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

Technical Report: Single-Shot Panoptic Segmentation

Mark Weber Jonathon Luiten Bastian Leibe
RWTH Aachen University, Germany
mark.weber1@rwth-aachen.de {luiten, leibe}@vision.rwth-aachen.de

Abstract

In this report, we propose a novel end-to-end single-shot method that segments `things` and `stuff` into a panoptic segmentation at almost video frame rate. Current state-of-the-art approaches mostly merge instance segmentation with semantic background segmentation and are far from reaching video frame rate. In contrast, our approach relaxes the requirement to use instance segmentation by using an object detector. Still, we are able to resolve inter- and intra-class overlaps. On top of a shared encoder-decoder backbone, we utilize multiple branches for semantic segmentation, object detection, and instance center prediction. Finally, our panoptic head combines all outputs into a panoptic segmentation and can even handle conflicting predictions between branches and certain false predictions. Our network achieves 32.6% PQ on COCO at 21.8 FPS, opening up the task to a broader field of applications.

1. Introduction

In this paper, we present a unified single-shot network to tackle the task of panoptic segmentation. Unlike previous methods, we base our model design choices on the speed-accuracy trade-off. We use conceptionally simple, yet effective components for the tasks of object detection and semantic segmentation. In contrast to many related methods that heuristically merge outputs from different branches, we propose a panoptic head that is capable of fusing detected objects and segmented regions to produce a coherent panoptic segmentation. Moreover, we present a novel technique that uses instance center predictions with object detection and semantic segmentation to solve inter- and intra-class overlaps. We extensively evaluate our method on the popular COCO dataset [12] and demonstrate the effectiveness of our approach. In particular, we compare our method also to concurrent work on single-shot panoptic segmentation and achieve a far better speed-accuracy trade-off.

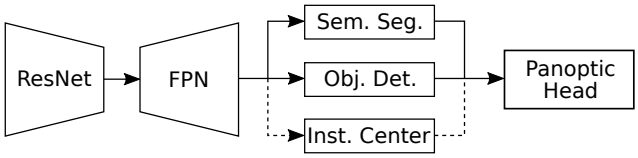

Figure 1: Our proposed network architecture leverages an encoder-decoder backbone, up to three branches and a panoptic head to produce panoptic segmentation.

2. Method

Our novel end-to-end network consists of a shared backbone that extracts multi-scale features in an encoder-decoder fashion. On top of this, we add multiple branches for semantic segmentation, object detection, and optionally instance center prediction. All these predictions are input to our panoptic head that produces panoptic segmentation without any post-processing merging steps. The overall network architecture is shown in Figure 1. Our network design goals are accuracy, speed, simplicity and comparability.

Backbone: Our backbone uses a residual network (ResNet) [4] as the encoder and a feature pyramid network (FPN) [10] as the decoder. We choose a ResNet-50 FPN that generates pyramid levels with scales from 1/128 to 1/4 resolution with each level having 256 feature dimensions.

Semantic Segmentation Branch: In all other previous work, the implementation of the semantic segmentation branch varies only slightly [6, 13, 15, 9]. Since all of them achieve similar performance, we chose [6] as our lightweight branch. However, unlike most related work, our semantic segmentation branch predicts all `things` and `stuff` classes, which is required for our panoptic head.

Object Detection Branch: For our object detection branch, we use the RetinaNet [11] architecture as it exhibits a good speed-accuracy trade-off. We keep the standard hyperparameters including nine anchor boxes.

Panoptic Head: Inspired by UPSNet [15], we propose a novel parameter-free panoptic head that works with object detections instead of instance logits. We aim to create

instance-aware logits Y than can be inferred like semantic logits. Therefore, we split our semantic logits X into X_{stuff} and X_{things}. We merely keep X_{stuff} as they do not need an instance ID. For X_{things}, we use the detection results, which include a class c and a bounding box b per object. We use c as the index to select the corresponding slice in X_{things}. In these 2D logits X_c, we use the box b to crop the logits, *i.e.*, we filter out all logit values outside b. We take these logits $X_{c,b}$ and stack them with the `stuff` logits X_{stuff}. The depth of Y varies with the number of detected objects.

Regions that are segmented as `things` by the semantic segmentation branch, but that were not detected by the detector are discarded. Moreover, falsely detected objects might still get low logit values from the semantic segmentation branch and therefore do not show up in the final panoptic segmentation. Thus, certain errors in one branch can be corrected by the other branch. The only problem that arises are overlapping bounding boxes with the same class. Since the same logit slice is selected in such cases, the overlapping regions will have the same value. Hence, an argmax operation cannot choose the correct instance. While our head addresses inter-class overlaps, intra-class overlaps cannot be resolved directly. Hence, we propose three policies to overcome this shortcoming.

