

Single-Shot Panoptic Segmentation

Mark Weber¹, Jonathon Luiten¹ and Bastian Leibe¹

Abstract— We present a novel end-to-end single-shot method that segments countable object instances (*things*) as well as background regions (*stuff*) into a non-overlapping panoptic segmentation at almost video frame rate. Current state-of-the-art methods are far from reaching video frame rate and mostly rely on merging instance segmentation with semantic background segmentation, making them impractical to use in many applications such as robotics. Our approach relaxes this requirement by using an object detector but is still able to resolve *inter-* and *intra-class* overlaps to achieve a non-overlapping segmentation. 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 as well as certain false predictions. Our network achieves 32.6% PQ on MS-COCO at 23.5 FPS, opening up panoptic segmentation to a broader field of applications.

I. INTRODUCTION

Scene understanding is one of the fundamental yet challenging tasks of computer vision. Traditional scene understanding tasks include object detection, instance segmentation, and semantic segmentation. The former tasks address classification and localization of countable foreground objects, while the latter focuses on segmenting image regions belonging to the same class without distinguishing different instances. Both tasks are necessary for a complete scene understanding. However, in recent years, the approaches used for these tasks have been quite different. For example, region-proposal networks [1] are used for object detection and instance segmentation, while fully convolutional networks [2] have been used for semantic segmentation.

Panoptic segmentation (Fig. 1) was introduced by Kirillov *et al.* to tackle the problem of diverging modeling strategies for different tasks in the field of visual scene understanding [3]. Current methods for this task are mostly two-stage approaches which extend the proposal-based Mask R-CNN [1]. Similarly to object detection and instance segmentation, these approaches achieve high accuracy but suffer from slow run-times that make them infeasible for many real-world applications. In contrast, single-shot methods have the potential to be faster and simpler, enabling many real-world applications with substantial efficiency requirements such as robotics and autonomous driving. For many such applications, the desire to navigate requires some scene understanding. The better this scene understanding is, the better the planning and navigation task becomes. So far, single-shot approaches for object detection like YOLO [4], SSD [5],

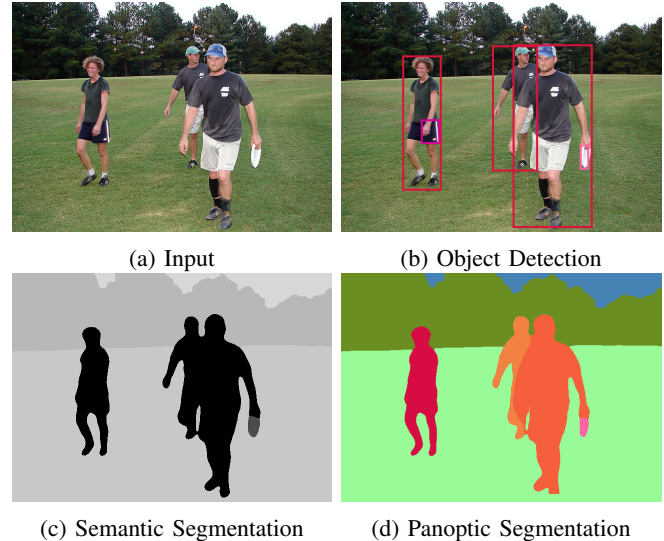


Fig. 1: Our single-shot panoptic segmentation network generates object detection and semantic segmentation results that our panoptic head combines to panoptic segmentation. Remarkably, our panoptic head can resolve inter- and intra-class overlaps and even handle wrong detections, *e.g.*, the detected handbag (top-right image).

and their successors have already made substantial impact in robotics. Hence, our goal is to open up panoptic segmentation to the robotics field and we argue it is important to not just push for algorithms with the absolute best performance, but also to explore the trade-off between speed and accuracy.

In this paper, we present a unified single-shot network to tackle the task of panoptic segmentation. Unlike previous methods that maximize only for accuracy, we base our model design choices on the speed-accuracy trade-off, which is of special interest for the robotics community. We use conceptually simple, yet efficient components for the tasks of object detection and semantic segmentation. In contrast to many related methods that heuristically merge outputs from different branches, we investigate more conflict-resolving merging procedures. We present a novel panoptic head that uses instance center predictions with object detection and semantic segmentation to extend on naive merging. This novel approach can resolve both intra- and inter-class overlaps to obtain instance segmentation and is able correct false predictions of individual components. We extensively evaluate our method on the popular COCO dataset [6] and demonstrate the effectiveness of our approach. In particular, we achieve huge run-time gains compared to state-of-the-art techniques. Furthermore, we compare our method also to one concurrent work on single-shot panoptic segmentation

¹RWTH Aachen University
mark.weber1@rwth-aachen.de

and achieve a far better speed-accuracy trade-off.

