UPSNet: A Unified Panoptic Segmentation Network

Yuwen Xiong[1,2*]   Renjie Liao[1,2*]   Hengshuang Zhao[3*†]
Rui Hu[1]   Min Bai[1,2]   Ersin Yumer[1]   Raquel Urtasun[1,2]
[1]Uber ATG   [2]University of Toronto   [3]The Chinese University of Hong Kong
{yuwen, rjliao, rui.hu, mbai3, yumer, urtasun}@uber.com
hszhao@cse.cuhk.edu.hk

Abstract

In this paper, we propose a unified panoptic segmentation network (UPSNet) for tackling the newly proposed panoptic segmentation task. On top of a single backbone residual network, we first design a deformable convolution based semantic segmentation head and a Mask R-CNN style instance segmentation head which solve these two subtasks simultaneously. More importantly, we introduce a parameter-free panoptic head which solves the panoptic segmentation via pixel-wise classification. It first leverages the logits from the previous two heads and then innovatively expands the representation for enabling prediction of an extra unknown class which helps better resolve the conflicts between semantic and instance segmentation. Additionally, it handles the challenge caused by the varying number of instances and permits back propagation to the bottom modules in an end-to-end manner. Extensive experimental results on Cityscapes, COCO and our internal dataset demonstrate that our UPSNet achieves state-of-the-art performance with much faster inference. Code has been made available at: https://github.com/uber-research/UPSNet.

1. Introduction

Relying on the advances in deep learning, computer vision systems have been substantially improved, especially in tasks such as semantic segmentation [40] and instance segmentation [15]. The former focuses on segmenting amorphous image regions which share similar texture or material such as grass, sky and road, whereas the latter focuses on segmenting countable objects such as people, bicycle and car. Since both tasks aim at understanding the visual scene at the pixel level, a shared model or representation could arguably be beneficial. However, the dichotomy of these two tasks lead to very different modeling strategies despite the inherent connections between them. For example, fully convolutional neural networks [26] are often adopted for semantic segmentation while proposal based detectors [31]

are frequently exploited for instance segmentation.

As an effort to leverage the possible complementariness of these two tasks and push the segmentation systems further towards real-world application, Kirillov *et al.* [17] unified them and proposed the so-called *panoptic segmentation* task. It is interesting to note that tasks with the same spirit have been studied under various names before deep learning became popular. Notable ones include image parsing [33], scene parsing [33] and holistic scene understanding [36]. In panoptic segmentation, countable objects (those that map to instance segmentation tasks well) are called *things* whereas amorphous and uncountable regions (those that map to semantic segmentation tasks better) are called *stuff*. For any pixel, if it belongs to *stuff*, the goal of a panoptic segmentation system is simply to predict its class label within the *stuff* classes. Otherwise the system needs to decide which instance it belongs to as well as which *thing* class it belongs to. The challenge of this task lies in the fact that the system has to give a unique answer for each pixel.

In this paper, we propose a unified panoptic segmentation network (UPSNet) to approach the panoptic segmentation problem as stated above. Unlike previous methods [17, 19] which have two separate branches designed for semantic and instance segmentation individually, our model exploits a single network as backbone to provide shared representations. We then design two heads on top of the backbone for solving these tasks simultaneously. Our semantic head builds upon deformable convolution [9] and leverages multi-scale information from feature pyramid networks (FPN) [22]. Our instance head follows the Mask R-CNN [15] design and outputs mask segmentation, bounding box and its associated class. As shown in experiments, these two lightweight heads along with the single backbone provide good semantic and instance segmentations which are comparable to separate models. More importantly, we design a panoptic head which predicts the final panoptic segmentation via pixel-wise classification of which the number of classes per image could vary. It exploits the logits from the above two heads and adds a new channel of logits corresponding to an extra *unknown* class. By doing so, it provides a better way of resolving the conflicts between semantic and instance segmentation. Moreover, our parameter-free panoptic head is very lightweight and could

---

*Equal contribution.
†This work was done when Hengshuang Zhao was an intern at Uber ATG.

be used with various backbone networks. It facilitates end-to-end training which is not the case for previous methods [17, 19]. To verify the effectiveness of our UPSNet, we perform extensive experiments on two public datasets: Cityscapes [6] and COCO [23]. Furthermore, we test it on our internal dataset which is similar in spirit to Cityscapes (*i.e.*, images are captured from ego-centric driving scenarios) but with significantly larger ($\approx 3\times$) size. Results on these three datasets manifest that our UPSNet achieves state-of-the-art performances and enjoys much faster inference compared to recent competitors.

