CVPR2020 Paper Sharing

Shen Gao

Apr. 17, 2020

Contents

FocalMix: Semi-Supervised Learning for 3D Medical Image Detection

Dong Wang[1]* Yuan Zhang[2]* Kexin Zhang[2,3]† Liwei Wang[1,2]
[1]Center for Data Science, Peking University
[2]Key Laboratory of Machine Perception, MOE, School of EECS, Peking University
[3]Yizhun Medical AI Co., Ltd
{wangdongcis,yuan.z,zhangkexin,wanglw}@pku.edu.cn

Contribution

- Proposed FocalMix, a novel semi-supervised learning framework for 3D medical image detection
- First to investigate semi-supervised learning in medical image detection.
- Proposed approach can also improve fully-supervised approaches.

MixMatch

- Proposed in *MixMatch: A Holistic Approach to Semi-Supervised Learning*.
- https://arxiv.org/abs/1905.02249

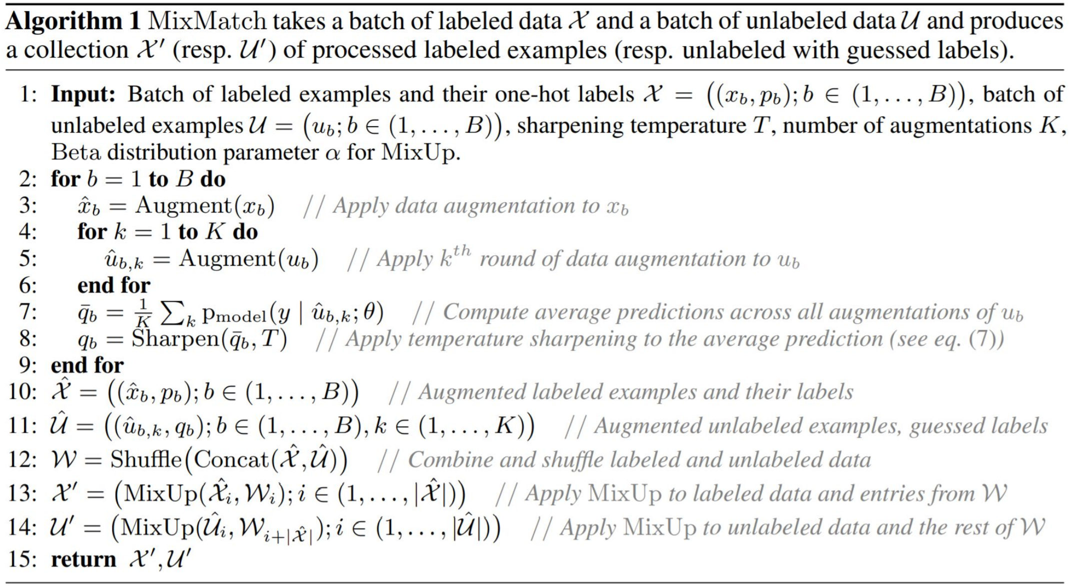
$$Sharpen(p,T)_i := p_i^{\frac{1}{T}} \Big/ \sum_{j=1}^{L} p_j^{\frac{1}{T}}$$

$$\lambda \sim Beta(\eta,\eta)$$

$$\tilde{\lambda} = max(\lambda, 1-\lambda)$$

$$\hat{x} = \tilde{\lambda}x + (1-\tilde{\lambda})x'$$

$$\hat{y} = \tilde{\lambda}y + (1-\tilde{\lambda})y'$$



Algorithm 1 MixMatch takes a batch of labeled data $\mathcal{X}$ and a batch of unlabeled data $\mathcal{U}$ and produces a collection $\mathcal{X}'$ (resp. $\mathcal{U}'$) of processed labeled examples (resp. unlabeled with guessed labels).

```
 1: Input: Batch of labeled examples and their one-hot labels $\mathcal{X} = \big((x_b, p_b); b \in (1,\ldots,B)\big)$, batch of
    unlabeled examples $\mathcal{U} = \big(u_b; b \in (1,\ldots,B)\big)$, sharpening temperature $T$, number of augmentations $K$,
    Beta distribution parameter $\alpha$ for MixUp.
 2: for $b = 1$ to $B$ do
 3:     $\hat{x}_b = \text{Augment}(x_b)$   // Apply data augmentation to $x_b$
 4:     for $k = 1$ to $K$ do
 5:         $\hat{u}_{b,k} = \text{Augment}(u_b)$   // Apply $k^{th}$ round of data augmentation to $u_b$
 6:     end for
 7:     $\bar{q}_b = \frac{1}{K}\sum_k \text{p}_{\text{model}}(y \mid \hat{u}_{b,k}; \theta)$   // Compute average predictions across all augmentations of $u_b$
 8:     $q_b = \text{Sharpen}(\bar{q}_b, T)$   // Apply temperature sharpening to the average prediction (see eq. (7))
 9: end for
10: $\hat{\mathcal{X}} = \big((\hat{x}_b, p_b); b \in (1,\ldots,B)\big)$   // Augmented labeled examples and their labels
11: $\hat{\mathcal{U}} = \big((\hat{u}_{b,k}, q_b); b \in (1,\ldots,B), k \in (1,\ldots,K)\big)$   // Augmented unlabeled examples, guessed labels
12: $\mathcal{W} = \text{Shuffle}\big(\text{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}})\big)$   // Combine and shuffle labeled and unlabeled data
13: $\mathcal{X}' = \big(\text{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1,\ldots,|\hat{\mathcal{X}}|)\big)$   // Apply MixUp to labeled data and entries from $\mathcal{W}$
14: $\mathcal{U}' = \big(\text{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1,\ldots,|\hat{\mathcal{U}}|)\big)$   // Apply MixUp to unlabeled data and the rest of $\mathcal{W}$
15: return $\mathcal{X}', \mathcal{U}'$
```

Focal Loss

- Proposed in *Focal Loss for Dense Object Detection*.
- https://arxiv.org/abs/1708.02002
- Can deal with imbalanced classes.
- α_t: weighting factor to balance different classes.
- γ: focusing parameter.

