

MaskLab: Instance Segmentation by Refining Object Detection with Semantic and Direction Features

Liang-Chieh Chen¹, Alexander Hermans^{2*}, George Papandreou¹, Florian Schroff¹, Peng Wang^{3*}, Hartwig Adam¹
 Google Inc.¹, RWTH Aachen University², UCLA³

Abstract

In this work, we tackle the problem of instance segmentation, the task of simultaneously solving object detection and semantic segmentation. Towards this goal, we present a model, called MaskLab, which produces three outputs: box detection, semantic segmentation, and direction prediction. Building on top of the Faster-RCNN object detector, the predicted boxes provide accurate localization of object instances. Within each region of interest, MaskLab performs foreground/background segmentation by combining semantic and direction prediction. Semantic segmentation assists the model in distinguishing between objects of different semantic classes including background, while the direction prediction, estimating each pixel's direction towards its corresponding center, allows separating instances of the same semantic class. Moreover, we explore the effect of incorporating recent successful methods from both segmentation and detection (e.g., atrous convolution and hypercolumn). Our proposed model is evaluated on the COCO instance segmentation benchmark and shows comparable performance with other state-of-art models.

1. Introduction

Deep Convolutional Neural Networks (ConvNets) [41, 40] have significantly improved the performance of computer vision systems. In particular, models based on Fully Convolutional Networks (FCNs) [64, 53] achieve remarkable results in object detection (localize instances) [22, 69, 25, 62, 51, 60, 19, 47] and semantic segmentation (identify semantic class of each pixel) [10, 46, 56, 52, 80, 73, 79, 54]. Recently, the community has been tackling the more challenging instance segmentation task [26, 28], whose goal is to localize object instances with pixel-level accuracy, jointly solving object detection and semantic segmentation.

Due to the intricate nature of instance segmentation, one could develop a system focusing on instance box-level detection first and then refining the prediction to obtain more



(a) Image

(b) Predicted masks

Figure 1. Instance segmentation aims to solve detection and segmentation jointly. We tackle this problem by refining the segmentation masks within predicted boxes (gray bounding boxes).

detailed mask segmentation, or conversely, one could target at sharp segmentation results before tackling the association problem of assigning pixel predictions to instances. The state-of-art instance segmentation model FCIS [44] employs the position-sensitive [16] inside/outside score maps to encode the foreground/background segmentation information. The usage of inside/outside score maps successfully segments foreground/background regions within each predicted bounding box, but it also doubles the number of output channels because of the redundancy of background encoding. On the other hand, the prior work of [70] produces three outputs: semantic segmentation, instance center direction (predicting pixel's direction towards its corresponding instance center), and depth estimation. However, complicate template matching is employed subsequently to decode the predicted direction for instance detection. In this work, we present MaskLab (short for Mask Labeling), seeking to combine the best from both detection-based and segmentation-based methods for solving instance segmentation.

Specifically, MaskLab builds on top of Faster R-CNN [62] and additionally produces two outputs: semantic segmentation and instance center direction [70]. The predicted boxes returned by Faster R-CNN bring object instances of different scales to a canonical scale, and MaskLab performs foreground/background segmentation within each predicted box by exploiting both semantic segmentation and direction prediction. The semantic segmentation prediction, encoding the pixel-wise classification information including

*Work done in part during an internship at Google Inc.

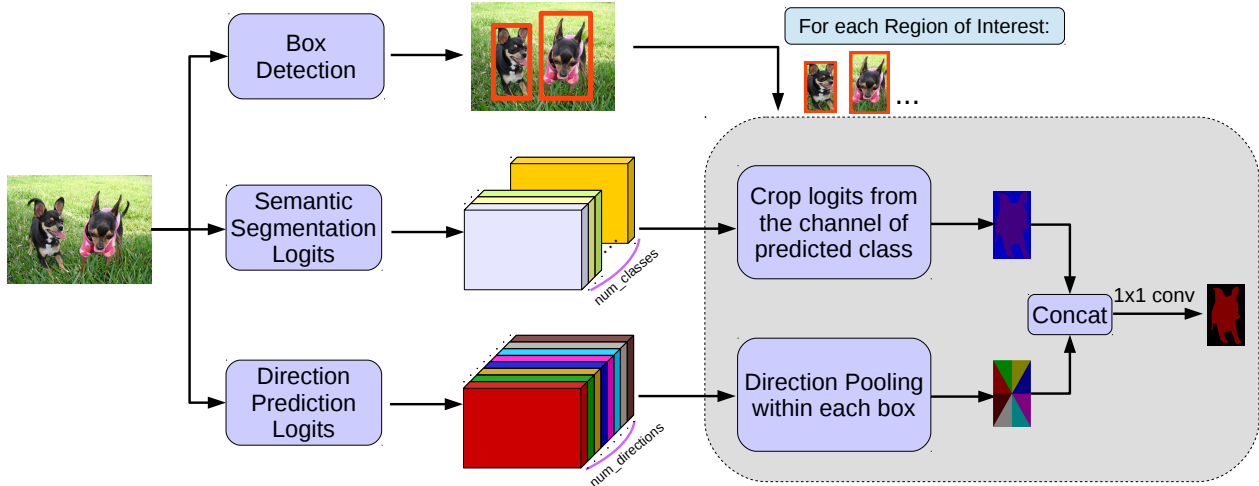


Figure 2. MaskLab generates three outputs, including refined box predictions (from Faster-RCNN), semantic segmentation logits (logits for pixel-wise classification), and direction prediction logits (logits for predicting each pixel’s direction toward its corresponding instance center). For each region of interest, we perform foreground/background segmentation by exploiting semantic segmentation and direction logits. Specifically, for the semantic segmentation logits, we pick the channel based on the predicted box label and crop the regions according to the predicted box. For the direction prediction logits, we perform the direction pooling to assemble the regional logits from each channel. These two cropped features are concatenated and passed through another 1×1 convolution for foreground/background segmentation.