2. Related Work

Semantic Segmentation: Semantic segmentation is one of the fundamental computer vision tasks that has a long history. Earlier work [11, 27] focused on introducing datasets for this task and showed the importance of global context by demonstrating the gains in bayesian frameworks, whether structured or free-form. Recent semantic segmentation methods that exploit deep convolutional feature extraction mainly approach this problem from either multi-scale feature aggregation [40, 26, 5, 14], or end-to-end structured prediction [2, 41, 1, 25, 4] perspectives. As context is crucial for semantic segmentation, one notable improvement to most convolutional models emerged from dilated convolutions [37, 38] which allows for a larger receptive field without the need of more free parameters. Pyramid scene parsing network (PSPNet) [40] that uses dilated convolutions in its backbone, and its faster variant [39] for real-time applications are widely utilized in practical applications. Based on FPN and PSP-Net, a multi-task framework is proposed in [35] and demonstrated to be versatile in segmenting a wide range of visual concepts.

Instance Segmentation: Instance segmentation deals not only with identifying the semantic class a pixel is associated with, but also the specific object instance that it belongs to. Beginning with the introduction of region-based CNN (R-CNN) [12], many early deep learning approaches to instance segmentation attacked the problem by casting the solution to instance segmentation as a two stage approach where a number of segment proposals are made, which is then followed by a voting between those proposals to choose the best [34, 7, 8, 13, 14, 30]. The common denominator for these methods is that the segmentation comes before classification, and are therefore slower. Li *et al*. [21] proposed a fully convolutional instance-aware segmentation method, where instance mask proposals [7] are married with fully convolutional networks [26]. Most recently, Mask R-CNN [15] introduced a joint approach to both mask prediction and recognition where one of the two parallel heads are dedicated to each task.

Panoptic Segmentation: Instance segmentation methods that focus on detection bounding box proposals, as mentioned above, ignore the classes that are not well suited for detection, *e.g.*, sky, street. On the other hand, semantic segmentation does not provide instance boundaries for classes like pedestrian and bicycle in a given image. Panoptic segmentation task, first coined by Kirillov *et al*. [17] unifies these tasks and defines an ideal output for *thing* classes as instance segmentations, as well as for

stuff classes as semantic segmentation. The baseline panoptic segmentation method introduced in [17] processes the input independently for semantic segmentation via a PSPNet, and for instance segmentation utilizing a Mask R-CNN [15], followed by simple heuristic decisions to produce a single void, stuff, or thing instance label per pixel. Recently, Li *et al*. [19] introduced a weakly- and semi-supervised panoptic segmentation method where they relieve some of the ground truth constraints by supervising thing classes using bounding boxes, and stuff classes by utilizing image level tags. De Gaus *et al*. [10] uses a single feature extraction backbone for the pyramid semantic segmentation head [40], and the instance segmentation head [15], followed by heuristics for merging pixel level annotations, effectively introducing an end-to-end version of [17] due to the shared backbone for the two task networks. Li *et al*. [20] propose the attention-guided unified network (AUNet) which leverages proposal and mask level attention to better segment the background. Similar post-processing heuristics as in [17] are used to generate the final panoptic segmentation. Li *et al*. [18] propose things and stuff consistency network (TASCNet) which constructs a binary mask predicting things vs. stuff for each pixel. An extra loss is added to enforce the consistency between things and stuff prediction.

In contrast to most of these methods, we use a single backbone network to provide both semantic and instance segmentation results. More importantly, we develop a simple yet effective panoptic head which helps accurately predict the instance and class label.

3. Unified Panoptic Sementation Network

In this section, we first introduce our model and then explain the implementation details. Following the convention of [17], we divide the semantic class labels into *stuff* and *thing*. Specifically, *thing* refers to the set of labels of instances (*e.g.* pedestrian, bicycle), whereas *stuff* refers to the rest of the labels that represent semantics without clear instance boundaries (*e.g.* street, sky). We denote the number of *stuff* and *thing* classes as N_{stuff} and N_{thing} respectively.