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t)$$

$$p_t = \begin{cases} p & \text{if y} = 1 \\ 1-p & \text{otherwise.} \end{cases}$$

FocalMix

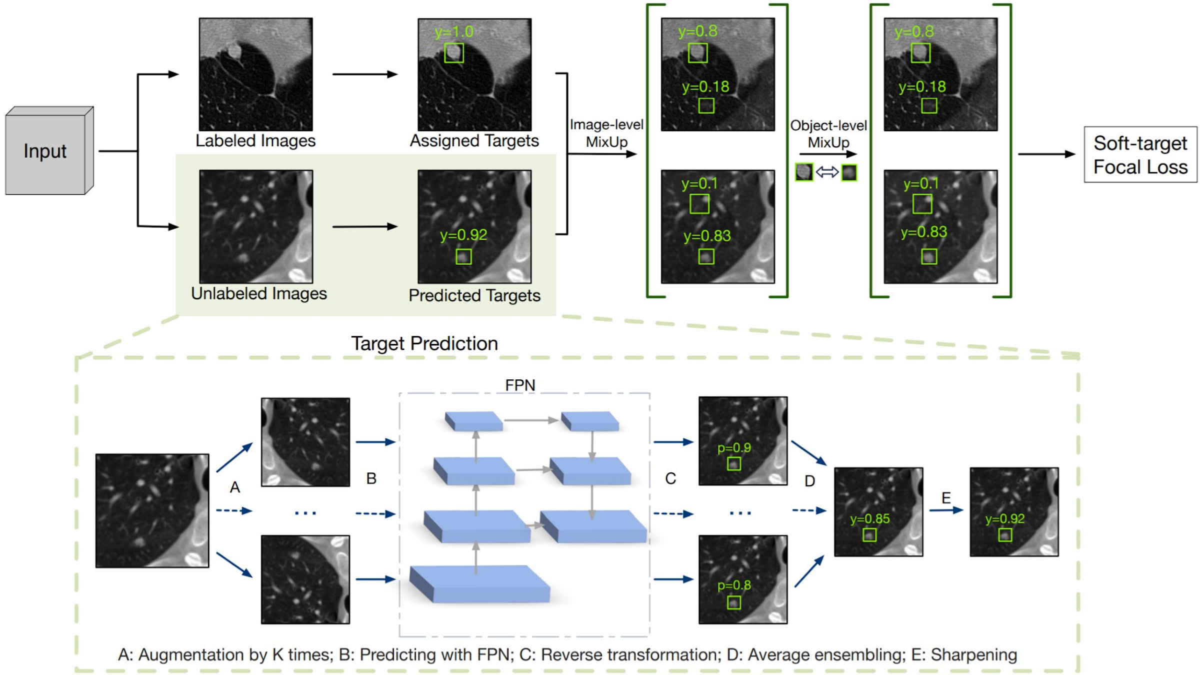


Figure 2: **Overview of our proposed method FocalMix.** For an input batch, the training targets of anchors in labeled images are assigned according to annotated boxes, while the unlabeled are predicted with the current model as shown in the lower part of the figure. After applying two levels of MixUp to the entire batch, we use the proposed soft-target focal loss to train the model. Throughout this paper, we only show a slice of each 3D CT scan with 3D anchors on it for ease of presentation.

FocalMix

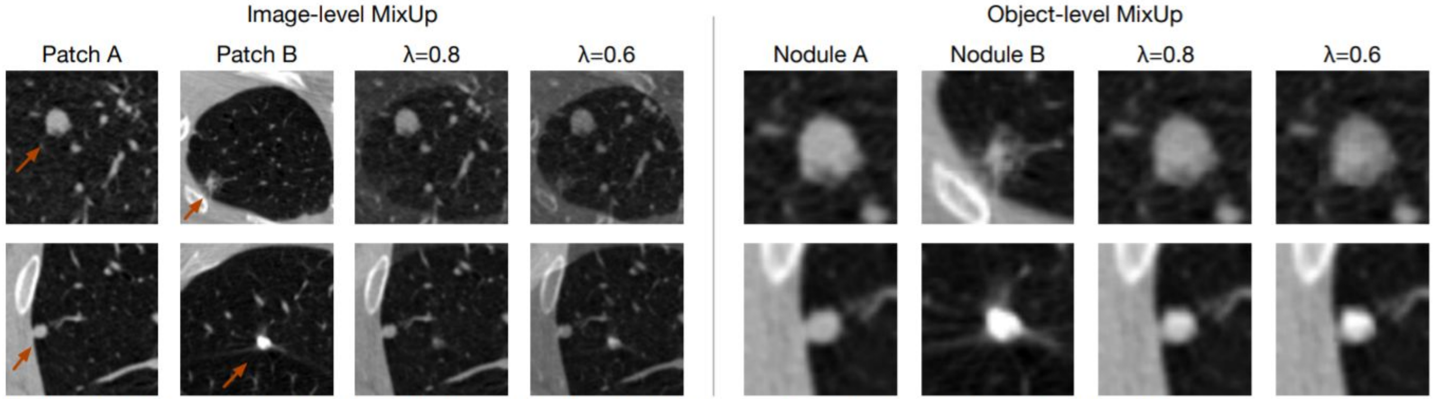


Figure 4: **Illustrative examples for two MixUp methods.** The left figure shows the image-level MixUp, where red arrows point to nodules in the original image. The right figure demonstrates the object-level MixUp, where we zoom in on the nodules and locate them in the center of each image patch for better visualization.

Soft-target Focal Loss

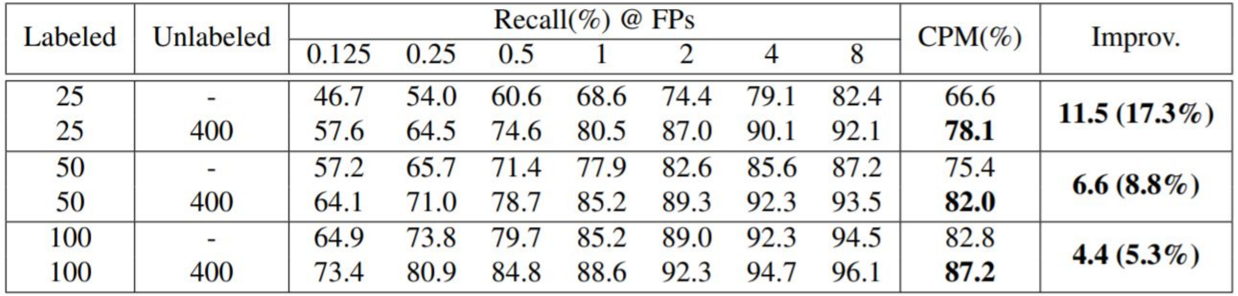
- Improved Focal loss.
- Can handle continuous class label y.
- Focal loss is a special case of SFL.

$$SFL(p) = [\alpha_0 + y(\alpha_1 - \alpha_0)] \, |y - p|^{\gamma} CE(y, p)$$

Experiment

- Datasets:
    - Luna16
    - NLST
- Metrics:
    - FROC (?)
    - CPM

Experiment



Labeled	Unlabeled	Recall(%) @ FPs							CPM(%)	Improv.
		0.125	0.25	0.5	1	2	4	8		
25	-	46.7	54.0	60.6	68.6	74.4	79.1	82.4	66.6	**11.5 (17.3%)**
25	400	57.6	64.5	74.6	80.5	87.0	90.1	92.1	**78.1**	
50	-	57.2	65.7	71.4	77.9	82.6	85.6	87.2	75.4	**6.6 (8.8%)**
50	400	64.1	71.0	78.7	85.2	89.3	92.3	93.5	**82.0**	
100	-	64.9	73.8	79.7	85.2	89.0	92.3	94.5	82.8	**4.4 (5.3%)**
100	400	73.4	80.9	84.8	88.6	92.3	94.7	96.1	**87.2**	

Table 1: **Main results on the LUNA16 dataset.** We evaluate FocalMix with {25, 50, 100} labeled CT scans, respectively. *Improv.* denotes the improvements in CPM over the fully-supervised baseline (relative improvements shown in parentheses).