II. RELATED WORK

Semantic Segmentation: State-of-the-art methods for semantic segmentation utilize a fully convolutional neural network (FCN) [7] and follow an encoder-decoder scheme. Several approaches incorporate contextual information by aggregating multi-scale features. The use of dilated convolutions [8], [9], [10], [11] has allowed methods to importantly exploit a larger image context without increasing the number of model parameters. Recently, methods using variants of pyramid pooling became dominant. PSPNet [2] uses spatial pyramid pooling, while DeepLab proposes an atrous spatial pyramid pooling module (ASPP) [11]. With this multi-scale contextual information, these methods demonstrate high accuracy on standard benchmarks. A fully convolutional component of our network is dedicated to predicting semantic segmentation by leveraging multi-scale features from a pyramid but refrains from using dilated convolutions since they are computationally more expensive.

Object Detection and Instance Segmentation: In contrast to semantic segmentation, methods for object detection and instance segmentation identify different instances of the same class. Two-stage methods using region proposals have become the dominant concept for high accuracy object detection and instance segmentation, *e.g.*, Mask R-CNN [1]. In these approaches a set of proposals is generated and then processed by a sub-network for classification and segmentation. While these methods achieve state-of-the-art accuracy, they are infeasible to use for many applications due to their computational costs. For object detection, single-shot methods address the speed-accuracy trade-off and achieve remarkable results. Besides YOLO [4] and SSD [5], RetinaNet [12] uses a dense field of anchor boxes and CornerNet [13] generates heatmaps to produce bounding boxes.

While single-shot detectors are closing the accuracy gap to two-stage detectors, single-shot instance segmentation methods lag behind state-of-the-art methods. Several methods learn embeddings which are clustered to produce instances [14], [15], [16]. While this concept is straightforward, current techniques still achieve low accuracy and are not necessarily faster than two-stage methods. A different way of producing instances is to predict instance centers per pixel and to combine these with predicted bounding boxes to instances [17]. However, this technique is far from achieving state-of-the-art accuracy. Since single-shot instance segmentation methods still gain much lower accuracy than their two-stage competitors, we choose to use a single-shot object detection network (RetinaNet [12]) instead that predicts object bounding boxes.

Panoptic Segmentation: Panoptic segmentation can be understood as a multi-task learning problem. Prior work on multi-task learning focuses on learning multiple tasks with a shared backbone [18], [19]. However, conflicting predictions of multiple branches have typically not been resolved. Panoptic segmentation, as defined by [3], expects

unique predictions for each pixel, which requires assigning one semantic class per pixel and an instance ID for all pixels belonging to a `thing` class. Since the introduction of panoptic segmentation, all published supervised methods have focused on a two-stage architecture [20], [21], [22], [23], [24]. The same holds for all high scoring entries in the COCO panoptic challenge [25].

The typical network design extends Mask R-CNN by adding a parallel semantic segmentation branch and a fusion step to combine instance and semantic segmentation to produce panoptic segmentation [21]. This fusion step needs to solve two major problems. First, the predicted instances by Mask R-CNN can overlap; and second, the output of both branches can be conflicting, *e.g.*, a pixel may be predicted to belong to a car instance by one branch and to the sky by the other branch. [3], [21], [22] all tackle the first problem by sorting the instances based on the confidence score above a certain threshold and defining an IoU threshold to skip instances that overlap with other instances too much. [23] propose a module that learns the spatial relationship between different semantic *classes* or *instances*, respectively. On top of the architecture [21], [22] adds attention layers to reduce conflicts between branches. All these approaches address the second problem merely by always taking the Mask R-CNN output and they only use the semantic segmentation for the remaining unassigned pixels. The only more sophisticated method is UPSNet [20] that leverages a panoptic head to combine semantic segmentation and instance segmentation predictions into instance-aware logits.

Since all of these approaches are based on Mask R-CNN [1], they all suffer from high computational cost and hence, cannot be used for applications that require real-time operation. Please note, that we compare our approach only to published peer-reviewed methods. To the best of our knowledge, there is no peer-reviewed published single-shot method for panoptic segmentation. Therefore, we follow the practice of [24] and also compare to concurrent work DeeperLab [26]. DeeperLab is a single-shot architecture that is based on DeepLab [11] and PersonLab [27]. Class-agnostic instances are predicted with the help of keypoint heatmaps. Their semantic class is determined with a majority vote from the semantic prediction. Predicted `stuff` classes are merely kept for remaining pixels.

In this work, we propose a single-shot architecture and thus, do not extend Mask R-CNN. Unlike most state-of-the-art approaches, we utilize a conflict-resolving merging head in our architecture that tackles the problem of conflicting predictions. In contrast to UPSNet [20], our merging head is capable of working with bounding boxes instead of instance segments, which opens our framework to a broad range of available detectors. Unlike DeeperLab [26], we show that we are capable of achieving close to video frame rate performance for our end-to-end network.