3.1. UPSNet Architecture

UPSNet consists of a shared convolutional feature extraction backbone and multiple heads on top of it. Each head is a sub-network which leverages the features from the backbone and serves a specific design purpose that is explained in further detail below. The overall model architecture is shown in Fig. 1.

Backbone: We adopt the original Mask R-CNN [15] backbone as our convolutional feature extraction network. This backbone exploits a deep residual network (ResNet) [16] with a feature pyramid network (FPN) [22].

Instance Segmentation Head: The instance segmentation head follows the Mask R-CNN design with a bounding box regression output, a classification output, and a segmentation mask output. The goal of the instance head is to produce instance
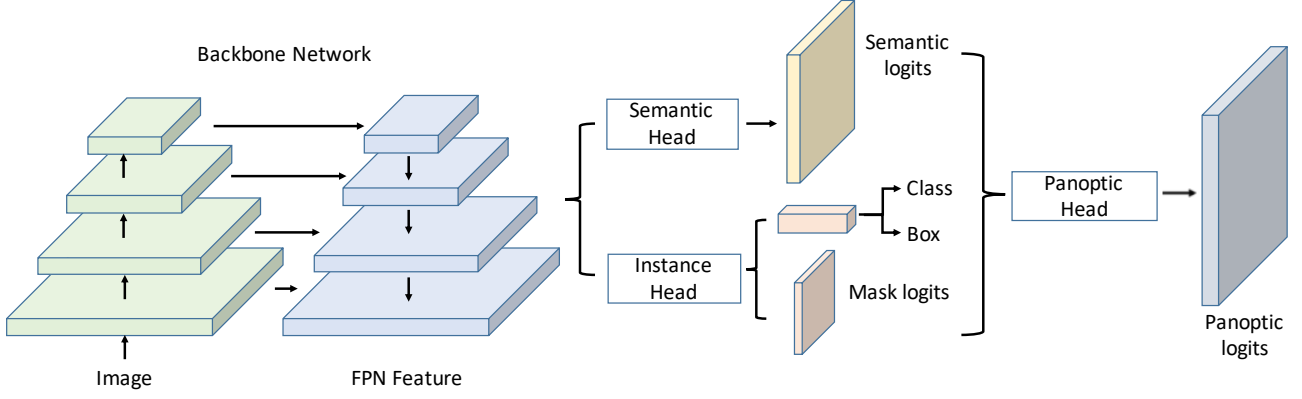
Figure 1: Overall architecture of our UPSNet.

aware representations that could identify *thing* classes better. Ultimately these representations are passed to the panoptic head to contribute to the logits for each instance.

Semantic Segmentation Head: The goal of the semantic segmentation head is to segment all semantic classes without discriminating instances. It could help improving instance segmentation where it achieves good results of *thing* classes. Our semantic head consists of a deformable convolution [9] based subnetwork which takes the multi-scale feature from FPN as input. In particular, we use P_2, P_3, P_4 and P_5 feature maps of FPN which contain 256 channels and are $1/4$, $1/8$, $1/16$ and $1/32$ of the original scale respectively. These feature maps first go through the same deformable convolution network independently and are subsequently upsampled to the $1/4$ scale. We then concatenate them and apply 1×1 convolutions with softmax to predict the semantic class. The architecture is shown in Fig. 2. As will be experimentally verified later, the deformable convolution along with the multi-scale feature concatenation provide semantic segmentation results as good as a separate model, *e.g.*, a PSPNet adopted in [17]. Semantic segmentation head is associated with the regular pixel-wise cross entropy loss. To put more emphasis on the foreground objects such as pedestrians, we also incorporate a RoI loss. During training, we use the ground truth bounding box of the instance to crop the logits map after the 1×1 convolution and then resize it to 28×28 following Mask R-CNN. The RoI loss is then the cross entropy computed over 28×28 patch which amounts to penalizing more on the pixels within instances for incorrect classification. As demonstrated in the ablation study later, we empirically found that this RoI loss helps improve the performance of panoptic segmentation without harming the semantic segmentation.