background class, is adopted to distinguish between objects of different semantic classes (e.g., person and background), and thus removes the duplicate background encoding in [44]. Additionally, direction prediction is used to separate object instances of the same semantic label. Our model employs the same assembling operation in [16, 44] to collect the direction information and thus gets rid of the complicate template matching used in [70]. Furthermore, motivated by the recent advances in both segmentation and detection, MaskLab further incorporates atrous convolution [11] to extract denser features maps, hypercolumn features [29] for refining mask segmentation [21], multi-grid [71, 20, 12] for capturing different scales of context, and a new TensorFlow operation [1], deformable crop and resize, inspired by the deformable pooling operation [20].

We demonstrate the effectiveness of the proposed model on the challenging COCO instance segmentation benchmark [48]. Our proposed model, MaskLab, shows comparable performance with other state-of-art models in terms of both mask segmentation (e.g., FCIS [44] and Mask R-CNN [31]) and box detection (e.g., G-RMI [35] and TDM [66]). Finally, we elaborate on the implementation details and provide detailed ablation studies of the proposed model.

2. Related Work

In this work, we categorize current instance segmentation methods based on deep neural networks into two types, depending on how the method approaches the problem by starting from either detection or segmentation modules.

Detection-based methods: This type of methods ex-

ploits state-of-art detection models (e.g., Fast-RCNN [25], Faster-RCNN [62] or R-FCN [19]) to either classify mask regions or refine the predicted boxes to obtain masks. There have been several methods developed for mask proposals, including CPMC [9], MCG [3], DeepMask [58], SharpMask [59], and instance-sensitive FCNs [16]. Recently, Zhang and He [76] propose a free-form deformation network to refine the mask proposals. Coupled with the mask proposals, SDS [28, 14] and CFM [17] incorporate mask-region features to improve the classification accuracy, while [29] exploit hypercolumn features (i.e., features from the intermediate layers). Li *et al.* [43] iteratively apply the prediction. Zagoruyko *et al.* [75] exploit object context at multiple scales. The work of MNC [18] shows promising results by decomposing instance segmentation into three sub-problems including box localization, mask refinement and instance classification. Hayer *et al.* [30] improve MNC by recovering the mask boundary error resulted from box prediction. Arnab *et al.* [4, 5] apply higher-order Conditional Random Fields (CRFs) to refine the mask results. FCIS [44], the first Fully Convolutional Network (FCN) [53] for instance segmentation, enriches the position-sensitive score maps from [16] by further considering inside/outside score maps. Mask-RCNN [31], built on top of FPN [47], adds another branch to obtain refined mask results from Faster-RCNN box prediction and demonstrates outstanding performance.

Segmentation-based methods: This type of methods generally adopt a two-stage processing, including segmentation and clustering. Pixel-level predictions are obtained by the segmentation module before the clustering process is applied to group them together for each object instance.

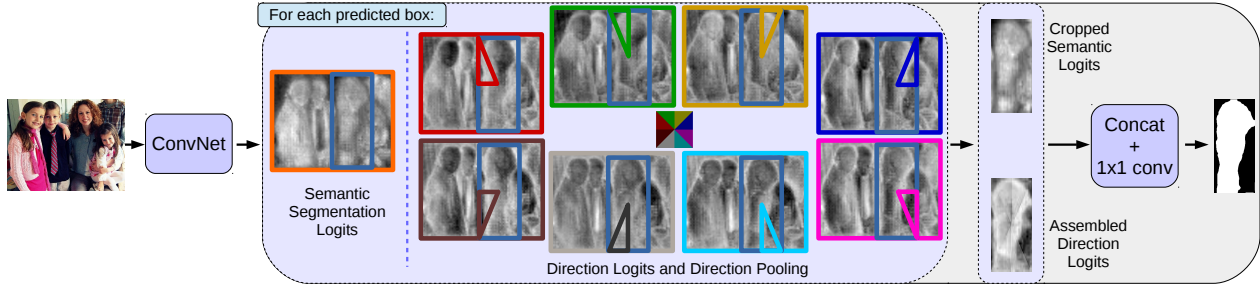


Figure 3. Semantic segmentation logits and direction prediction logits are used to perform foreground/background segmentation within each predicted box. In particular, segmentation logits are able to distinguish between instances of different semantic classes (*e.g.*, person and background), while direction logits (directions are color-coded) further separate instances of the same semantic class (*e.g.*, two persons in the predicted blue box). In the assembling operation, regional logits (the color triangular regions) are copied from each direction channel, similar to [16, 44]. For example, the region specified by the red triangle copies the logits from the red direction channel encoding instance direction from 0 degree to 45 degree. Note the weak activations in the pink channel encoding instance direction from 180 degree to 225 degree.

Proposal-free network [45] applies spectral clustering to group segmentation results from DeepLab [10], while Zhang *et al.* [78] exploit depth ordering within an image patch. In addition to semantic and depth information, Uhrig *et al.* [70] further train an FCN to predict instance center direction. Zhang *et al.* [77] propose a novel fully connected CRF [39] (with fast inference by permutohedral lattice [2]) to refine the results. Liu *et al.* [50] segment objects in multi-scale patches and aggregate the results. Levinkov *et al.* [42] propose efficient local search algorithms for instance segmentation. Wu *et al.* [72] exploit a localization network for grouping, while Bai and Urtasun [6] adopt a Watershed Transform Net. Furthermore, Liu *et al.* [49] propose to sequentially solve the grouping problem and gradually compose object instances. [38, 36] exploit boundary detection information, while [55, 23, 8] propose to cluster instances w.r.t. the learned embedding values.