III. METHOD

Our proposed end-to-end network consists of a shared convolutional backbone that extracts multi-scale features in

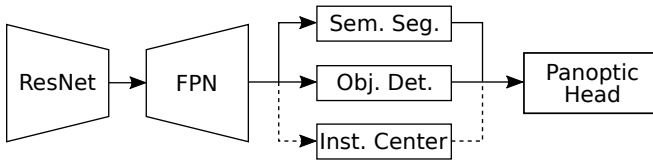


Fig. 2: Our proposed network architecture leverages an encoder-decoder backbone, up to three branches and a panoptic head to produce panoptic segmentation.

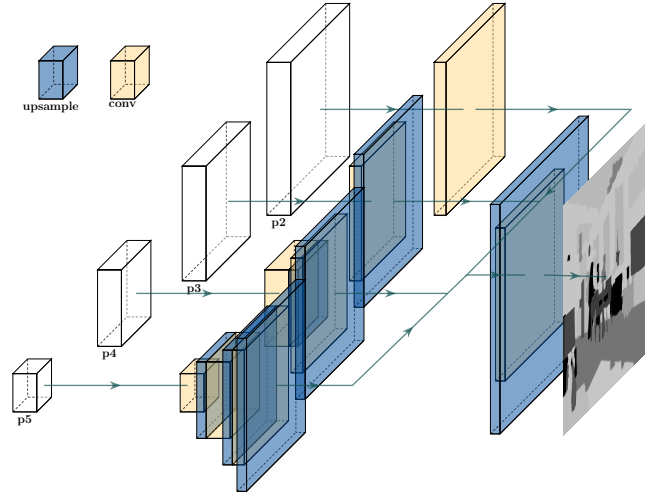


Fig. 3: The semantic segmentation branch leverages the levels p2 through p5 to produce logits. Each feature pyramid tensor is convolved (yellow) and upsampled (blue) to the same spatial resolution and finally element-wise summed up before the logits are computed and upsampled to the original resolution.

an encoder-decoder fashion. On top of this, we add multiple branches for the tasks of semantic segmentation, object detection, and optionally instance center prediction. All these predictions are input to our final panoptic head that produces panoptic segmentation without any post-processing merging steps involved. The overall network architecture is shown in Figure 2. Our network design goals are accuracy, speed, simplicity and comparability.

A. Proposed Architecture

Backbone: Our backbone uses a residual network (ResNet) [28] as the encoder and a feature pyramid network (FPN) [29] as the decoder. Hence, it offers multi-scale contextual information that can be used for both semantic segmentation as well as for object detection. We choose a standard ResNet-50 FPN to make comparisons to previous methods as fair as possible. Our shared backbone generates pyramid levels with scales from 1/128 to 1/4 resolution with each level having 256 feature dimensions.

Semantic Segmentation Branch: Similar to [21], we use a lightweight fully convolutional branch to predict semantic segmentation. We take the feature pyramid levels p5 to p2, which correspond to 1/32 to 1/4 resolution, and alternate convolutions and upsampling operations until all pyramid levels are on a quarter resolution. After convolutions, we use

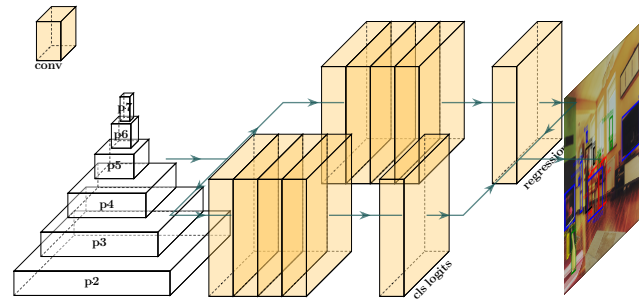


Fig. 4: The object detection branch utilizes all levels of the feature pyramid to predict bounding boxes and corresponding classes. Each pyramid level is fed through 4 convolutions (yellow) for each task. Finally, offsets for all anchor boxes and corresponding logits are predicted.

GroupNorm layers [30] for normalization and ReLU activation functions. Finally, we aggregate all upsampled pyramid levels by accumulating them and using an additional convolution to produce semantic logits. The network is sketched in Figure 3. In most previous work, the implementation of the semantic segmentation branch varies only slightly [23], [20], [22], [24]. Since all of them achieve similar performance, we chose this lightweight branch for our purpose. 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: While our method is not limited to one specific object detector, we use the RetinaNet [12] architecture for demonstration purpose as it exhibits a good speed-accuracy trade-off. RetinaNet contains two fully convolutional subnetworks that classify and regress anchor boxes. The same network weights are used throughout all pyramid levels from 1/128 to 1/4 to detect `things` of all sizes. Figure 4 shows the object detection branch. We keep the standard hyperparameters to demonstrate the out-of-the-box performance of our overall network.