Panoptic Segmentation Head: Given the semantic and instance segmentation results from the above described two heads, we combine their outputs (specifically per pixel logits) in the panoptic segmentation head.

The logits from semantic head is denoted as X of which the channel size, height and width are $N_{\text{stuff}} + N_{\text{thing}}$, H and W respectively. X can then be divided along channel dimension into two tensors X_{stuff} and X_{thing} which are logits corresponding to

stuff and *thing* classes. For any image, we determine the number of instances N_{inst} according to the number of ground truth instances during training. During inference, we rely on a mask pruning process to determine N_{inst} which is explained in Section 3.2. N_{stuff} is fixed since number of *stuff* classes is constant across different images, whereas N_{inst} is not constant since the number of instances per image can be different. The goal of our panoptic segmentation head is to first produce a logit tensor Z which is of size $(N_{\text{stuff}} + N_{\text{inst}}) \times H \times W$ and then uniquely determine both the class and instance ID for each pixel.

We first assign X_{stuff} to the first N_{stuff} channels of Z to provide the logits for classifying *stuffs*. For any instance i, we have its mask logits Y_i from the instance segmentation head which is of size 28×28. We also have its box B_i and class ID C_i. During training B_i and C_i are ground truth box and class ID whereas during inference they are predicted by Mask R-CNN. Therefore, we can obtain another representation of i-th instance from semantic head X_{mask_i} by only taking the values inside box B_i from the channel of X_{thing} corresponding to C_i. X_{mask_i} is of size $H \times W$ and its values outside box B_i are zero. We then interpolate Y_i back to the same scale as X_{mask_i} via bilinear interpolation and pad zero outside the box to achieve a compatible shape with X_{mask_i}, denoted as Y_{mask_i}. The final representation of i-th instance is $Z_{N_{\text{stuff}}+i} = X_{\text{mask}_i} + Y_{\text{mask}_i}$. Once we fill in Z with representations of all instances, we perform a softmax along the channel dimension to predict the pixel-wise class. In particular, if the maximum value falls into the first N_{stuff} channel, then it belongs to one of stuff classes. Otherwise the index of the maximum value tells us the instance ID. The architecture is shown in Fig. 3. During training, we generate the ground truth instance ID following the order of the ground truth boxes we used to construct the panoptic logits. The panoptic segmentation head is then associated with the standard pixel-wise cross entropy loss.

During inference, once we predict the instance ID following the above procedure, we still need to determine the class ID of each instance. One can either use the class ID C_{inst} predicted by Mask R-CNN or the one predicted by the semantic head C_{sem}. As shown later in the ablation study, we resort to a better heuristic rule. Specifically, for any instance, we know which pixels correspond to it, *i.e.*, those of which the *argmax* of Z along channel
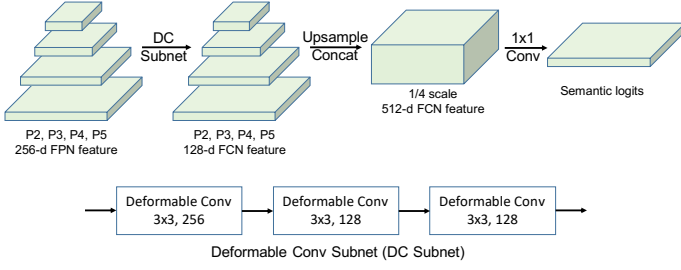
Figure 2: Architecture of our semantic segmentation head.

dimension equals to its instance ID. Among these pixels, we first check whether C_{inst} and C_{sem} are consistent. If so, then we assign the class ID as C_{inst}. Otherwise, we compute the mode of their C_{sem}, denoting as $\hat{C}_{sem}$. If the frequency of the mode is larger than 0.5 and $\hat{C}_{sem}$ belongs to *stuff*, then the predicted class ID is $\hat{C}_{sem}$. Otherwise, we assign the class ID as C_{inst}. In short, while facing inconsistency, we trust the majority decision made by the semantic head only if it prefers a *stuff* class. The justification of such a conflict resolution heuristic is that semantic head typically achieves very good segmentation results over *stuff* classes.