Highest-Confidence Policy: This policy is similar to the way previous work handles overlapping instances (of any class) predicted by Mask R-CNN. With this policy, very likely predictions could hide false positives.

Smallest-First Policy: The PQ metric [7], used to measure the quality of panoptic segmentation, puts much emphasis on small instances since even the smallest ones count as much towards the score as the largest segments. Hence, sorting overlapping instances in increasing order of size prevents large instances from overshadowing smaller ones.

Closest-Center Policy: Both policies described above suffer from the effect that the overlap will be assigned to only one instance. Hence, they introduce straight contours/sharp corners originating from the bounding boxes. While this issue might not be reflected much in the final score since the number of intra-class bounding box overlaps might be somewhat limited, the visual quality for these cases will suffer tremendously. Therefore, we propose a third class-agnostic instance center prediction branch, which predicts the center of the most likely bounding box. We utilize the same architecture as for semantic segmentation but predict only the absolute offset from the pixel's location. The overlap is resolved by adding the predicted offset to the pixels' location and computing the L2 distance to all centers of boxes containing that pixel. The smallest distance gives the most likely instance for each pixel. This policy enables accurate contours assuming good instance center predictions.

Unknown Predictions: For some pixels, the semantic segmentation branch might predict a `thing` class, but the detector detects no object at this location. In our current setup, the panoptic head will predict the most likely `stuff` class for these pixels. However, it might be beneficial to make an `unknown` prediction for these cases during inference. Hence, we also investigate such post-processing steps.

Joint Training: Our object detection branch is trained with focal loss and a smooth L1 loss. The semantic segmentation branch uses a cross-entropy loss, and our instance center prediction branch uses a L1 loss. We train with a weighted combination of the loss terms. For the proposed *smallest-first* and *highest-confidence* policies, we train only two branches but add the instance center prediction branch for the *closest-center* policy.

3. Experiments

We demonstrate that our network is a simple and fast yet accurate method for panoptic segmentation. We show that our individual branches achieve expected scores and maintain this in a multi-task setting. We confirm that our panoptic head can combine results from object detection and semantic segmentation and thus, relax the requirement to use instance segmentation. Moreover, we show that our panoptic head can handle conflicting predictions and overlaps.

Dataset: We perform all our experiments on the challenging COCO dataset [12] with panoptic segmentation annotations. The dataset contains annotations for 80 `things` categories and 53 `stuff` categories.

Evaluation Criteria: For semantic segmentation and object detection, we use mean Intersection-over-Union (mIoU) and Average Precision (AP) , respectively. For panoptic segmentation, we use the standard PQ metric [7].

Experimental Setup: We use ImageNet pre-trained weights for our ResNet-50 encoder [4]. The FPN decoder and all branches are trained from scratch. In contrast to most related work that train with a batch size of 16, we use a batch size of 4. Hence, we freeze the BatchNorm layers [5] in the encoder and omit fine-tuning their statistics. We train with Adam, a learning rate of 1e-5, a weight decay of 1e-4 and set $\beta_1 = 0.9$ and $\beta_2 = 0.99$. We limit the training to 14 epochs and decay the learning rate by a factor of 10 after the 7th and 10th epoch. All images are scaled to at least 576px as long as the longer side has less than 864px. For training, we only apply left-right flipping as augmentation.

3.1. Individual Components

We validate our implementation of each branch by training them separately for their dedicated task.

Semantic Segmentation: We compare with the single-scale scores by [6] for the same architecture on COCO Stuff. While [6] uses a much larger ResNeXt-152, it is only trained on a custom split of the training data. Hence, the

(a) Input (b) Ground Truth (c) Prediction

Figure 2: We plot the direction & magnitude of the offsets.

COCO Panoptic						
Weights		Pan. Seg.			Seg.	Det.
Sem.	Obj.	**PQ**	PQ^{th}	PQ^{st}	mIoU	AP
1.00	-	-	-	-	**50.0**	-
-	1.00	-	-	-	-	**31.2**
0.20	0.80	28.9	34.1	21.1	48.4	**30.5**
0.25	0.75	29.4	34.6	21.7	49.0	**30.5**
0.33	0.66	**29.6**	**34.7**	21.9	49.6	30.2
0.50	**0.50**	**29.6**	34.2	22.5	50.2	29.2
0.66	0.33	29.4	33.6	**23.0**	**50.4**	27.7
0.75	0.25	28.8	32.7	22.9	50.2	26.7
0.80	0.20	28.4	32.2	22.8	50.2	25.9

Table 1: We show the influence of the weighting on COCO *val2017*.