In addition to the two categories, there is other interesting work. For example, [63, 61] propose recurrent neural networks to sequentially segment an instance at a time. [37] propose a weakly supervised instance segmentation model given only bounding box annotations.

Our proposed MaskLab model combines the advantages from both detection-based and segmentation-based methods. In particular, MaskLab builds on top of Faster-RCNN [62] and additionally incorporates semantic segmentation (to distinguish between instances of different semantic classes, including background class) and direction features [70] (to separate instances of the same semantic label). Our work is most similar to FCIS [44], Mask R-CNN [31], and the work of [70]; we build on top of Faster R-CNN [62] instead of R-FCN [19] (and thus replace the complicated template matching for instance detection in [70]), exploit semantic segmentation prediction to remove duplicate background encoding in the inside/outside score maps, and we also simplify the position-sensitive pooling to direction pooling.

3. MaskLab

Overview: Our proposed model, MaskLab, employs ResNet-101 [32] as feature extractor. It consists of three components with all features shared up to conv4 (or res4x) block and one extra duplicate conv5 (or res5x) block is used for the box classifier in Faster-RCNN [62]. Note that the original conv5 block is shared for both semantic segmentation and direction prediction. As shown in Fig. 2, MaskLab, built on top of Faster-RCNN [62], produces box prediction (in particular, refined boxes after the box classifier), semantic segmentation logits (logits for pixel-wise classification) and direction prediction logits (logits for predicting each pixel’s direction towards its corresponding instance center [70]). Semantic segmentation logits and direction prediction logits are computed by another 1×1 convolution added after the last feature map in the conv5 block of ResNet-101. Given each predicted box (or region of interest), we perform foreground/background segmentation by exploiting those two logits. Specifically, we apply a class-agnostic (*i.e.*, with weights shared across all classes) 1×1 convolution on the concatenation of (1) cropped semantic logits from the semantic channel predicted by Faster-RCNN and (2) cropped direction logits after direction pooling.

Semantic and direction features: MaskLab generates semantic segmentation logits and direction prediction logits for an image. The semantic segmentation logits are used to predict pixel-wise semantic labels, which are able to separate instances of different semantic labels, including the background class. On the other hand, the direction prediction logits are used to predict each pixel’s direction towards its corresponding instance center and thus they are useful to further separate instances of the same semantic labels.

Given the predicted boxes and labels from the box prediction branch, we first select the channel w.r.t. the predicted label (*e.g.*, the person channel) from the semantic segmentation logits, and crop the regions w.r.t. the predicted box. In order

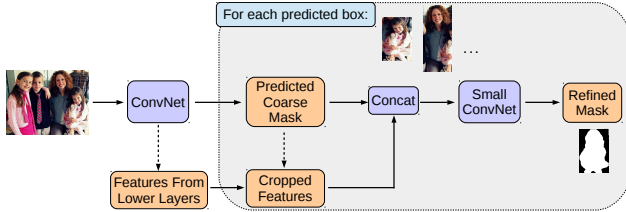
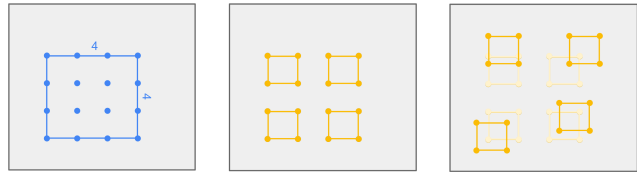


Figure 4. Mask refinement. Hypercolumn features are concatenated with the coarse predicted mask and then fed to another small ConvNet to produce the final refined mask predictions.

to exploit the direction information, we perform the same assembling operation in [16, 44] to gather regional logits (specified by the direction) from each direction channel. The cropped semantic segmentation logits along with the pooled direction logits are then used for foreground/background segmentation. We illustrate the details in Fig. 3, which shows that the segmentation logits for ‘person’ clearly separate the person class from background and the tie class, and the direction logits are able to predict the pixel’s direction towards its instance center. After assembling the direction logits, the model is able to further separate the two persons within the specified box region. Note that our proposed direction prediction logits are class-agnostic instead of having the logits for each semantic class as in FCIS [44], yielding more compact models. Specifically, for mask segmentation with K classes, our model requires $(K + 32)$ channels (K for semantic segmentation and 32 for direction pooling), while [44] outputs $2 \times (K + 1) \times 49$ channels (2 for inside/outside score maps and 49 for position grids).

Mask refinement: Motivated by [21] which applies another network consisting of only few layers for segmentation refinement, we further refine the predicted coarse masks by exploiting the hypercolumn features [29]. Specifically, as shown in Fig. 4, the generated coarse mask logits (by only exploiting semantic and direction features) are concatenated with features from lower layers of ResNet-101, which are then processed by three extra convolutional layers in order to predict the final mask.