Unknown Prediction: In this section, we explain a novel mechanism to classify a pixel as the *unknown* class instead of making a wrong prediction. To motivate our design, we consider a case where a pedestrian is instead predicted as a bicycle. Since the prediction missed the pedestrian, the false negative (FN) value of pedestrian class will be increased by 1. On the other hand, predicting it as a bicycle will be increasing the false positive (FP) of bicycle class also by 1. Recall that, the panoptic quality (PQ) [17] metric for panoptic segmentation is defined as,

$$PQ = \underbrace{\frac{\sum_{(p,g)\in TP} \text{IoU}(p,g)}{|TP|}}_{SQ} \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{RQ},$$

which consist of two parts: recognition quality (RQ) and semantic quality (SQ). It is clear that increasing either FN or FP degrades this measurement. This phenomena extends to wrong predictions of the stuff classes as well. Therefore, if a wrong prediction is inevitable, predicting such pixel as *unknown* is preferred since it will increase FN of one class by 1 without affecting FP of the other class.

To alleviate the issue, we compute the logits of the extra unknown class as $Z_{unknown} = \max(X_{thing}) - \max(X_{mask})$ where X_{mask} is the concatenation of X_{mask_i} of all masks along channel dimension and of shape $N_{inst} \times H \times W$. The maximum is taken along the channel dimension. The rationale behind this is that for any pixel if the maximum of X_{thing} is larger than the maximum of X_{mask}, then it is highly likely that we are missing some instances (FN). The construction of the logits is shown in Fig. 3. To generate the ground truth for the *unknown* class, we randomly sample 30% ground truth masks and set them as *unknown* during training. In evaluating the metric, any pixel belonging to *unknown* is ignored, *i.e.*, setting to *void* which will not contribute to the results.
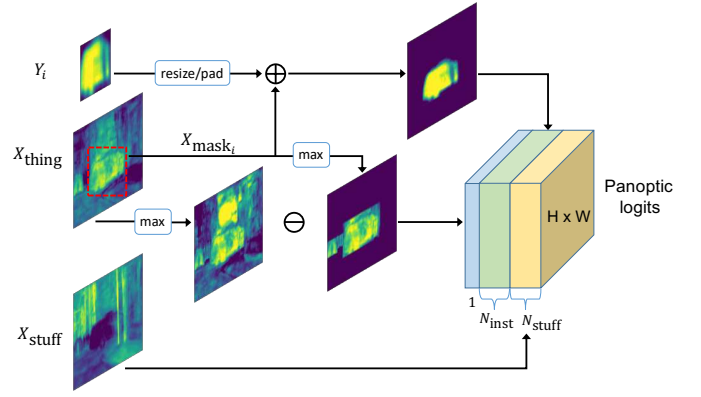

Figure 3: Architecture of our panoptic segmentation head.

3.2. Implementation Details

In this section, we explain the implementation details of UP-SNet. We follow most of settings and hyper-parameters of Mask R-CNN which will be introduced in the supplementary material. Hereafter, we only explain those which are different.

Training: We implement our model in PyTorch [28] and train it with 16 GPUs using the distributed training framework Horovod [32]. Images are preprocessed following [15]. Each mini-batch has 1 image per GPU. As mentioned, we use ground truth box, mask and class label to construct the logits of panoptic head during training. Our region proposal network (RPN) is trained end-to-end with the backbone whereas it was trained separately in [15]. Due to the high resolution of images, *e.g.*, 1024×2048 in Cityscapes, logits from semantic head and panoptic head are downsampled to $1/4$ of the original resolution. Although we do not fine-tune batch normalization (BN) layers within the backbone for simplicity, we still achieve comparable results with the state-of-the-art semantic segmentation networks like PSPNet. Based on common practice in semantic [4, 40] and instance segmentation [29, 24], we expect the performance to be further improved with BN layers fine-tuned. Our UPSNet contains 8 loss functions in total: semantic segmentation head (whole image and RoI based pixel-wise classification losses), panoptic segmentation head (whole image based pixel-wise classification loss), RPN (box classification, box regression) and instance segmentation head (box classification, box regression and mask segmentation). Different weighting schemes on these multi-task loss functions could lead to very different training results. As shown in the ablation study, we found the loss balance strategy, *i.e.*, assuring the scales of all losses are roughly on the same order of magnitude, works well in practice.