COCO Panoptic						
Policy	**PQ**	PQ^{th}	PQ^{st}	mIoU	AP
HC	**29.6**	**34.2**	**22.5**	50.2	29.2
SF	29.3	33.8	**22.5**	50.2	29.2

Table 2: We evaluate the *highest-confidence* (HC) and *smallest-first* (SF) policy on the COCO *val2017* dataset.

scores are not directly comparable. However, our score of 26.9% comes close to the reference score of 27.8% mIoU.

Object Detection: We follow the proposed hyper-parameters by Lin *et al*. [11] for our RetinaNet implementation. We compare score using the same ResNet-50 backbone at a similar input size. We obtain slightly worse results with 33.1% AP compared to 34.3% AP reference score, which we attribute to different training configurations.

Instance Center Prediction: In our proposed network, the instance center predictions are used to resolve id assignment for intra-class overlaps. Hence, we evaluate the branch by tracking correct id assignments for overlaps with ground-truth bounding boxes. For 90.8% of the pixels belonging to multiple boxes of the same class, the branch can assign the correct ground-truth bounding box. We note that due to the 'crowd' annotations, many overlaps are not covered. Therefore, we also show qualitative examples in Figure 2.

3.2. Panoptic Segmentation

We now investigate the performance of our panoptic head, show the effect of using `unknown` predictions and examine the speed-accuracy trade-off.

Policies: To evaluate our network, we already have to select a policy to resolve intra-class overlaps. We choose the *highest-confidence* policy as it is used in previous work to resolve overlaps in Mask R-CNN. Moreover, we apply a confidence threshold of 0.4 to the outputs of our detector. Table 1 shows the accuracy of our network with the *highest-confidence* policy on the COCO *val2017* dataset. The scores provide evidence that equal weighting achieves the best balance between `things` and `stuff` for panoptic segmentation. Still, the results show that range of weights performs similarly well. We perform the same ablations for the *smallest-first* policy, but only compare the best results in Table 2. The results indicate that the overlap resolution by confidence score sorting is superior to size-based sorting.

By design, the *highest-confidence* and *smallest-first* policies lead to corner-shaped assignments of intra-class overlaps. To achieve visually plausible contours, we introduced the third branch and the *closest-center* policy. We confirm that this setup works under different loss weightings in Table 3. Since the initial loss for the instance center prediction is two order of magnitudes higher than for the rest, we use a small weight. While the score increases only slightly, the comparison in Figure 3 shows the strength of our approach. We argue that the benefits are not well reflected in COCO. The advantages play out in images having a lot of intra-class bounding box overlaps. However, those often contain crowd labels, which are excluded from the evaluation.

Unknown Predictions: We investigate the effect of using `unknown` predictions and removing small `stuff` regions. We set the threshold to 4096 pixels for `stuff` regions, as

COCO Panoptic						
Weights		Pan. Seg.			Seg.	Det.
Sem./Obj.	Inst.	**PQ**	PQ^{th}	PQ^{st}	mIoU	AP
1.0/-	-	-	-	-	**50.0**	-
-/1.0	-	-	-	-	-	**31.2**
0.5	0.001	29.7	34.5	**22.6**	**50.3**	**29.5**
0.5	0.005	29.8	**34.8**	22.3	49.9	29.3
0.5	0.01	**29.9**	**34.8**	22.3	49.7	29.4
0.5	GT	31.4	37.3	22.5	50.2	29.2

Table 3: We list the PQ scores of our final neural network under three different weightings and the *closest-center* policy on the COCO *val2017* split. Compared to the other policies, the performance increases by 0.3–0.6% PQ.

(a) Input (b) Smallest-First (c) Closest-Center

Figure 3: The *closest-center* policy achieves much more visually plausible results.

COCO Panoptic				
Stuff Rem.	Unk.	**PQ**	PQ^{th}	PQ^{st}
-	-	29.9	34.8	22.5
✓	-	32.2 *(+2.3)*	34.8	28.4 *(+5.9)*
✓	✓	32.4 *(+2.5)*	34.8	28.6 *(+6.1)*

Table 4: We experiment with `unknown` predictions (Unk.) and small `stuff` removal (Stuff Rem.) on COCO *val2017*.