Panoptic Head: Taking inspiration from UPSNet [20], we propose a novel parameter-free panoptic head that is capable of working with object detections instead of instance logits. Our goal is to create instance-aware logits Y than can be inferred like semantic logits but offer additional instance information. Therefore, we split our semantic logits X into `stuff` X_{stuff} and `things` X_{things} logits. We merely copy X_{stuff} into Y as they do not need an instance ID. For X_{things} , we use the object 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 bounding 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} . After this selection, $X_{c,b}$ will still have the same width and height as the semantic logits. However, the number of channels in Y directly depends on the number of detected objects and therefore, varies for each input.

In order to illustrate this point, consider the following

example. Suppose three cars have been detected. Our network then selects the car logit slice $X_{c=\text{car}}$ three times, crops the corresponding bounding boxes b_1, b_2, b_3 and stacks them with X_{stuff} to gain instance-aware logits Y . Hence, we end up with three new logit slices, each corresponding to a car instance. Regions that are segmented as `things` by the semantic segmentation branch, but that were not detected by the object 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, false predictions in one branch can be corrected by the other branch, which we consider a significant improvement over simple merging by copying.

Since the original logits remain, an argmax operation is still capable of achieving precise contours of objects, even if the detector does not provide such information. The only problem that arises comes from overlapping bounding boxes with the same class, while overlaps of different classes are not an issue. Since the same logit slice is selected in such cases, the overlapping regions will have the same value in our final result. Hence, an argmax operation cannot choose the correct instance. To overcome this problem, we propose and investigate different strategies.

B. Overlap Resolution

All previously published work requires instance segmentation results to produce panoptic segmentation. Part of our work is the relaxation of this requirement to object detectors. While the panoptic head addresses *inter-class* overlaps, *intra-class* overlaps cannot be resolved directly. Hence, we propose three different policies to overcome this shortcoming. Combined with the panoptic head, we can solve both kinds of overlaps, thus making the use of instance segmentation subnetworks optional.

Highest-Confidence Policy: This policy is similar to the way previous work handles overlapping instances (of any class) predicted by Mask R-CNN. Assuming the confidence score correlates strongly to the probability of the existence of an object, sorting the overlapping bounding boxes in decreasing order of confidence leads to a higher score. With this policy, very likely predictions could hide false positives.

Smallest-First Policy: The PQ metric [3], 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 with 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 of the result for these cases will suffer tremendously. Therefore, we propose a third branch in our framework. The task of this third branch is

class-agnostic instance center prediction, which we specify as the prediction of the center of the bounding box.

Uhrig et al. [17] used instance center predictions to cluster instances directly with limited success. However, clustering predictions to achieve instance segmentation has to solve *inter-* and *intra-class* overlaps, while our method only needs to resolve *intra-class* overlaps. *Intra-class* overlaps, intuitively, formulate only a small amount of all potential overlaps. We utilize the same lightweight architecture as for semantic segmentation but predict only the absolute offset in x- and y-direction 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 the 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.

C. Inference and Training

Unknown Predictions: While the task of panoptic segmentation is defined as assigning a semantic class and for `things` also an instance ID to every pixel, previous work makes use of an `unknown` prediction class in some cases. For example, [21], [20] remove `stuff` regions that occupy fewer pixels than a certain threshold. Moreover, they make `unknown` predictions for certain `stuff` areas that their branches produce conflicting assignments. While this practice is debatable as it weakens the task requirements, it is encouraged by improved scores in the PQ metric. For some pixels, the semantic segmentation branch might predict a `thing` class, but the object 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 the effect of such post-processing steps for our method.

Joint Training: During training, we combine four different loss functions. Our object detection branch is trained with focal loss L_{FL} for classification and a smooth L1 loss L_{reg} for box regression. The semantic segmentation branch uses a cross-entropy loss L_s per pixel, and our instance center prediction branch uses a L1 loss L_i per pixel. Our network is trained with a weighted combination of the loss terms: $L = \lambda_o(L_{\text{FL}} + L_{\text{reg}}) + \lambda_s L_s + \lambda_i L_i$. For the proposed *smallest-first* and *highest-confidence* policies, we train our network without the instance center prediction, but we add this branch for the *closest-center* policy. We investigate different weighting schemes experimentally. We note that methods choosing weights automatically are not feasible for us as they are either too expensive, e.g., require multiple backward passes [31], or are known to have optimization problems when using the Adam optimizer [18].

IV. EXPERIMENTS

Our goal is to demonstrate that our network can serve as a simple and fast yet accurate method for panoptic segmentation in applications with strong run-time constraints. We show that our individual branches achieve expected

accuracy and maintain this in a multi-task setting. We confirm that our panoptic head is capable of combining results from object detection and semantic segmentation to panoptic segmentation and thus, relax the requirement to use instance segmentation input. Moreover, we show that our panoptic head can handle conflicting predictions and overlaps. We exhaustively evaluate our proposed policies, different weightings, and post-processing steps.