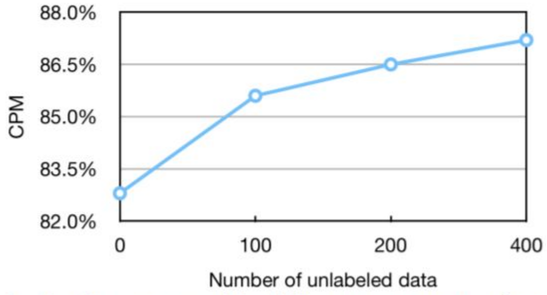
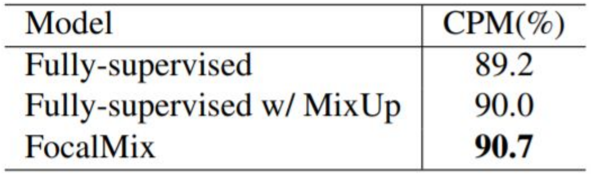


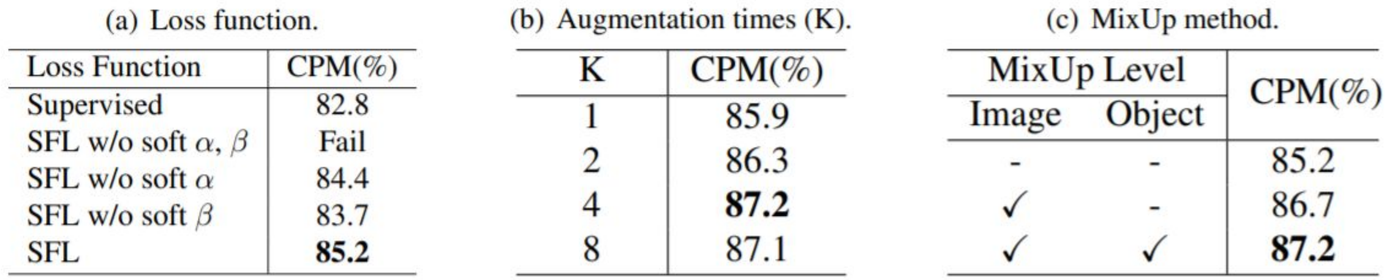
Figure 3: **Performance with different amounts of unlabeled data on LUNA16.** We use 100 labeled images.



Model	CPM(%)
Fully-supervised	89.2
Fully-supervised w/ MixUp	90.0
FocalMix	**90.7**

Table 4: **FocalMix with larger scale labeled and unlabeled data.** We use all the labeled data in LUNA16 and unlabeled data selected from NLST.

Experiment



(a) Loss function.

Loss Function	CPM(%)
Supervised	82.8
SFL w/o soft α, β	Fail
SFL w/o soft α	84.4
SFL w/o soft β	83.7
SFL	**85.2**

(b) Augmentation times (K).

K	CPM(%)
1	85.9
2	86.3
4	**87.2**
8	87.1

(c) MixUp method.

MixUp Level		CPM(%)
Image	Object	
-	-	85.2
✓	-	86.7
✓	✓	**87.2**

Table 3: **Ablation study.** Models are trained with 100 labeled scans and 400 unlabeled ones. *Fail* denotes a divergent result.

Cars Can't Fly up in the Sky: Improving Urban-Scene Segmentation via Height-driven Attention Networks

Sungha Choi[1,3] **Joanne T. Kim**[2,3] **Jaegul Choo**[4]

[1]A&B Center, LG Electronics, Seoul, South Korea
[2]Lawrence Livermore National Laboratory, Livermore, CA, USA
[3]Korea University, Seoul, South Korea [4]KAIST, Daejeon, South Korea

`{shachoi,tengyee}@korea.ac.kr jchoo@kaist.ac.kr`

Contribution

- Height-driven attention networks (HANet), a novel lightweight add-on module that can improves the performance by scaling the activation of channels according to the vertical position of a pixel.
- SOTA performance on the Cityscapes benchmark.
- Experimentally confirm that height position is crucial to improve the segmentation performance on urban scene.

Motivation

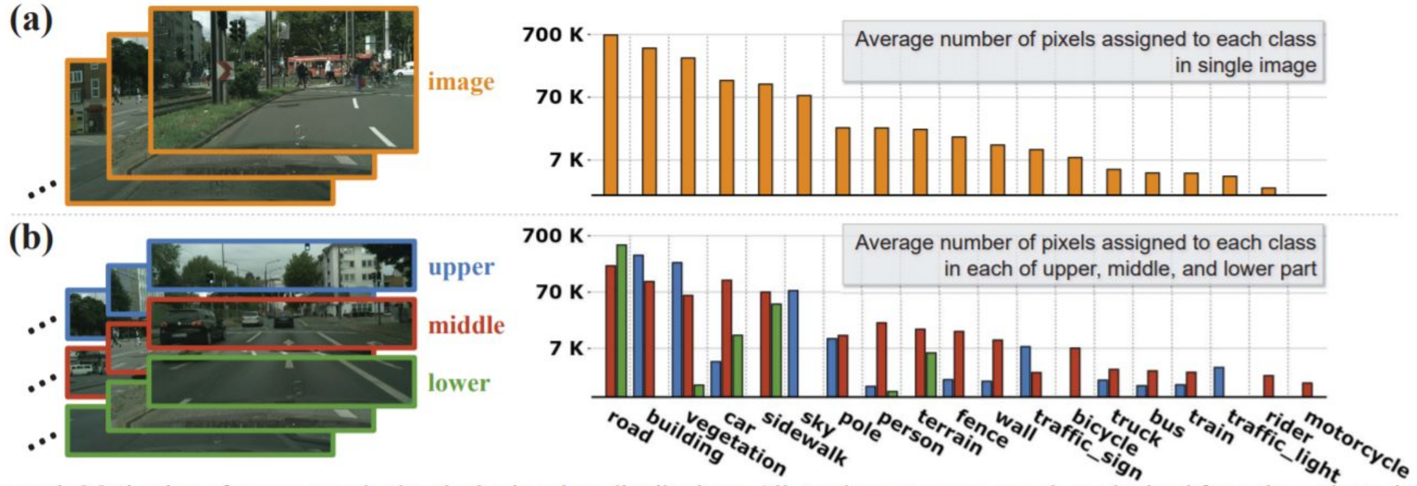


Figure 1: Motivation of our approach, the pixel-wise class distributions. All numbers are average values obtained from the entire training set of the Cityscapes dataset with its pixel-level class labels [12]. Note that there exists a total of 2048K pixels per image, and the y-axis is in log-scale. (a) Each bar represents the average number of pixels assigned to each class contained in a single image. For example, on average, about 685K pixels per image are assigned to the road class. (b) Each part of an image divided into three horizontal sections has a significantly different class distribution from each other. For example, the upper region has just 38 pixels of the road class, while the lower region has 480K pixels of it.