Deformable crop and resize: Following Dai *et al.* [20], who demonstrate significant improvement in object detection by deforming convolution and pooling operations, we modify the key TensorFlow operation used for box classification, “crop and resize” (similar to RoIAlign in Mask R-CNN [31]), to support deformation as well. As shown in Fig. 5, “crop and resize” first crops a specified bounding box region from the feature maps and then bilinearly resizes them to a specified size (*e.g.*, 4×4). We further divide the regions into several sub-boxes (*e.g.*, 4 sub-boxes and each has size 2×2) and employ another small network to learn the offsets for each sub-box. Finally, we perform “crop and resize” again w.r.t. each deformed sub-box. In summary, we use “crop and resize” twice to implement the deformable pooling in [20].



(a) Crop and resize (b) 2×2 sub-boxes (c) Deformed sub-boxes

Figure 5. Deformable crop and resize. (a) The operation, crop and resize, crops features within a bounding box region and resizes them to a specified size 4×4 . (b) The 4×4 region is then divided into 4 small sub-boxes, and each has size 2×2 . (c) Another small network is applied to learn the offsets of each sub-box. Then we perform crop and resize again w.r.t. to the deformed sub-boxes.

4. Experimental Evaluation

We conduct experiments on the COCO dataset [48]. Our proposed model is implemented in TensorFlow [1] on top of the object detection library developed by [35].

4.1. Implementation Details

We employ the same hyper-parameter settings as in [35, 67], and only discuss the main difference below.

Atrous convolution: We apply the atrous convolution [34, 27, 64, 57], which has been successfully explored in semantic segmentation [13, 79, 12], object detection [19, 35] and instance segmentation [78, 44], to extract denser feature maps. Specifically, we extract features with $output_stride = 8$ ($output_stride$ denotes the ratio of input image spatial resolution to final output resolution).

Weight initialization: For the 1×1 convolution applied to the concatenation of semantic and direction features, we found that the training converges faster by initializing the convolution weights to be (0.5, 1), putting a slightly larger weight on the direction features, which is more important in instance segmentation, as shown in the experimental results.

Mask training: During training, only groundtruth boxes are used to train the branches that predict semantic segmentation logits and direction logits, since direction logits may not align well with instance center if boxes are jittered. We employ sigmoid function to estimate both the coarse and refined mask results. Our proposed model is trained end-to-end without piecewise pretraining of each component.

4.2. Quantitative Results

We first report the ablation studies on a *minimal* set and then evaluate the best model on *test-dev* set, with the metric mean average precision computed using mask IoU.

Mask crop size: The TensorFlow operation, “crop and resize”, is used at least in two places: one for box classification and one for cropping semantic and direction features for foreground/background segmentation (another one for deformed sub-boxes if “deformable crop and resize” is used). In the former case, we use the same setting as in [35, 67],

Mask crop size	mAP@0.5
21	50.92%
41	51.29%
81	51.17%
161	51.36%
321	51.24%

Table 1. Using crop size = 41 is sufficient for mask segmentation.

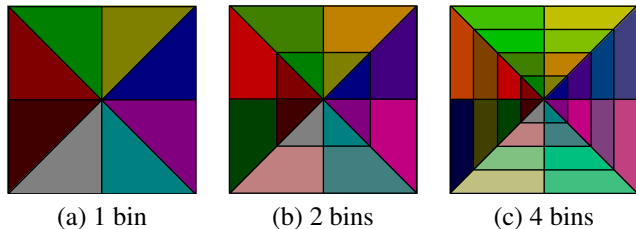


Figure 6. We quantize the distance within each direction region. In (b), we split each original direction region into 2 regions. Our final model uses 4 bins for distance quantization as shown in (c).

while in the latter case, the crop size determines the mask segmentation resolution. Here, we experiment with the effect of using different crop size in Tab. 1 and observe that using crop size more than 41 does not change the performance significantly and thus we use 41 throughout the experiments.

Effect of semantic and direction features: In Tab. 2, we experiment with the effect of semantic and direction features. Given only semantic segmentation features, the model attains an mAP@0.75 performance of 24.44%, while using only direction features the performance improves to 27.4%, showing that direction feature is more important than the semantic segmentation feature. When employing both features, we achieve 29.72%. We observe that the performance can be further improved if we also quantize the distance in the direction pooling. As illustrated in Fig. 6, we also quantize the distance with different number of bins. For example, when using 2 bins, we split the same direction region into 2 regions. We found that using 4 bins can further improves performance to 30.57%. Note that quantizing the distance bins improves more at high mAP threshold (*cf.* mAP@0.5 and mAP@0.75 in Tab. 2). In the case of using x distance bins, the channels of direction logits become $8 \times x$, since we use 8 directions by default (*i.e.*, 360 degree is quantized into 8 directions). Thus, our model generates $32 = 8 \times 4$ channels for direction pooling in the end.

Number of directions: In Tab. 3, we explore the effect of different numbers of directions for quantizing the 360 degree. We found that using 8 directions is sufficient to deliver good performance, when adopting 4 bins for distance quantization. Our model thus uses $32 = 8 \times 4$ (8 for direction and 4 for distance quantization) channels for direction pooling throughout the experiments.

Mask refinement: We adopt a small ConvNet consisting

Semantic	Direction	mAP@0.5	mAP@0.75
✓		48.41%	24.44%
	✓(1)	50.21%	27.40%
✓	✓(1)	51.83%	29.72%
✓	✓(4)	52.26%	30.57%

Table 2. Effect of semantic and direction features. Direction features are more important than semantic segmentation features in the model, and the best performance is obtained by using both features and adopting 4 bins to quantize the distance in direction pooling. We show number of bins for distance quantization in parentheses.

