

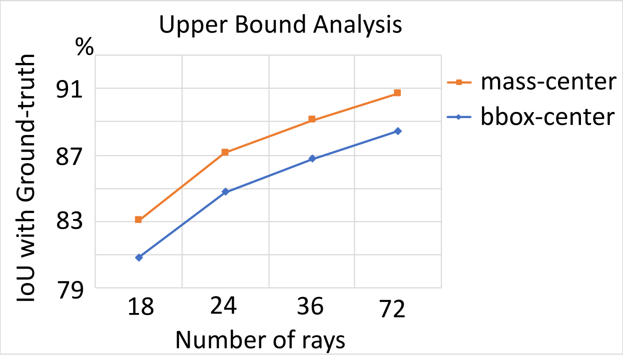


Figure 7 – Upper Bound Analysis. More rays can model instance mask with higher IoU with Ground Truth, and mass-center is more friendly to represent an instance than box-center.

Polar IoU Loss *vs.* Smooth-l_1 Loss We test both Polar IoU Loss and Smooth-l_1 Loss in our architecture. We note that the regression loss of Smooth-l_1 Loss is *significantly* larger than the classification loss, since our architecture is a task of dense distance prediction. To cope with the imbalance, we select different factor α to regression loss in Smooth-l_1 Loss. Experiments results are shown in Table 1b. Our Polar IoU Loss achieves 27.7% AP without balancing regression loss and classification loss. In contrast, the best setting for Smooth-l_1 Loss achieves 25.1% AP, a gap of 2.6% AP, showing that Polar IoU Loss is more effective than Smooth-l_1 loss for training the regression task of distance between mass-center and contours.

We hypothesize that the gap may come from two folds. First, the Smooth-l_1 Loss may need more hyper-parameter search to achieve better performance, which can be time-consuming compared to the Polar IoU Loss. Second, Polar IoU Loss predicts all rays of one instance as a whole, which is superior to Smooth-l_1 Loss.

In Figure 6 we compare some results using the Smooth-l_1 Loss and Polar IoU Loss respectively. Smooth-l_1 Loss exhibits systematic artifacts, suggesting that it lacks supervision of the level of the whole object. PolarMask shows more smooth and precise contours.

Polar Centerness *vs.* Cartesian Centerness The comparison experiments are shown in Table 1c. Polar Centerness improves by 1.4% AP overall. Particularly, AP$_{75}$ and AP$_L$ are raised considerably, 2.3% AP and 2.6% AP, respectively.

We explain as follows. On the one hand, low-quality masks make more negative effect on high-IoU. On the other hand, large instances have more possibility of large difference between maximum and minimum lengths of rays, which is exactly the problem which Polar Centerness is committed to solve.

Box Branch Most of previous methods of instance segmentation require the bounding box to locate area of object and then segment the pixels inside the object. In contrast, *PolarMask is capable to directly output the mask without bounding box.*

In this section, we test whether the additional bounding box can help improve the mask AP as follows. If the ray reaches outside of the bounding box, the ray is cut off at the boundary. From Table 1d, we can see that bounding box makes little difference to performance of mask prediction. Thus, we do not have the bounding box prediction head in PolarMask for simplicity and faster speed.

Backbone Architecture Table 1e shows the results of PolarMask on different backbones. It can be seen that better feature extracted by deeper and advanced design networks improve the performance as expected.

Speed *vs.* Accuracy Larger image sizes yield higher accuracy, in slower inference speeds. Table 1f shows the speed/accuracy trade-off for different input image scales, defined by the shorter image side. The FPS is reported on an outdated Titan-X GPU.