Method

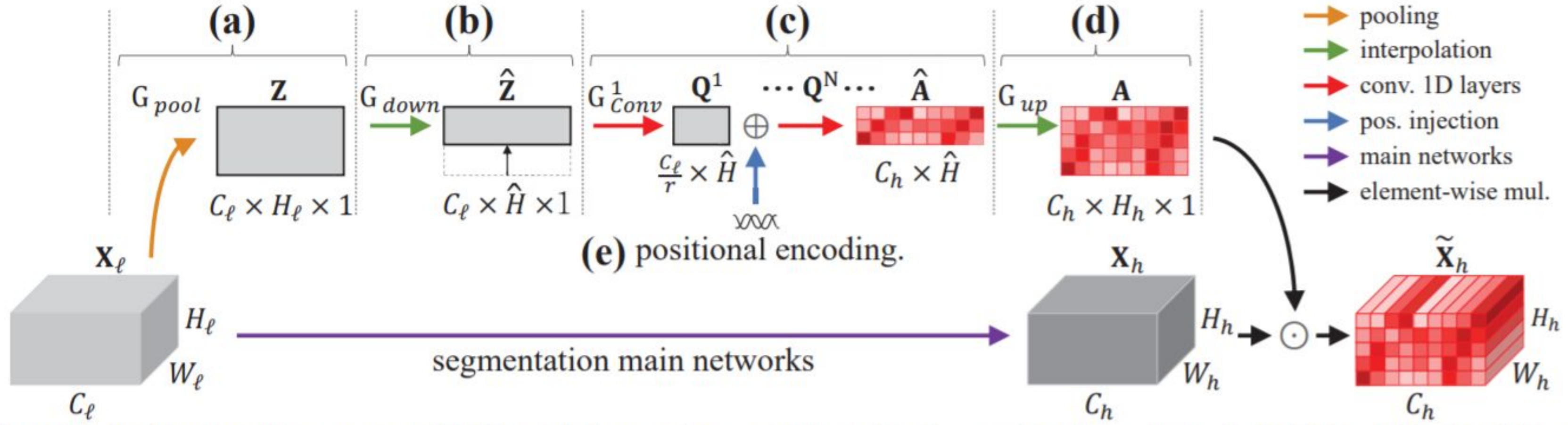
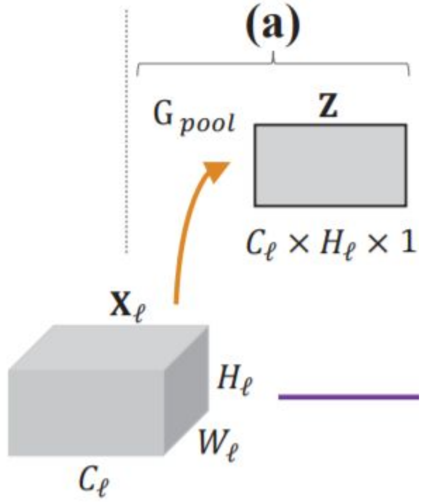


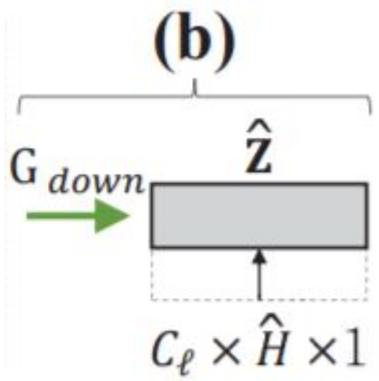
Figure 2: Architecture of our proposed HANet. Each operation *op* is notated as G_{op}, and feature maps are in bold–X_ℓ: lower-level feature map, Z: width-wise pooled X_ℓ, $\hat{Z}$: down-sampled Z, Q^n: n-th intermediate feature map of 1D convolution layers, $\hat{A}$: down-sampled attention map, A: final attention map, X_h: higher-level feature map, $\tilde{X}_h$: transformed new feature map. Details can be found in Section 3.1.

Method

- Do average pooling on W.

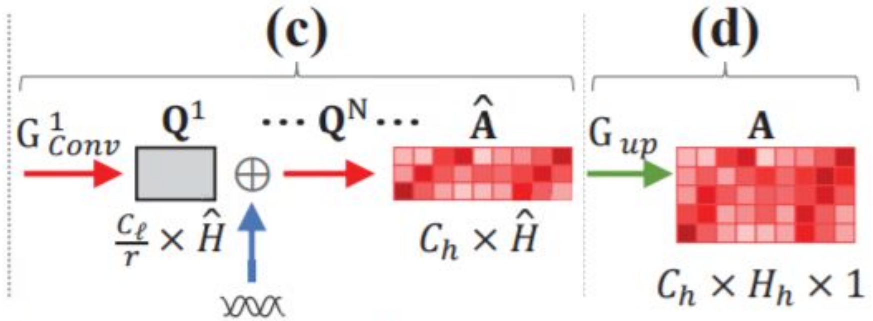


- Downsample.



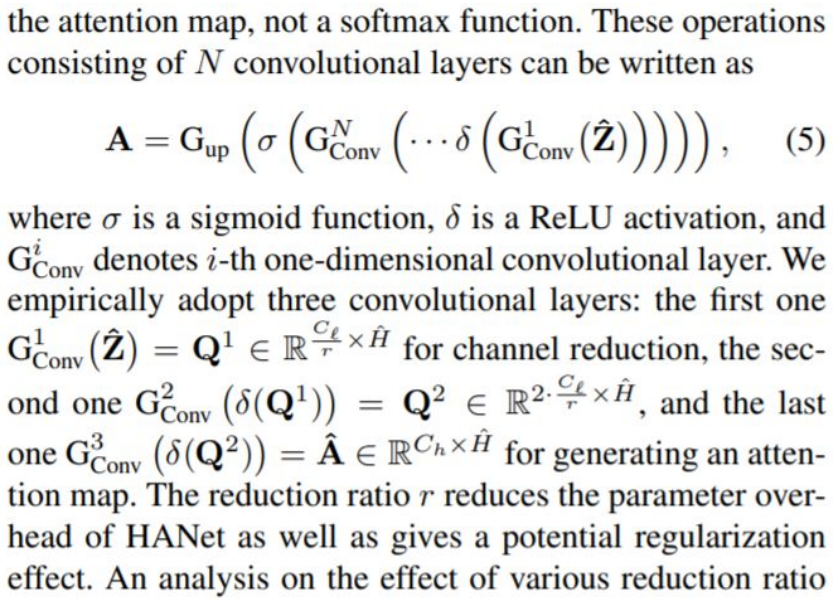
Method

- Compute attention map and incorporate positional encoding.