Distance bins	Directions	mAP@0.5	mAP@0.75
4	2	53.51%	33.80%
4	4	53.85%	34.39%
4	6	54.10%	34.86%
4	8	54.13%	34.82%

Table 3. Effect of different numbers of directions (*i.e.*, how many directions for quantizing the 360 degree) when using four bins for distance quantization.

conv1	conv2	conv3	mAP@0.5	mAP@0.75
			52.26%	30.57%
✓			52.68%	32.92%
✓	✓		53.26%	33.89%
✓	✓	✓	52.55%	32.88%

Table 4. Mask refinement. The best performance is obtained when using features from conv1 and conv2 (*i.e.*, last feature map in res2x block). Note conv3 denotes the last feature map in res3x block.

of three 5×5 convolution layers with 64 filters. We have experimented with replacing the small ConvNet with other structures (*e.g.*, more layers and more filters) but have not observed any significant difference. In Tab. 4, we experiment with different features from lower-level of ResNet-101. Using conv1 (the feature map generated by the first convolution) improves the mAP@0.75 performance to 32.92% from 30.57%, while using both conv1 and conv2 (*i.e.*, the last feature map in res2x block) obtains the best performance of 33.89%. We have observed no further improvement when adding more lower-level features.

Multi-grid: Motivated by the success of employing a hierarchy of different atrous rates in semantic segmentation [71, 20, 12], we modify the atrous rates in (1) the last residual block shared for predicting both semantic and direction features, and (2) the block for box classifier. Note that there are only three convolutions in those blocks. As shown in Tab. 5, it is more effective to apply different atrous rates for the box classifier. We think current evaluation metric (mAP^r) favors detection-based methods (as also pointed out by [6]) and thus it is more effective to improve the detection

		Box Classifier		
		(1, 1, 1)	(1, 2, 1)	(1, 2, 4)
Sem/Dir	(4, 4, 4)	34.82%	35.59%	35.35%
	(4, 8, 4)	35.07%	35.60%	35.78%
	(4, 8, 16)	34.89%	35.43%	35.51%

Table 5. Multi-grid performance (mAP@0.75). Within the parentheses, we show the three atrous rates used for the three convolutions in the residual block. It is effective to adopt different atrous rates for the box classifier. Further marginal improvement is obtained when we also change the atrous rates in the last block that is shared by semantic segmentation and direction prediction logits.

branch over the segmentation branch in our proposed model.

Pretrained network: We experimentally found that it is beneficial to pretrain the network. Recall that we duplicate one extra conv5 (or res5x) block in original ResNet-101 for box classification. As shown in Tab. 6, initializing the box classifier in Faster R-CNN with the ImageNet pretrained weights improves the performance from 33.89% to 34.82% (mAP@0.75). If we further pretrain ResNet-101 on the COCO semantic segmentation annotations and employ it as feature extractor, the model yields about 1% improvement. This finding bears a similarity to [7] which adopts the semantic segmentation regularizer.

Putting everything together: We then employ the best multi-grid setting from Tab. 5 and observe about 0.7% improvement (mAP@0.75) over the one pretrained with segmentation annotations, as shown in Tab. 6. Following [47, 31], if the input image is resized to have a shortest side of 800 pixels and the Region Proposal Network adopts 5 scales, we observe another 1% improvement. Using the implemented “deformable crop and resize” brings extra 1% improvement. Additionally, we employ scale augmentation, specifically random scaling of inputs during training (with shortest side randomly selected from {480, 576, 688, 800, 930}), and attain performance of 40.41% (mAP@0.75). Finally, we exploit the model that has been pretrained on the JFT-300M dataset [33, 15, 67], containing 300M images and more than 375M noisy image-level labels, and achieve performance of 41.59% (mAP@0.75).

Atrous convolution for denser feature maps: We employ atrous convolution, a powerful tool to control output resolution, to extract denser feature maps with $output_stride = 8$. We have observed that our performance drops from 40.41% to 38.61% (mAP@0.75), if we change $output_stride = 16$.

Test-dev mask results: After finalizing the design choices on the *minival* set, we then evaluate our model on the *test-dev* set. As shown in Tab. 7, our MaskLab model outperforms FCIS+++ [44], although FCIS+++ employs scale augmentation and on-line hard example mining [65] during training as well as multi-scale processing and horizontal flip

BC	Seg	MG	Anc	DC	RS	JFT	mAP@0.5	mAP@0.75
							53.26%	33.89%
✓							54.13%	34.82%
✓	✓						55.03%	35.91%
✓	✓	✓					55.64%	36.65%
✓	✓	✓	✓				57.44%	37.57%
✓	✓	✓	✓	✓			58.69%	38.61%
✓	✓	✓	✓	✓	✓		60.55%	40.41%
✓	✓	✓	✓	✓	✓	✓	61.80%	41.59%

Table 6. **BC:** Initialize the Box Classifier branch with ImageNet pretrained model. **Seg:** Pretrain the whole model on COCO semantic segmentation annotations. **MG:** Employ multi-grid in last residual block. **Anc:** Use (800, 1200) and 5 anchors. **DC:** Adopt deformable crop and resize. **RS:** Randomly scale inputs during training. **JFT:** Further pretrain the model on JFT dataset.

during test. Our ResNet-101 based model performs better than the ResNet-101 based Mask R-CNN [31], and attains similar performance as the ResNet-101-FPN based Mask R-CNN. Our ResNet-101 based model with scale augmentation during training, denoted as MaskLab+ in the table, performs 1.9% better, attaining similar mAP with Mask R-CNN built on top of the more powerful ResNeXt-101-FPN [47, 74]. Furthermore, pretraining MaskLab+ on the JFT dataset achieves performance of 38.1% mAP.