Dataset: We perform all our experiments on the COCO dataset [6]. The dataset contains 118K training, 5K validation, and 20K test images with panoptic segmentation annotations for 80 *things* and 53 *stuff* categories. The large number of classes makes this dataset much more challenging under run-time constraints than datasets with only a dozen categories.

Evaluation Criteria: For single-task performance, we use mean Intersection-over-Union (mIoU) [32] and Average Precision (AP) [6] for semantic segmentation and object detection, respectively. For panoptic segmentation, we use the standard PQ metric [3]. The PQ^{th} and PQ^{st} scores refer to the PQ scores for the *things* and *stuff* subset of the data, respectively. The metric is defined as follows:

$$PQ = \frac{\sum_{(p,q) \in \text{TP}} \text{IoU}(p,q)}{|\text{TP}| + \frac{1}{2}|\text{FP}| + \frac{1}{2}|\text{FN}|}$$

where TP, FP and FN correspond to the set of true positives, false positives and false negatives. Predicted segments that have an IoU of greater than 0.5 with a ground-truth segment belong to the TP set.

Experimental Setup: We use ImageNet pre-trained weights for our ResNet-50 encoder [28]. The FPN decoder and all branches are trained from scratch. We implement our model with PyTorch [33] and train it on two Tesla V100 GPUs. 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 [34] 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 such that the shorter side is at least 576px as long as the longer side has less than 864px. For training, we only apply left-right flipping as augmentation.

A. Individual Components

We validate our implementation of each branch by training them separately for their dedicated task. For brevity, we focus on object detection and instance center prediction. Our semantic segmentation branch comes with 26.9% mIoU close to the reference score by [21] of 27.8% mIoU on the COCO Stuff dataset with 92 classes and, thus, confirms that our network is indeed able to perform semantic segmentation. We validate our object detection branch by comparing to the official RetinaNet [12] scores on the COCO Detection dataset with 80 classes. The instance center prediction is evaluated on the COCO Panoptic dataset by recording the

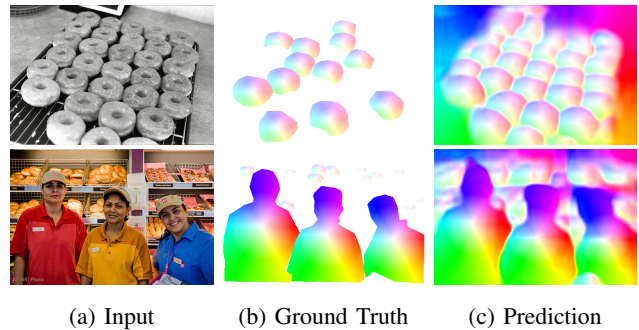


Fig. 5: The results provide evidence that the instance center prediction branch can achieve smooth contours of overlapping instances of the same task. The visualization reflects the direction and magnitude of the offset vectors.

number of cases that *intra-class* overlaps of ground-truth bounding boxes are resolved correctly.

Object Detection: We follow the proposed hyper-parameters by Lin *et al.* [12] for our RetinaNet implementation. The reference implementation was trained with a batch size of 16 and SGD. 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. Hence, it can indeed be used for this task. We note that due to the 'crowd' annotations in COCO, many *intra-class* overlaps are not considered for computing the final score, which limits our ability to showcase the strength of this branch. Therefore, we also show qualitative examples in Figure 5.

B. Panoptic Segmentation

Having verified that our components work individually, we now investigate the performance of our panoptic head. Furthermore, we show the effect of the commonly applied `unknown` predictions. Finally, we examine the speed-accuracy trade-off and compare it to related work.

Policies: For evaluating our network, we already have to select a policy to resolve *intra-class* overlaps in our panoptic head. We choose the *highest-confidence* policy as this strategy is used in previous work to resolve overlaps in Mask R-CNN. Additionally, we apply a confidence threshold of 0.4 to the outputs of our detector. One challenge in multi-task training is setting the loss weights $\lambda_o, \lambda_s, \lambda_i$ such that the overall accuracy is maximized. Table I shows the accuracy of our proposed network with the *highest-confidence* policy on the COCO *val2017* dataset. We note that the single-task models are trained on the same dataset with the same training configuration, while previously, we reported scores on different datasets. The scores provide

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 I: We show the influence of the weighting on the single-task’s and PQ scores on the COCO *val2017* dataset. Overall, an equal weighting achieves the best balance for panoptic segmentation with the *highest-confidence* policy.

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 II: We evaluate the *highest-confidence* (HC) and *smallest-first* (SF) policy on the COCO *val2017* dataset. The evidence supports the concept of resolving intra-class overlaps based on the confidence and not on the size.

evidence that equal weighting achieves the best balance between `things` and `stuff` for panoptic segmentation. Still, the results suggest that there is a range of weights that perform similarly well. Notably, the single-task performance for semantic segmentation is resilient to weight changes and even slightly benefits from the multi-task training. However, the detection task suffers from multi-task training, and its score highly depends on the weighting.