(e) positional encoding.

$$PE_{(p,2i)} = sin(p/100^{2i/C})$$
$$PE_{(p,2i+1)} = cos(p/100^{2i/C})$$


the attention map, not a softmax function. These operations consisting of N convolutional layers can be written as

$$\mathbf{A} = \mathrm{G}_{\mathrm{up}}\left(\sigma\left(\mathrm{G}^N_{\mathrm{Conv}}\left(\cdots\delta\left(\mathrm{G}^1_{\mathrm{Conv}}(\hat{\mathbf{Z}})\right)\right)\right)\right), \qquad (5)$$

where σ is a sigmoid function, δ is a ReLU activation, and $\mathrm{G}^i_{\mathrm{Conv}}$ denotes i-th one-dimensional convolutional layer. We empirically adopt three convolutional layers: the first one $\mathrm{G}^1_{\mathrm{Conv}}(\hat{\mathbf{Z}}) = \mathbf{Q}^1 \in \mathbb{R}^{\frac{C_\ell}{r} \times \hat{H}}$ for channel reduction, the second one $\mathrm{G}^2_{\mathrm{Conv}}(\delta(\mathbf{Q}^1)) = \mathbf{Q}^2 \in \mathbb{R}^{2 \cdot \frac{C_\ell}{r} \times \hat{H}}$, and the last one $\mathrm{G}^3_{\mathrm{Conv}}(\delta(\mathbf{Q}^2)) = \hat{\mathbf{A}} \in \mathbb{R}^{C_h \times \hat{H}}$ for generating an attention map. The reduction ratio r reduces the parameter overhead of HANet as well as gives a potential regularization effect. An analysis on the effect of various reduction ratio

Add-on module

- Example baseline: DeepLabv3+.
- Added 5 HANet modules.

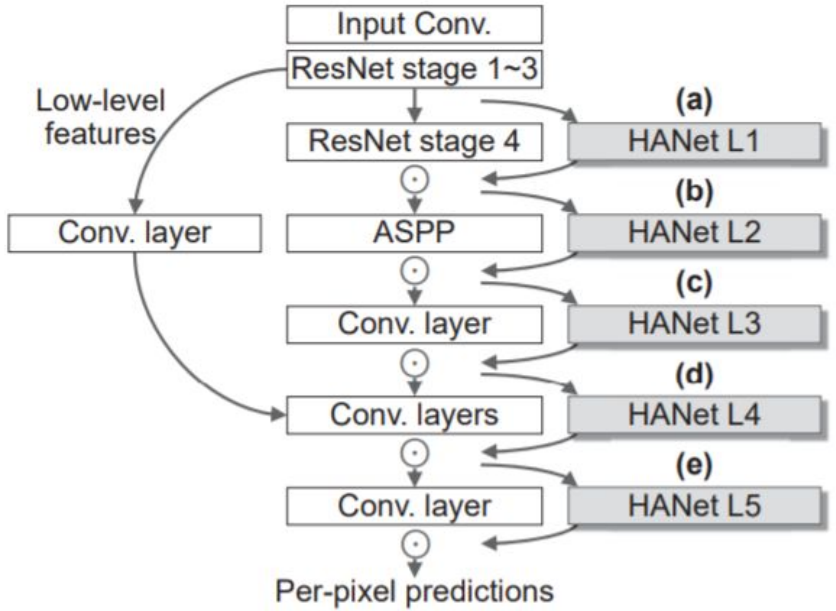
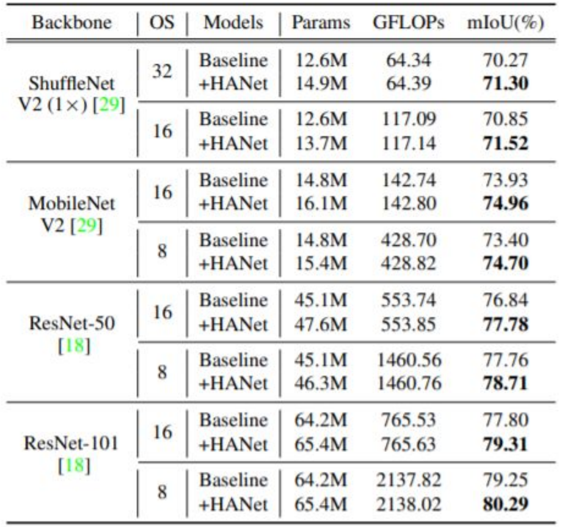


Figure 3: Semantic segmentation networks incorporating HANet in five different layers.

Experiment

- Dataset
  - Cityscapes
  - BDD100K



Backbone	OS	Models	Params	GFLOPs	mIoU(%)
ShuffleNet V2 (1×) [29]	32	Baseline	12.6M	64.34	70.27
		+HANet	14.9M	64.39	**71.30**
	16	Baseline	12.6M	117.09	70.85
		+HANet	13.7M	117.14	**71.52**
MobileNet V2 [29]	16	Baseline	14.8M	142.74	73.93
		+HANet	16.1M	142.80	**74.96**
	8	Baseline	14.8M	428.70	73.40
		+HANet	15.4M	428.82	**74.70**
ResNet-50 [18]	16	Baseline	45.1M	553.74	76.84
		+HANet	47.6M	553.85	**77.78**
	8	Baseline	45.1M	1460.56	77.76
		+HANet	46.3M	1460.76	**78.71**
ResNet-101 [18]	16	Baseline	64.2M	765.53	77.80
		+HANet	65.4M	765.63	**79.31**
	8	Baseline	64.2M	2137.82	79.25
		+HANet	65.4M	2138.02	**80.29**

Table 2: Comparison of mIoU, the number of model parameters and FLOPs between the baseline and HANet on Cityscapes validation set according to various backbone networks and output stride. Adding HANet to the baseline consistently increase the mIoU with minimal cost increase.