COCO Panoptic				
Name	Backbone	**PQ**	PQ^{th}	PQ^{st}
PanFPN [6]	ResNet-50	39.0	45.9	28.7
OANet [13]	ResNet-50	39.0	48.3	24.9
AUNet [9]	ResNet-50	39.6	**49.1**	25.2
UPSNet [15]	ResNet-50	**42.5**	48.5	**33.4**
JSIS† [3]	ResNet-50	26.9	29.3	23.3
OCFusion† [8]	ResNet-50	41.0	49.0	29.0
AdaptIS† [14]	ResNet-50	35.9	40.3	29.3
DeeperLab† [16]	LWMNV2	24.1	-	-
DeeperLab† [16]	WMNV2	27.9	-	-
DeeperLab† [16]	Xception-71	33.8	-	-
Ours (SF)	ResNet-50	31.8	33.8	28.9
Ours (HC)	ResNet-50	32.1	34.2	28.8
Ours (CC)	ResNet-50	32.4	34.8	28.6

Table 5: We compare with previous and concurrent (†) work on COCO *val2017*. SF, HC, and CC refer to our policies.

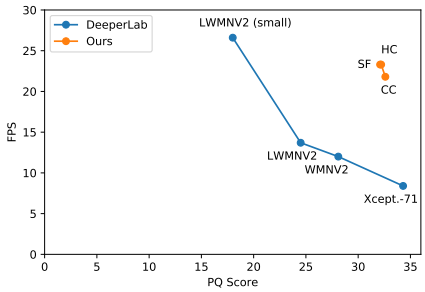

Figure 4: We compare the speed-accuracy trade-off.

is best practice [6] and list results in Table 4.

Final Results: We compare our method to two-stage methods and the concurrent single-shot approach DeeperLab [16] in Table 5. We do not use test-time tricks or backbone optimizations, *e.g.*, deformable convolutions [2], due to run-time considerations. For a fair comparison, we report scores for a ResNet-50. Our network performs better than two DeeperLab variants. DeeperLab using the Xception-71 [1] is slightly better but uses also a far deeper backbone with 42.1M parameters, while our encoder has only 23.6M.

We plot the accuracy and FPS of the single-shot methods in Figure 4. All run-times, including DeeperLab, are measured on the same V100 GPU. For DeeperLab, we use the official timings but include the timings for merging. We argue the FPS must measure the time from input to output to fully cover the run-time to obtain panoptic segmentation. Moreover, all fully convolutional networks run faster (and usually perform worse) on smaller input due to fewer computations. Hence, we compare methods with similar input sizes for a fair evaluation. All our networks run significantly faster under these conditions than the competitor DeeperLab. With 21.8 to 23.3 FPS and 32.6% PQ on COCO *test-dev*, it is evident that our network achieves a significant better trade-off making it applicable in a much broader field.

Acknowledgement: This project has been funded, in parts, by ERC Consolidator Grant DeeViSe (ERC-2017-COG-773161). Simulations were performed with computing resources granted by the RWTH Aachen University under projects rwth0431.

References

[1] Francois Chollet. Xception: Deep Learning With Depthwise Separable Convolutions. In *CVPR*, 2017.

[2] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable Convolutional Networks. In *ICCV*, 2017.

[3] Daan de Geus, Panagiotis Meletis, and Gijs Dubbelman. Panoptic Segmentation with a Joint Semantic and Instance Segmentation Network. *arXiv:1809.02110*, 2019.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

[5] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *ICML*, 2015.

[6] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic Feature Pyramid Networks. In *CVPR*, 2019.

[7] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic Segmentation. In *CVPR*, 2019.

[8] Justin Lazarow, Kwonjoon Lee, and Zhuowen Tu. Learning Instance Occlusion for Panoptic Segmentation. *arXiv:1906.05896*, 2019.

[9] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-Guided Unified Network for Panoptic Segmentation. In *CVPR*, 2019.

[10] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017.

[11] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *ICCV*, 2017.

[12] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

[13] Huanyu Liu, Chao Peng, Changqian Yu, Jingbo Wang, Xu Liu, Gang Yu, and Wei Jiang. An End-To-End Network for Panoptic Segmentation. In *CVPR*, 2019.

[14] Konstantin Sofiiuk, Olga Barinova, and Anton Konushin. AdaptIS: Adaptive Instance Selection Network. *arXiv:1909.07829*, 2019.

[15] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. UPSNet: A Unified Panoptic Segmentation Network. In *CVPR*, 2019.

[16] Tien-Ju Yang, Maxwell D. Collins, Yukun Zhu, Jyh-Jing Hwang, Ting Liu, Xiao Zhang, Vivienne Sze, George Papandreou, and Liang-Chieh Chen. DeeperLab: Single-Shot Image Parser. *arXiv:1902.05093*, 2019.