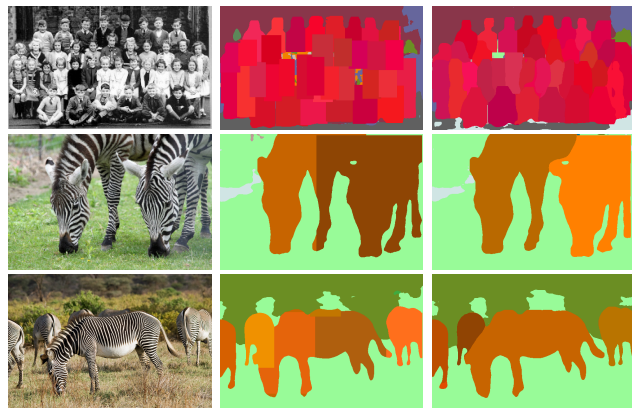
We can use the same trained networks to test the *smallest-first* policy, as it only takes effect during inference. Similar to the *highest-confidence* policy, an equal weighting performs best. Table II shows a comparison of both policies. The results indicate that the often applied overlap resolution by confidence score sorting is superior to size-based sorting.

By the design of the *highest-confidence* and *smallest-first* policy, they lead to corner-shaped assignments of *intra-class* overlaps. To overcome this, we introduced the third branch in combination with the *closest-center* policy to achieve visually plausible contours. We confirm that this setup works to achieve panoptic segmentation and investigate different loss weightings in Table III. Since the initial loss for the instance center prediction is almost two order of magnitudes higher than for the other branches, we use a small weighting for the third branch. While the score increases only slightly, the qualitative comparison in Figure 6 shows the strength of our approach. We argue that the benefits of this approach are not well reflected in the COCO dataset. The advantages of the *closest-center* policy play out in images that have a lot of overlapping instances of the same class. However, those regions are often not densely annotated but are labeled as crowd regions and 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 III: 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, performance increases by 0.3–0.6% PQ. We also show that better scores of the individual tasks do not correspond directly to better PQ scores and give an upper bound that can be achieved with ground-truth instance centers.



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

Fig. 6: We compare the panoptic segmentation produced by our *smallest-first* and *closest-center* policy. The latter achieves much more visually plausible results.

is best practice [21]. Table IV lists the scores of our *closest-center* policy with these post-processing steps. Remarkably, the removal of small `stuff` regions leads to an increase of 2.3% PQ. Since several related methods use the same removal step without any ablations, we hypothesize that the used semantic segmentation branches suffer from the same issue, which is over-segmentation of `stuff` regions.

Final Results: We use unknown predictions to report our final scores as is common practice. Table V provides results from published two-stage methods and the concurrent single-shot approach DeeperLab [26]. In contrast to some other approaches, our scores are not obtained with test-time tricks (multi-scale, l-r flipping) or other backbone optimizations, *e.g.*, deformable convolutions [35], that are known to improve scores [20], [36], due to run-time considerations. Moreover, all approaches achieve consistently better scores with deeper and wider backbones. For a fair comparison of our method, we report scores for a ResNet-50 encoder. Notably, our network performs more accurately than two DeeperLab variants. The configuration using the Xception-71 backbone [37] performs slightly better but uses also a far deeper backbone. The Xception-71 encoder has 42.1M parameters, while our encoder has only 23.6M parameters.

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 IV: We experiment with unknown predictions (Unk.) and small stuff removal (Stuff Rem.). We evaluate our network with the *closest-center* policy on COCO *val2017*. Using a threshold for stuff regions and adding unknown predictions increases the score by 2.5% PQ.

COCO Panoptic				
Name	Backbone	PQ	PQ th	PQ st
PanFPN [21]	ResNet-50	39.0	45.9	28.7
OANet [23]	ResNet-50	39.0	48.3	24.9
AUNet [22]	ResNet-50	39.6	49.1	25.2
UPSNet [20]	ResNet-50	42.5	48.5	33.4
AdaptIS [24]	ResNet-50	35.9	40.3	29.3
DeeperLab [†] [26]	LWMNV2	24.1	-	-
DeeperLab [†] [26]	WMNV2	27.9	-	-
DeeperLab [†] [26]	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 V: We report the performance on COCO *val2017* of related work using two-stage networks, the concurrent ([†]) work on single-shot methods, and our approach. We limit this comparison to the ResNet-50 for two-stage methods. SF, HC, and CC refer to our policies.