Test-dev box results: We also show box detection results on COCO test-dev in Tab. 8. Our ResNet-101 based model even without scale augmentation during training performs better than G-RMI [35] and TDM [66] which employ more expensive yet powerful Inception-ResNet-v2 [68] as feature extractor. All our model variants perform comparably or better than Mask R-CNN variants in the box detection task. Our best single-model result is obtained with scale augmentation during training, 41.9% mAP with an ImageNet pretrained network and 43.0% mAP with a JFT pretrained network.

4.3. Qualitative Results

Semantic and direction features: In Fig. 7, we visualize the ‘person’ channel in the learned semantic segmentation logits. We have observed that there can be some high activations in the non-person regions (*e.g.*, regions that are near elephant’s legs and kite), since the semantic segmentation branch is only trained with groundtruth boxes without any negative ones. This, however, is being handled by the box detection branch which filters out wrong box predictions. More learned semantic segmentation and direction prediction logits are visualized in Fig. 3.

Deformable crop and resize: In Fig. 8, we visualize the learned deformed sub-boxes. Interestingly, unlike the visualization results of deformable pooling in [20] which learns to focus on object parts, our sub-boxes are deformed in a circle-shaped arrangement, attempting to capture longer context for box classification. We note that incorporating context to improve detection performance has been used in, *e.g.*, [24, 81, 75], and our model is also able to learn this.

Method	Feature Extractor	mAP	mAP@0.5	mAP@0.75	mAP _S	mAP _M	mAP _L
FCIS [44]	ResNet-101	29.2%	49.5%	-	-	-	-
FCIS+++ [44]	ResNet-101	33.6%	54.5%	-	-	-	-
Mask R-CNN [31]	ResNet-101	33.1%	54.9%	34.8%	12.1%	35.6%	51.1%
Mask R-CNN [31]	ResNet-101-FPN	35.7%	58.0%	37.8%	15.5%	38.1%	52.4%
Mask R-CNN [31]	ResNeXt-101-FPN	37.1%	60.0%	39.4%	16.9%	39.9%	53.5%
MaskLab	ResNet-101	35.4%	57.4%	37.4%	16.9%	38.3%	49.2%
MaskLab+	ResNet-101	37.3%	59.8%	39.6%	19.1%	40.5%	50.6%
MaskLab+	ResNet-101 (JFT)	38.1%	61.1%	40.4%	19.6%	41.6%	51.4%

Table 7. Instance segmentation single model *mask* mAP on COCO test-dev. **MaskLab+**: Employ scale augmentation during training.

Method	Feature Extractor	mAP	mAP@0.5	mAP@0.75	mAP _S	mAP _M	mAP _L
G-RMI [35]	Inception-ResNet-v2	34.7%	55.5%	36.7%	13.5%	38.1%	52.0%
TDM [66]	Inception-ResNet-v2	37.3%	57.8%	39.8%	17.1%	40.3%	52.1%
Mask R-CNN [31]	ResNet-101-FPN	38.2%	60.3%	41.7%	20.1%	41.1%	50.2%
Mask R-CNN [31]	ResNeXt-101-FPN	39.8%	62.3%	43.4%	22.1%	43.2%	51.2%
MaskLab	ResNet-101	39.6%	60.2%	43.3%	21.2%	42.7%	52.4%
MaskLab+	ResNet-101	41.9%	62.6%	46.0%	23.8%	45.5%	54.2%
MaskLab+	ResNet-101 (JFT)	43.0%	63.9%	47.1%	24.8%	46.7%	55.2%

Table 8. Object detection single model *box* mAP on COCO test-dev. **MaskLab+**: Employ scale augmentation during training.

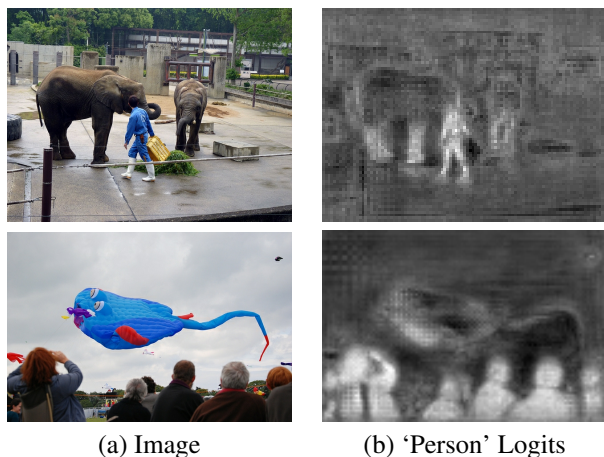


Figure 7. ‘Person’ channel in the predicted semantic segmentation logits. Note the high activations on non-person regions, since the semantic segmentation branch is only trained with groundtruth boxes. This, however, is being handled by the box detection branch which filters out wrong box predictions.

Predicted masks: We show some qualitative results produced by our proposed model in Fig. 9. We further visualize our failure mode in the last row, mainly resulting from detection failure (e.g., missed-detection and wrong class prediction) and segmentation failure (e.g., coarse boundary result).



Figure 8. Visualization of learned deformed sub-boxes. The 49 (arranged in a 7×7 grid) sub-boxes (each has size 2×2) are color-coded w.r.t. the top right panel (e.g., the top-left sub-box is represented by light blue color). Our “deformable crop and resize” tend to learn circle-shaped context for box classification.

5. Conclusion

In this paper, we have presented a model, called MaskLab, that produces three outputs: box detection, semantic segmentation and direction prediction, for solving the problem of instance segmentation. MaskLab, building on top of state-of-art detector, performs foreground/background segmentation by utilizing semantic segmentation and direction prediction. We have demonstrated the effectiveness of MaskLab on the challenging COCO instance segmentation benchmark and shown promising results.