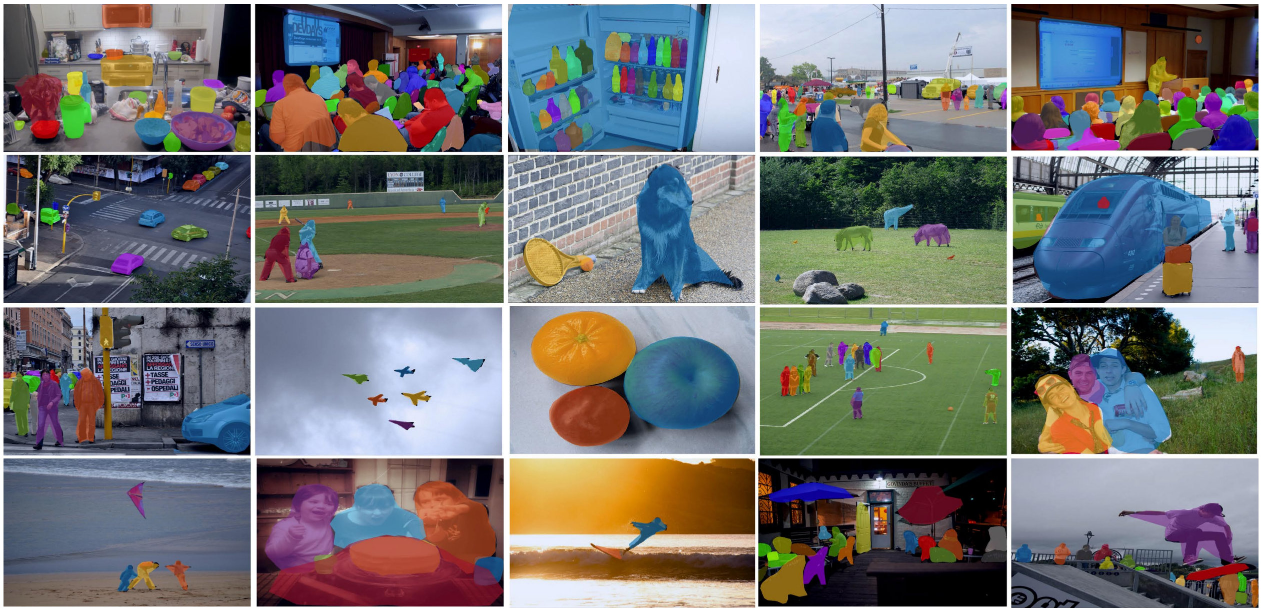


Figure 8 – Results of PolarMask on COCO `test-dev` images with ResNet-101-FPN, achieving 30.4% mask AP (Table 2).

method	backbone	epochs	aug	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
two-stage									
MNC [7]	ResNet-101-C4	12	○	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [16]	ResNet-101-C5-dilated	12	○	29.2	49.5	-	7.1	31.3	50.0
Mask R-CNN [12]	ResNeXt-101-FPN	12	○	37.1	60.0	39.4	16.9	39.9	53.5
one-stage									
ExtremeNet [29]	Hourglass-104	100	✓	18.9	44.5	13.7	10.4	20.4	28.3
TensorMask [4]	ResNet-101-FPN	72	✓	37.1	59.3	39.4	17.1	39.1	51.6
YOLACT [2]	ResNet-101-FPN	48	✓	31.2	50.6	32.8	12.1	33.3	47.1
PolarMask	ResNet-101-FPN	12	○	30.4	51.9	31.0	13.4	32.4	42.8
PolarMask	ResNeXt-101-FPN	12	○	32.9	55.4	33.8	15.5	35.1	46.3

Table 2 – **Instance segmentation** mask AP on the COCO `test-dev`. The standard training strategy [11] is training by 12 epochs; and 'aug' means data augmentation, including multi-scale and random crop. ✓ is training with 'aug', ○ is without 'aug'.

4.2. Comparison to state-of-the-art

We evaluate PolarMask on the COCO dataset and compare `test-dev` results to state-of-the-art methods including both one-stage and two-stage models, shown in Table 2. PolarMask outputs are visualized in Figure 8.

Without any bells and whistles, PolarMask is able to achieve competitive performance with more complex one-stage methods. Since our aim is to design a conceptually simple and flexible mask prediction module, many improvements methods [24, 22], such as multi-scale training and longer training time is beyond the scope of this work. We argue that the gap of YOLACT [2] and PolarMask comes from more training epochs and data augmentation. If these methods are applied to PolarMask, the performance can be readily improved. Besides, the gap of TensorMask [4] and PolarMask arises from tensor bipyramid and aligned representation. Considering these methods are time-costing and memory-costing, we do not plug them to PolarMask.

5. Conclusion

PolarMask is a single shot anchor-free instance segmentation method with two paralleled branches: classifying mass-center of instances and regressing the dense lengths of rays between sampled locations around the mass-center and contours. Different from previous works that typically solve mask prediction as binary classification in a spatial layout, PolarMask puts forward polar representation and transforms mask prediction to dense distance regression. PolarMask is designed almost as simple and clean as single shot object detectors, introducing negligible computing overhead. We hope that the proposed PolarMask framework can serve as a fundamental and strong baseline for single shot instance segmentation task.

References

[1] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5221–5229, 2017.

[2] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. *arXiv*, 2019.

[3] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Mmdetection: Open mmlab detection toolbox and benchmark, 2019.

[4] Xinlei Chen, Ross Girshick, Kaiming He, and Piotr Dollár. Tensormask: A foundation for dense object segmentation. *arXiv preprint arXiv:1903.12174*, 2019.

[5] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[6] Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *European Conference on Computer Vision*, pages 534–549. Springer, 2016.

[7] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158, 2016.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009.

[9] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[11] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron. `https://github.com/facebookresearch/detectron`, 2018.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[14] Lichao Huang, Yi Yang, Yafeng Deng, and Yinan Yu. Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874*, 2015.

[15] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6409–6418, 2019.

[16] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2359–2367, 2017.

[17] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[19] 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 *European conference on computer vision*, pages 740–755. Springer, 2014.

[20] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018.

[21] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

[22] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. Megdet: A large mini-batch object detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6181–6189, 2018.

[23] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[24] Bharat Singh and Larry S Davis. An analysis of scale invariance in object detection snip. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3578–3587, 2018.

[25] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. *Proc. International Conf. Computer Vision (ICCV)*, 2019.

[26] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. *arXiv preprint arXiv:1904.11490*, 2019.

[27] Jiahui Yu, Yuning Jiang, Zhangyang Wang, Zhimin Cao, and Thomas Huang. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM International Conference on Multimedia*, pages 516–520. ACM, 2016.

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

[29] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 850–859, 2019.