Inference: During inference, once we obtained output boxes, masks and predicted class labels from the instance segmentation head, we apply a mask pruning process to determine which mask will be used for constructing the panoptic logits. In particular, we first perform the class-agnostic non-maximum suppression with the box IoU threshold as 0.5 to filter out some overlapping boxes. Then we sort the predicted class probabilities of the remaining boxes and keep those whose probability are larger than 0.6. For

Models	PQ	SQ	RQ	PQ[Th]	PQ[St]	mIoU	AP
JSIS-Net [10]	26.9	72.4	35.7	29.3	23.3	-	-
RN50-MR-CNN	38.6	76.4	47.5	46.2	27.1	-	-
MR-CNN-PSP	41.8	**78.4**	51.3	47.8	32.8	53.9	34.2
Ours	**42.5**	78.0	**52.4**	**48.5**	**33.4**	**54.3**	**34.3**

Multi-scale	PQ	SQ	RQ	PQ[Th]	PQ[St]	mIoU	AP
MR-CNN-PSP-M	42.2	78.5	51.7	47.8	33.8	55.3	34.2
Ours-M	**43.2**	**79.2**	**52.9**	**49.1**	**34.1**	**55.8**	**34.3**

Table 1: Panoptic segmentation results on COCO. Superscripts Th and St stand for thing and stuff. '-' means inapplicable.

each class, we create a canvas which is of the same size as the image. Then we interpolate masks of that class to the image scale and paste them onto the corresponding canvas one by one following the decreasing order of the probability. Each time we copy a mask, if the intersection between the current mask and those already existed over the size of the current mask is larger than a threshold, we discard this mask. Otherwise we copy the non-intersecting part onto the canvas. The threshold of this intersection over itself is set to 0.3 in our experiments. Logits from the semantic segmentation head and panoptic segmentation head are of the original scale of the input image during inference.

4. Experiments

In this section, we present the experimental results on COCO [23], Cityscapes [6] and our internal dataset.

COCO We follow the setup of COCO 2018 panoptic segmentation task which consists of 80 and 53 classes for thing and stuff respectively. We use train2017 and val2017 subsets which contain approximately 118k training images and 5k validation images.

Cityscapes Cityscapes has 5000 images of ego-centric driving scenarios in urban settings which are split into 2975, 500 and 1525 for training, validation and testing respectively. It consists of 8 and 11 classes for *thing* and *stuff*.

Our Dataset We also use an internal dataset which is similar to Cityscapes and consists of 10235 training, 1139 validation and 1186 test images of ego-centric driving scenarios. Our dataset consists of 10 and 17 classes for *thing* (*e.g.*, car, bus) and *stuff* (*e.g.*, building, road) respectively.

Experimental Setup For all datasets, we report results on the validation set. To evaluate the performance, we adopt panoptic quality (PQ), recognition quality (RQ) and semantic quality (SQ) [17] as the metrics. We also report average precision (AP) of mask prediction, mean IoU of semantic segmentation on both *stuff* and *thing* and the inference run-time for comparison. At last, we show results of ablation study on various design components of our model. Full results with all model variants are shown in the supplementary material.

We set the learning rate and weight decay as 0.02 and 0.0001 for all datasets. For COCO, we train for 90K iterations and decay the learning rate by a factor of 10 at 60K and 80K iterations.

For Cityscapes, we train for 12K iterations and apply the same learning rate decay at 9K iterations. For our dataset, we train for 36K iterations and apply the same learning rate decay at 24K and 32K iterations. Loss weights of semantic head are 0.2, 1.0 and 1.0 on COCO, Cityscapes and ours respectively. RoI loss weights are one fifth of those of semantic head. Loss weights of panoptic head are 0.1, 0.5 and 0.3 on COCO, Cityscapes and ours respectively. All other loss weights are set to 1.0.