Acknowledgments We would like to acknowledge valuable discussions with Xuming He and Chen Sun, comments from Menglong Zhu and Xiao Zhang, and the support from the Google Mobile Vision team.

References

- [1] M. Abadi, A. Agarwal, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv:1603.04467*, 2016.
- [2] A. Adams, J. Baek, and M. A. Davis. Fast high-dimensional filtering using the permutohedral lattice. In *Eurographics*, 2010.
- [3] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014.
- [4] A. Arnab, S. Jayasumana, S. Zheng, and P. Torr. Higher order conditional random fields in deep neural networks. In *ECCV*, 2016.
- [5] A. Arnab and P. Torr. Pixelwise instance segmentation with a dynamically instantiated network. In *CVPR*, 2017.
- [6] M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In *CVPR*, 2017.
- [7] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *CVPR*, 2016.
- [8] B. D. Brabandere, D. Neven, and L. V. Gool. Semantic instance segmentation with a discriminative loss function. *arXiv:1708.02551*, 2017.
- [9] J. Carreira and C. Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts. *PAMI*, 34(7):1312–1328, 2012.
- [10] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.
- [11] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017.
- [12] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Re-thinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017.
- [13] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille. Attention to scale: Scale-aware semantic image segmentation. In *CVPR*, 2016.
- [14] Y.-T. Chen, X. Liu, and M.-H. Yang. Multi-instance object segmentation with occlusion handling. In *CVPR*, 2015.
- [15] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv:1610.02357*, 2016.
- [16] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016.
- [17] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *CVPR*, 2015.
- [18] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016.
- [19] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, 2016.
- [20] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, 2017.
- [21] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015.
- [22] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014.
- [23] A. Fathi, Z. Wojna, V. Rathod, P. Wang, H. O. Song, S. Guadarrama, and K. P. Murphy. Semantic instance segmentation via deep metric learning. *arXiv:1703.10277*, 2017.
- [24] S. Gidaris and N. Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *ICCV*, 2015.
- [25] R. Girshick. Fast r-cnn. In *ICCV*, 2015.
- [26] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.
- [27] A. Giusti, D. Ciresan, J. Masci, L. Gambardella, and J. Schmidhuber. Fast image scanning with deep max-pooling convolutional neural networks. In *ICIP*, 2013.
- [28] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.
- [29] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015.
- [30] Z. Hayder, X. He, and M. Salzmann. Boundary-aware instance segmentation. In *CVPR*, 2017.
- [31] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017.
- [32] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv:1512.03385*, 2015.
- [33] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS*, 2014.
- [34] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets: Time-Frequency Methods and Phase Space*, pages 289–297. 1989.
- [35] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*, 2017.
- [36] L. Jin, Z. Chen, and Z. Tu. Object detection free instance segmentation with labeling transformations. *arXiv:1611.08991*, 2016.
- [37] A. Khoreva, R. Benenson, J. Hosang, M. Hein, and B. Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *CVPR*, 2017.
- [38] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. Instancecut: from edges to instances with multicut. In *CVPR*, 2017.
- [39] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*, 2011.
- [40] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
- [41] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proc. IEEE*, 1998.

- [42] E. Levinkov, J. Uhrig, S. Tang, M. Omran, E. Insafutdinov, A. Kirillov, C. Rother, T. Brox, B. Schiele, and B. Andres. Joint graph decomposition & node labeling: Problem, algorithms, applications. In *CVPR*, 2017.
- [43] K. Li, B. Hariharan, and J. Malik. Iterative instance segmentation. In *CVPR*, 2016.
- [44] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, 2017.
- [45] X. Liang, Y. Wei, X. Shen, J. Yang, L. Lin, and S. Yan. Proposal-free network for instance-level object segmentation. *arXiv preprint arXiv:1509.02636*, 2015.
- [46] G. Lin, C. Shen, A. van den Hengel, and I. Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *CVPR*, 2016.
- [47] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.
- [48] T.-Y. Lin et al. Microsoft COCO: Common objects in context. In *ECCV*, 2014.
- [49] S. Liu, J. Jia, S. Fidler, and R. Urtasun. Sgn: Sequential grouping networks for instance segmentation. In *ICCV*, 2017.
- [50] S. Liu, X. Qi, J. Shi, H. Zhang, and J. Jia. Multi-scale patch aggregation (mpa) for simultaneous detection and segmentation. In *CVPR*, 2016.
- [51] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.
- [52] Z. Liu, X. Li, P. Luo, C. C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *ICCV*, 2015.
- [53] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.
- [54] P. Luo, G. Wang, L. Lin, and X. Wang. Deep dual learning for semantic image segmentation. In *ICCV*, 2017.
- [55] A. Newell and J. Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *NIPS*, 2017.
- [56] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille. Weakly- and semi-supervised learning of a dcnn for semantic image segmentation. In *ICCV*, 2015.
- [57] G. Papandreou, I. Kokkinos, and P.-A. Savalle. Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In *CVPR*, 2015.
- [58] P. O. Pinheiro, R. Collobert, and P. Dollár. Learning to segment object candidates. In *NIPS*, 2015.
- [59] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016.
- [60] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.
- [61] M. Ren and R. S. Zemel. End-to-end instance segmentation with recurrent attention. In *CVPR*, 2017.
- [62] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.
- [63] B. Romera-Paredes and P. H. S. Torr. Recurrent instance segmentation. In *ECCV*, 2016.
- [64] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014.
- [65] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016.
- [66] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv:1612.06851*, 2016.
- [67] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017.
- [68] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017.
- [69] C. Szegedy, S. Reed, D. Erhan, D. Anguelov, and S. Ioffe. Scalable, high-quality object detection. *arXiv:1412.1441*, 2014.
- [70] J. Uhrig, M. Cordts, U. Franke, and T. Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. *arXiv:1604.05096*, 2016.
- [71] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell. Understanding convolution for semantic segmentation. *arXiv:1702.08502*, 2017.
- [72] Z. Wu, C. Shen, and A. van den Hengel. Bridging category-level and instance-level semantic image segmentation. *arXiv:1605.06885*, 2016.
- [73] Z. Wu, C. Shen, and A. van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *arXiv:1611.10080*, 2016.
- [74] S. Xie, R. Girshick, P. Dollr, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.
- [75] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár. A multipath network for object detection. In *BMVC*, 2016.
- [76] H. Zhang and X. He. Deep free-form deformation network for object-mask registration. In *ICCV*, 2017.
- [77] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In *CVPR*, 2016.
- [78] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. Monocular object instance segmentation and depth ordering with cnns. In *ICCV*, 2015.
- [79] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017.
- [80] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015.
- [81] Y. Zhu, R. Urtasun, R. Salakhutdinov, and S. Fidler. segdeepm: Exploiting segmentation and context in deep neural networks for object detection. In *CVPR*, 2015.