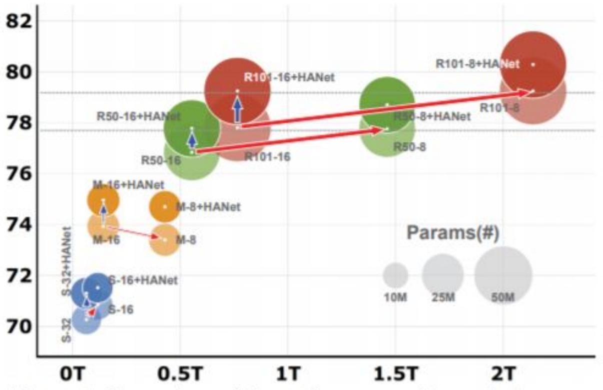


Figure 4: Comparison of the performance and complexity among the baseline and HANet on the various backbone networks. x-axis denotes teraFLOPs and y-axis denotes mIoU. The circle size denotes the number of model parameters. The texts in the colored circle indicate backbone networks, output stride, and whether HANet is adopted to the baseline. S, M, R50, and R101 denote ShuffleNetV2, MobileNetV2, ResNet-50, and -101, respectively. (e.g., S-16: Baseline, ShuffleNetV2, and output stride 16)

Experiment

Layers 1	2	3	4	5	Positional encoding	Ratio r	Pooling method	mIoU
✓					✓	32	average	78.52
✓	✓				✓	32	average	78.85
✓	✓	✓			✓	32	average	78.72
✓	✓	✓	✓		✓	32	average	**79.31**
✓	✓	✓	✓	✓	✓	32	average	78.79
✓	✓	✓	✓		✓	16	average	79.15
✓	✓	✓	✓		✓	64	average	79.08
✓	✓	✓	✓			32	average	78.25
✓	✓	✓	✓		✓	32	max	78.87
Baseline								77.80
Baseline + CoordConv [26] (Height + Width)								78.82

Table 3: Ablation studies and hyper-parameter impacts with regard to the HANet injected layers, using positional encodings or not, and channel reduction ratio. ResNet-101, output stride 16 on Cityscapes validation set.

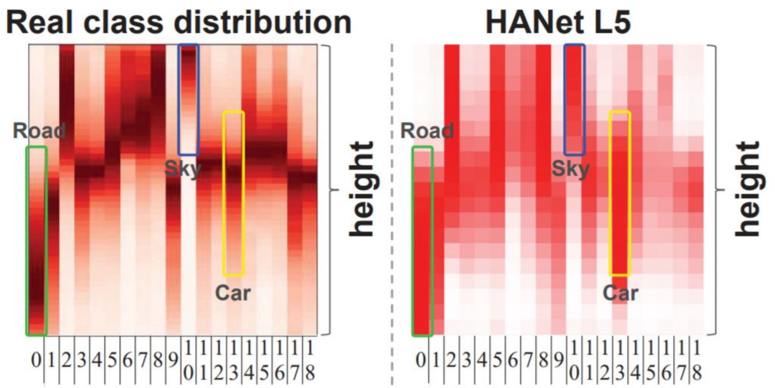
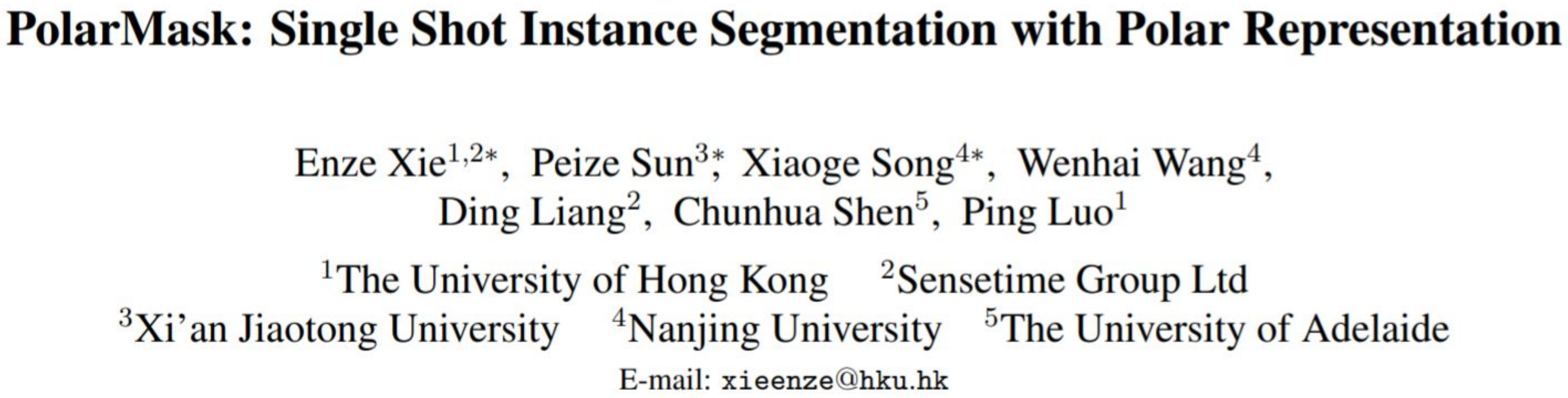


Figure 7: Height-wise class distributions and attention map visualization (L5). The number ranging from 0 to 18 indicates a different class. The darker it is colored, the higher probability (more pixels) assigned to a particular class. The attention visualization follows the patterns in the real class distribution.



PolarMask: Single Shot Instance Segmentation with Polar Representation

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

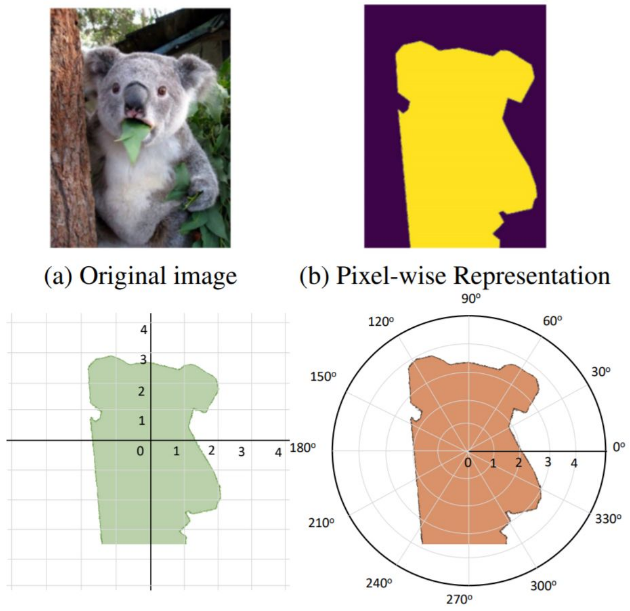
[1]The University of Hong Kong [2]Sensetime Group Ltd
[3]Xi'an Jiaotong University [4]Nanjing University [5]The University of Adelaide
E-mail: xieenze@hku.hk

https://zhuanlan.zhihu.com/p/84890413

Contribution

- A brand new framework for instance segmentation.
- Polar IoU Loss and Polar Centerness.
- Show that the complexity of instance segmentation, in terms of both design and computation complexity, can be the same as bounding box object detection.