We mainly compare with the combined method used in [17]. For a fair comparison, we adopt the model which uses a Mask R-CNN with a ResNet-50-FPN and a PSPNet with a ResNet-50 as the backbone and apply the combine heuristics to compute the panoptic segmentation. We denote our implementation of the combined method as 'MR-CNN-PSP' and its multi-scale testing version as 'MR-CNN-PSP-M'. Unless specified otherwise, the combined method hereafter refers to 'MR-CNN-PSP'. For PSP-Net, we use 'poly' learning rate schedule as in [3] and train 220K, 18K and 76K on COCO, Cityscapes and our dataset with mini-batch size 16. We test all available models with the multi-scale testing. Specifically, we average the multi-scale logits of PSPNet for the combined method and the ones of semantic segmentation head for our UPSNet. For simplicity, we just use single scale testing on Mask R-CNN of the combined method and our instance segmentation head. During evaluation, due to the sensitivity of PQ with respect to RQ, we predict all *stuff* segments of which the areas are smaller than a threshold as *unknown*. The thresholds on COCO, Cityscapes and our dataset are 4096, 2048 and 2048 respectively. To be fair, we apply this area thresholding to all methods.

4.1. COCO

We compare with several recent published methods including JSIS-Net [10], RN50-MR-CNN [1] and the combined method [17] on COCO. Since authors in [17] do not report results on COCO, we use our MR-CNN-PSP model as the alternative to do the experiments. JSIS-Net uses a ResNet-50 wheres RN50-MR-CNN uses two separate ResNet-50-FPNs as the backbone. Our UP-SNet adopts a ResNet-50-FPN as the backbone. In order to better leverage context information, we use a global average pooling over the feature map of the last layer in the 5-th stage of ResNet ('res5'), reduce its dimension to 256 and add back to FPN before producing P_5 feature map. Table 1 shows the results of all metrics. The mIoU metric is computed over the 133 classes of stuff and thing in the COCO 2018 panoptic segmentation task which is different from previous 172 classes of COCO-Stuff. We are among the first to evaluate mIoU on this 133-class subset. From the table, we can see that our UPSNet achieves better performance in all metrics except the SQ. It is typically the case that the an increase in RQ leads to the slight decrease of SQ since we include more TP segments which could have possibly lower IoU. Note that even with multi-scale testing, MR-CNN-PSP is still worse than ours on PQ. Moreover, from the mIoU column, we can see that the performance of our semantic head

---

[1]https://competitions.codalab.org/competitions/19507#results

Models	backbone	PQ	SQ	RQ	PQTh	SQTh	RQTh	PQSt	SQSt	RQSt
Megvii (Face++)	ensemble model	53.2	83.2	62.9	62.2	85.5	72.5	39.5	79.7	48.5
Caribbean	ensemble model	46.8	80.5	57.1	54.3	81.8	65.9	35.5	78.5	43.8
PKU_360	ResNeXt-152-FPN	46.3	79.6	56.1	58.6	83.7	69.6	27.6	73.6	35.6
JSIS-Net [10]	ResNet-50	27.2	71.9	35.9	29.6	71.6	39.4	23.4	72.3	30.6
AUNet [20]	ResNeXt-152-FPN	46.5	**81.0**	56.1	**55.9**	**83.7**	**66.3**	32.5	77.0	40.7
Ours	ResNet-101-FPN	**46.6**	80.5	**56.9**	53.2	81.5	64.6	**36.7**	**78.9**	**45.3**

Table 2: Panoptic segmentation results on MS-COCO 2018 *test-dev*. The top 3 rows contain results of top 3 models taken from the official leadboard.

Models	PQ	SQ	RQ	PQTh	PQSt	mIoU	AP
Li *et al.* [19]	53.8	-	-	42.5	62.1	71.6	28.6
MR-CNN-PSP	58.0	79.2	71.8	52.3	62.2	75.2	32.8
TASCNet [18]	55.9	-	-	50.5	59.8	-	-
Ours	**59.3**	**79.7**	**73.0**	**54.6**	**62.7**	**75.2**	**33.3**
TASCNet-COCO [18]	59.2	-	-	56.0	61.5	-	-
Ours-COCO	**60.5**	**80.9**	**73.5**	**57.0**	**63.0**	**77.8**	**37.8**
Multi-scale	PQ	SQ	RQ	PQTh	PQSt	mIoU	AP
Kirillov *et al.* [17]	61.2	80.9	74.4	54.0	**66.4**	**80.9**	36.4
MR-CNN-PSP-M	59.2	79.7	73.0	52.3	64.2	76.9	32.8
Ours-M	60.1	80.3	73.5	55.0	63.7	76.8	33.3
Ours-101-M-COCO	**61.8**	**81.3**	**74.8**	**57.6**	64.8	79.2	**39.0**