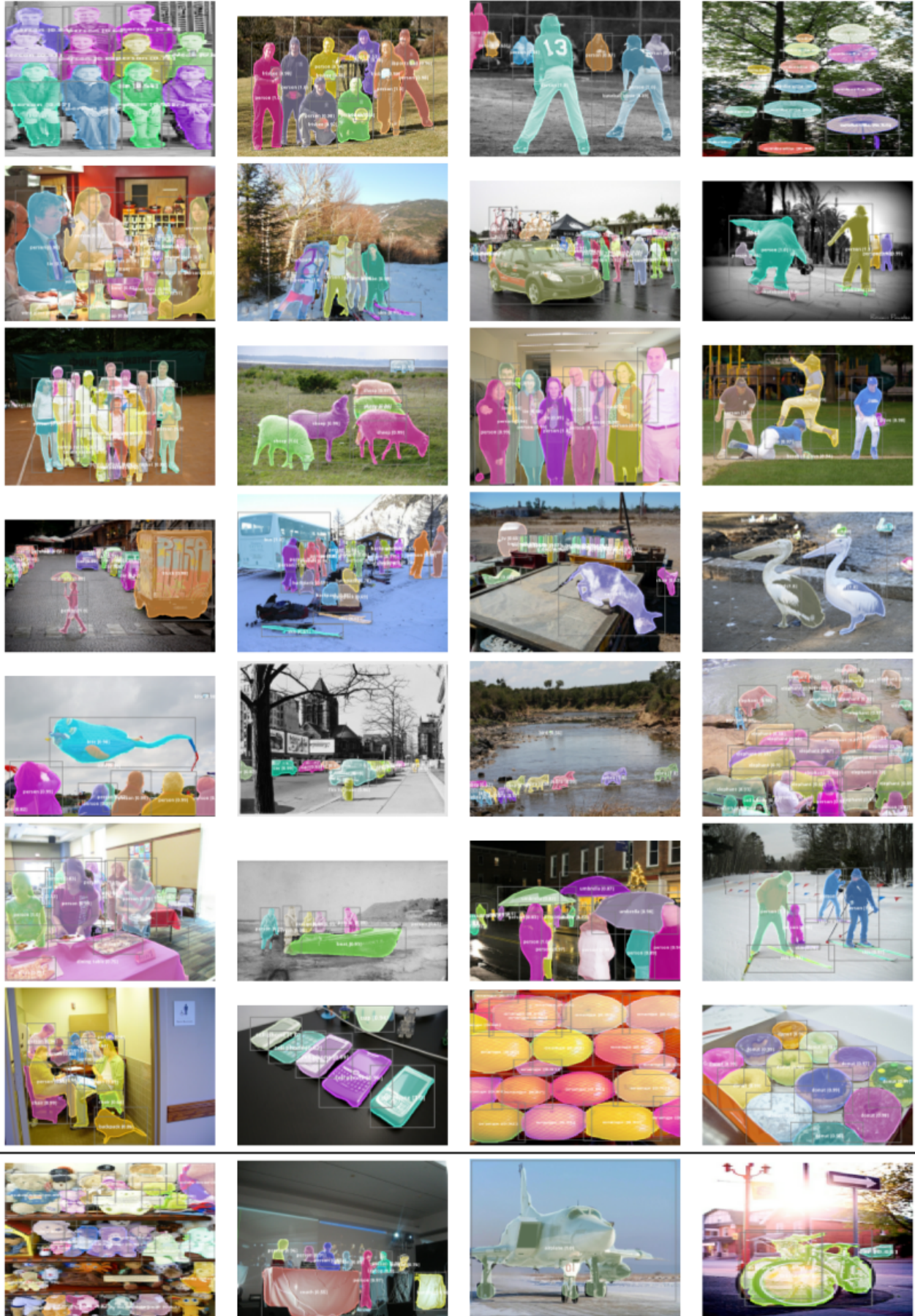


Figure 9. Visualization results on the *minival* set. As shown in the figure (particularly, last row), our failure mode comes from two parts: (1) detection failure (missed-detection and wrong classification), and (2) failure to capture sharp object boundary.