Polar Representation



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

(c) Cartesian Representation (d) Polar Representation

Figure 1 – Instance segmentation with different mask representations. (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.

FCOS

- Proposed in *FCOS: Fully Convolutional One-Stage Object Detection*.
- https://arxiv.org/abs/1904.01355

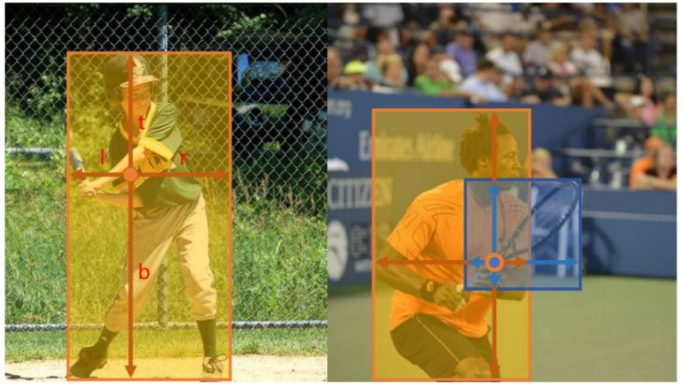


Figure 1 – As shown in the left image, FCOS works by predicting a 4D vector (l, t, r, b) encoding the location of a bounding box at each foreground pixel (supervised by ground-truth bounding box information during training). The right plot shows that when a location residing in multiple bounding boxes, it can be ambiguous in terms of which bounding box this location should regress.

Method

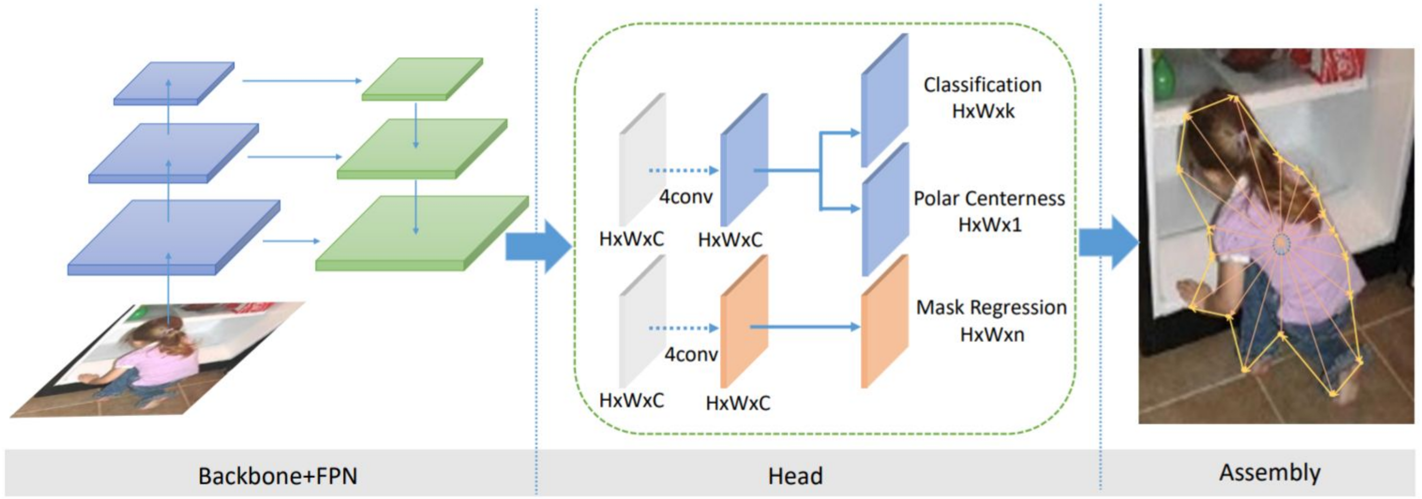


Figure 2 – The overall pipeline of PolarMask. The left part contains the backbone and feature pyramid to extract features of different levels. The middle part is the two heads for classification and polar mask regression. H, W, C are the height, width, channels of feature maps, respectively, and k is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

Instance modeling

- Polar representation: given a mask, sample a candidate center and emit n rays with same angle interval from the center to the counter.
- Use the mass center as the instance center.
- A point near center is regarded as center sample, while other points are negative sample.
- Given a contour and a center, the length of n rays are easy to calculate.

Centerness

- Down-weight the low-quality masks.

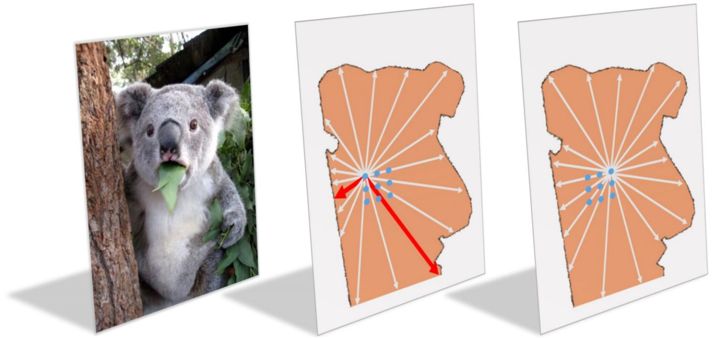


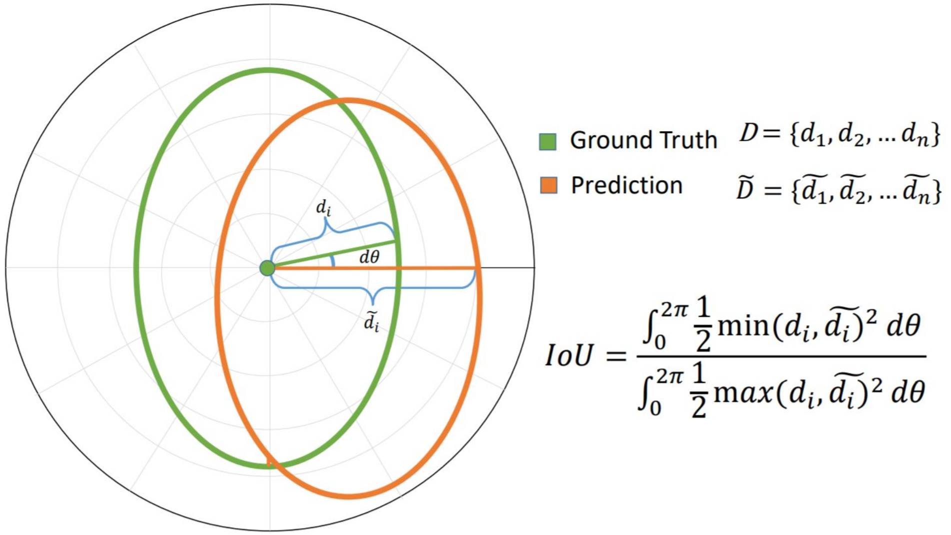
Figure 4 – Polar Centerness. Polar Centerness is used to down-weight such regression tasks as the high diversity of rays' lengths as shown in red lines in the middle plot. These examples are always hard to optimize and produce low-quality masks. During inference, the polar centerness predicted by the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