The large Xception-71 is also the reason the frame rate of the most accurate DeeperLab is still far from video frame rate. We report the accuracy and FPS of the single-shot methods in Table VI. All run-times from our method and DeeperLab are measured on a Tesla-V100-SXM2 GPU. We calculate a robust FPS score by taking the median FPS of 6 runs over the whole COCO validation set. For DeeperLab, we use the official time measurements but include the timings from intermediate results to the final panoptic segmentation. In the case of DeeperLab, the intermediate results are heatmaps and semantic segmentation. However, 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 as fewer computations are necessary. Hence, we compare methods with similar input conditions for a fair evaluation. All our networks run significantly faster under these conditions than the competitor DeeperLab. With 23.5 to 23.9 FPS, we almost achieve video frame rate. Moreover, our method is 4 times faster than state-of-the-art two-stage approach UPSNet [20]. Using the official implementation with a ResNet-50, UPSNet achieves 5.8 FPS. Figure 7 shows that our network achieves a significantly better speed-accuracy trade-off, making our approach more broadly applicable. We show qualitative results in Figure 8.

V. CONCLUSION

In this paper, we present a novel end-to-end single-shot panoptic segmentation approach that achieves remarkable accuracy, while setting a new state-of-the-art run-time performance at almost video frame rate. Our panoptic head

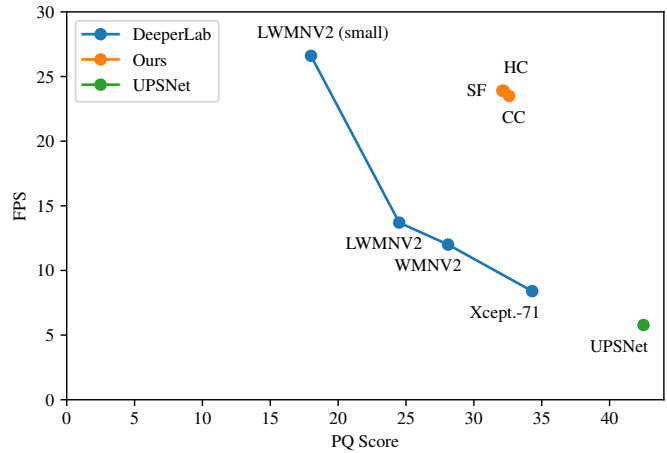


Fig. 7: We visualize the trade-off between accuracy and speed. DeeperLab trades a slightly better score for a low run-time that is far from video-rate. The plot shows that our network achieves a better trade-off and therefore, is beneficial for real-world applications.

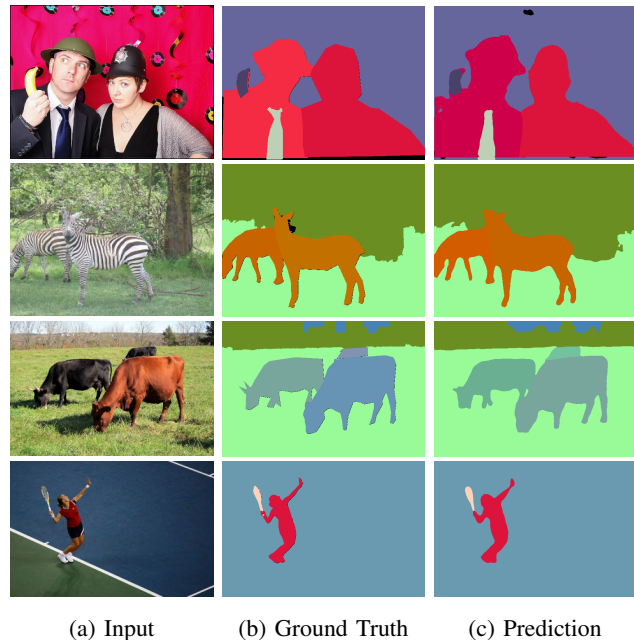


Fig. 8: Qualitative results from our final network with the *closest-center* policy on the COCO *val2017* dataset.

does not simply merge outputs of sub-networks but produces instance-aware panoptic logits. In contrast to previous work, our proposed method does not rely on instance segmentation, but can be used with detectors widely used in the robotics community. Still, our panoptic head can resolve *inter-* and *intra-class* overlaps by combining semantic segmentation, object detection and instance center prediction. This panoptic head is also able to correct errors in each of its inputs before creating the final panoptic segmentation result, which is superior to merge-by-copy methods.

Acknowledgements: 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

COCO Panoptic Test						
Name	Backbone	Input size	PQ	PQ th	PQ st	FPS
Ours (SF)	ResNet-50	576 × [576, 864]	32.1	34.3	28.8	23.9
Ours (HC)	ResNet-50	576 × [576, 864]	32.2	34.5	28.8	23.9
Ours (CC)	ResNet-50	576 × [576, 864]	32.6	35.0	29.0	23.5
DeeperLab [26]	LWMNV2	321 × 321	18.0	18.5	17.2	26.6
DeeperLab [26]	LWMNV2	641 × 641	24.5	26.9	20.9	13.7
DeeperLab [26]	WMNV2	641 × 641	28.1	30.8	24.1	12.0
DeeperLab [26]	Xcept.-71	641 × 641	34.3	37.5	29.6	8.4

TABLE VI: We compare our method to DeeperLab on COCO *test-dev*. The FPS incorporates the time from input to output. Under similar input conditions, we outperform DeeperLab in all cases regarding the run-time and in two out of three cases regarding the score. SF, HC, and CC refer to our *smallest-first*, *highest-confidence* and *closest-center* policy.

project rwth0431. We would like to thank Tobias Fischer for helpful discussions.