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

Mask Assembling

- Obtain a confidence score by multiplying centerness with classification result.
- Assemble masks from at most 1k top-scoring predictions per FPN level, after thresholding the confidence scores at 0.05.
- Non-maximum suppression

Polar IoU Loss



🟩 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. As shown in Figure 5, in the polar coordinate system, for one instance, mask IoU is calculated as follows:

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

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

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

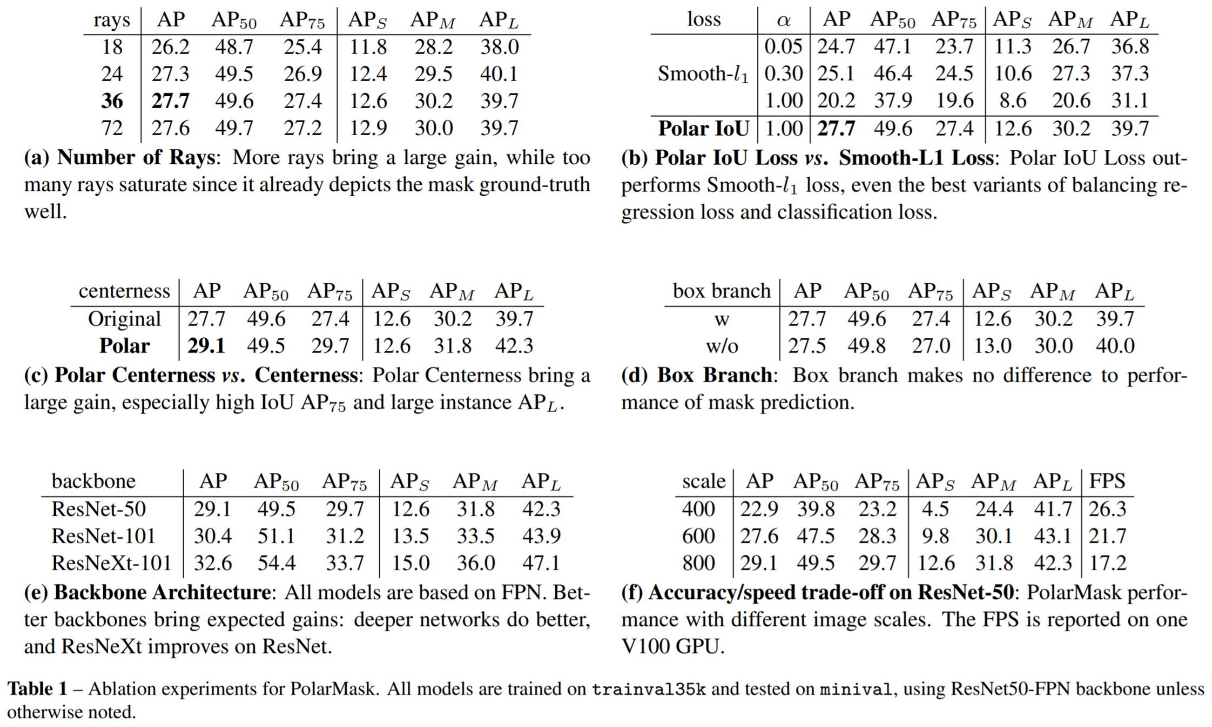
When N approaches infinity, the discrete form is equal to continuous form. We assume that the rays are uniformly emitted, so $\Delta\theta = \frac{2\pi}{N}$, which further simplifies the expression. We empirically observe that the power form has little impact on the performance (± 0.1 mAP difference) 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:

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

Experiment



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
	0.05	24.7	47.1	23.7	11.3	26.7	36.8
Smooth-l_1	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
Original	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.* 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	26.3
600	27.6	47.5	28.3	9.8	30.1	43.1	21.7
800	29.1	49.5	29.7	12.6	31.8	42.3	17.2

(f) Accuracy/speed trade-off on ResNet-50: PolarMask performance with different image scales. The FPS is reported on one V100 GPU.

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

Experiment

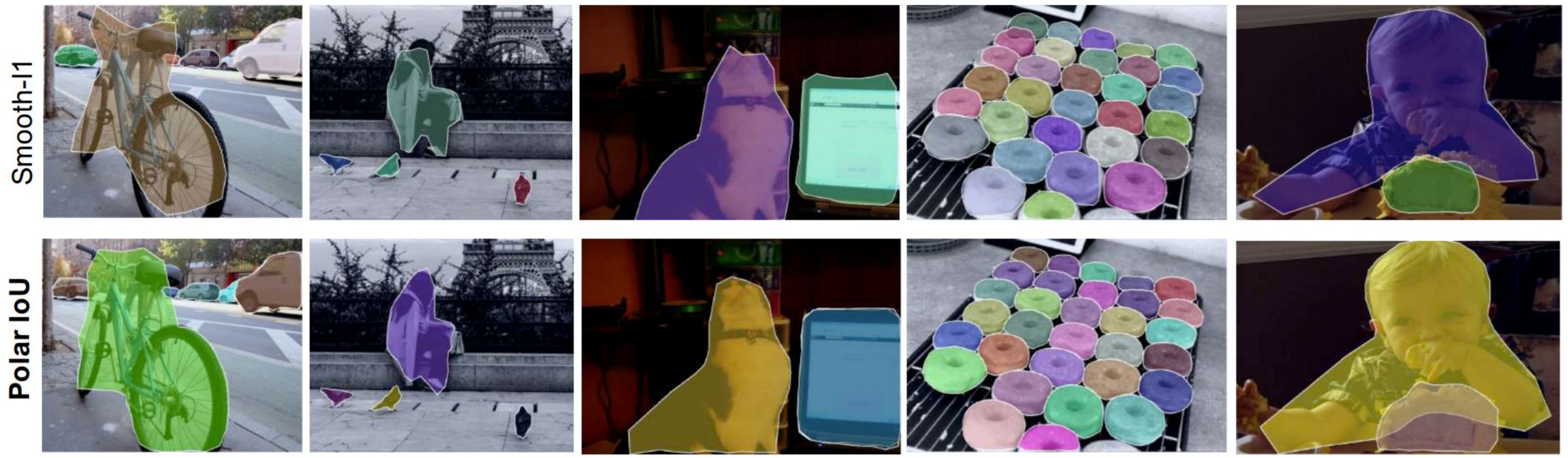


Figure 6 – Visualization of PolarMask with Smooth-l_1 loss and Polar IoU loss. Polar IoU Loss achieves to regress more accurate contour of instance while Smooth-l_1 Loss exhibits systematic artifacts.