REFERENCES

- [1] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [2] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid Scene Parsing Network,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [3] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, “Panoptic Segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “SSD: Single Shot MultiBox Detector,” in *European Conference on Computer Vision (ECCV)*, September 2016.
- [6] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, “Microsoft COCO: Common Objects in Context,” in *European Conference on Computer Vision (ECCV)*, Sept. 2014.
- [7] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [8] 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 *International Conference on Learning Representations (ICLR)*, 2015.
- [9] —, “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs,” *arXiv:1606.00915 [cs]*, June 2016.
- [10] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking Atrous Convolution for Semantic Image Segmentation,” *arXiv:1706.05587 [cs]*, June 2017.
- [11] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation,” in *European Conference on Computer Vision (ECCV)*, Aug. 2018.
- [12] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal Loss for Dense Object Detection,” in *IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017.
- [13] H. Law and J. Deng, “CornerNet: Detecting Objects as Paired Key-points,” in *European Conference on Computer Vision (ECCV)*, 2018.
- [14] S. Kong and C. Fowlkes, “Recurrent Pixel Embedding for Instance Grouping,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [15] B. De Brabandere, D. Neven, and L. Van Gool, “Semantic Instance Segmentation with a Discriminative Loss Function,” *arXiv:1708.02551 [cs]*, Aug. 2017.
- [16] 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 [cs]*, Mar. 2017.
- [17] J. Uhrig, E. Rehder, B. Frohlich, U. Franke, and T. Brox, “Box2pix: Single-Shot Instance Segmentation by Assigning Pixels to Object Boxes,” in *IEEE Intelligent Vehicles Symposium (IV)*, June 2018.
- [18] A. Kendall, Y. Gal, and R. Cipolla, “Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [19] K. Pfeiffer, A. Hermans, I. Sáránci, M. Weber, and B. Leibe, “Visual person understanding through multi-task and multi-dataset learning,” in *German Conference on Pattern Recognition (GCPR)*, 2019.
- [20] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun, “UPSNNet: A Unified Panoptic Segmentation Network,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [21] A. Kirillov, R. Girshick, K. He, and P. Dollár, “Panoptic Feature Pyramid Networks,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [22] Y. Li, X. Chen, Z. Zhu, L. Xie, G. Huang, D. Du, and X. Wang, “Attention-Guided Unified Network for Panoptic Segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [23] H. Liu, C. Peng, C. Yu, J. Wang, X. Liu, G. Yu, and W. Jiang, “An End-To-End Network for Panoptic Segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [24] K. Sofiuk, O. Barinova, and A. Konushin, “AdaptIS: Adaptive Instance Selection Network,” in *IEEE International Conference on Computer Vision (ICCV)*, 2019.
- [25] “Joint Recognition Challenge Workshop at ECCV 2018,” <http://cocodataset.org/workshop/coco-mapillary-eccv-2018.html>.
- [26] T.-J. Yang, M. D. Collins, Y. Zhu, J.-J. Hwang, T. Liu, X. Zhang, V. Sze, G. Papandreou, and L.-C. Chen, “DeeperLab: Single-Shot Image Parser,” *arXiv:1902.05093 [cs]*, Feb. 2019.
- [27] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. Murphy, “Personlab: Person Pose Estimation and Instance Segmentation with a Bottom-Up, Part-Based, Geometric Embedding Model,” in *European Conference on Computer Vision (ECCV)*, 2018.
- [28] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [29] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature Pyramid Networks for Object Detection,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [30] Y. Wu and K. He, “Group Normalization,” in *European Conference on Computer Vision (ECCV)*, Aug. 2018.
- [31] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, “Grad-Norm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks,” in *Proceedings of the 35th International Conference on Machine Learning*, Jul 2018.
- [32] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International Journal of Computer Vision*, vol. 111, no. 1, Jan 2015.
- [33] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in PyTorch,” in *NIPS Autodiff Workshop*, 2017.
- [34] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, 2015.
- [35] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, “Deformable Convolutional Networks,” in *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [36] J. Lazarow, K. Lee, and Z. Tu, “Learning Instance Occlusion for Panoptic Segmentation,” *arXiv:1906.05896 [cs]*, Aug. 2019.
- [37] F. Chollet, “Xception: Deep Learning With Depthwise Separable Convolutions,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.