Table 3: Panoptic segmentation results on Cityscapes. '-COCO' means the model is pretrained on COCO. '-101' means the model uses ResNet-101 as the backbone. Unless specified, all models use ResNet-50 as the backbone and are pretrained on ImageNet.

is even better than a separate PSPNet which verifies its effectiveness. With multi-scale testing, both MR-CNN-PSP and UPSNet are improved and ours is still better. We also add the comparisons on the *test-dev* of MS-COCO 2018 in Table 2. Although we just use ResNet-101 as the backbone, we achieve slightly better results compared to the recent AUNet [20] which uses ResNeXt-152. We also list the top three results on the leadboard which uses ensemble and other tricks. It is clear from the table that we are on par with the second best model without using any such tricks. In terms of the model size, RN50-MR-CNN, MR-CNN-PSP and UPSNet consists of 71.2M, 91.6M and 46.1M parameters respectively. Therefore, our model is significantly lighter. We show visual examples of panoptic segmentation on this dataset in the first two rows of Fig. 4. From the 1-st row of the figure, we can see that the combined method completely ignores the cake and other objects on the table whereas ours successfully segments them out. This is due to the inherent limitations of the combine heuristic which first pastes the high confidence segment, *i.e.*, table, and then ignores all highly overlapped objects thereafter.

4.2. Cityscapes

We compare our model on Cityscapes with Li *et al.* [19], the combined method [17] and TASCNet [18]. Note that the method in [19] uses a ResNet-101 as the backbone whereas all other re-

Models	PQ	SQ	RQ	PQTh	PQSt	mIoU	AP
MR-CNN-PSP	45.5	77.1	57.6	40.1	48.7	70.5	29.6
Ours	**47.1**	**77.1**	**59.4**	**43.8**	**49.0**	**70.8**	**30.4**

Table 4: Panoptic segmentation results on our dataset.

ported methods use ResNet-50 within their backbones. We use 2 deformable convolution layers for the semantic head. The results are reported in Table 3. It is clear from the table that both our UPSNet and MR-CNN-PSP significantly outperform the method in [19], especially on PQTh. This may possibly be caused by the fact that their CRF based instance subnetwork performs worse compared to Mask R-CNN on instance segmentation. Under the same single scale testing, our model achieves better performance than the combined method. Although multi-scale testing significantly improves both the combined method and UPSNet, ours is still slightly better. Results reported in [17] are different from the ones of our MR-CNN-PSP-M since: 1) they use ResNet-101 as the backbone for PSPNet; 2) they pre-train Mask R-CNN on COCO and PSPNet on extra coarsely labeled data. We also have a model variant, denoting as 'Ours-101-M', which adopts ResNet-101 as the backbone and pre-trains on COCO. As you can see, it outperforms the reported metrics of the combined method. We show the visual examples of panoptic segmentation on this dataset in the 3-rd and 4-th rows of Fig. 4. From the 3-rd row of the figure, we can see that the combined method tends to produce large black area, *i.e.*, *unknown*, whenever the instance and semantic segmentation conflicts with each other. In contrast, our UPSNet resolves the conflicts better. Moreover, it is interesting to note that some *unknown* prediction of our model has the vertical or horizontal boundaries. This is caused by the fact that instance head predicts nothing whereas semantic head predicts something for these out-of-box areas. The logits of *unknown* class will then stand out by design to avoid contributing to both FP and FN as described in section 3.1.

4.3. Our Dataset

Last but not least, we compare our model on our own dataset with the combined method [17]. All reported methods use ResNet-50 within their backbones. We use 2 deformable convolution layers for the semantic head. The results are reported in Table 4. We can observe that similar to COCO, our model performs

Model	COCO	Cityscapes	Our Dataset
Img. Size	800×1300	1024×2048	1200×1920
MR-CNN-PSP	188 (186 + 2)	1105 (583 + 522)	1016 (598 + 418)
Ours	**171**	**236**	**264**
Speedup	$10\% \times$	$368\% \times$	$285\% \times$

Table 5: Run time (ms) comparison. Note the bracket below the MR-CNN-PSP results contains their breakdown into the network inference (left) and the combine heurisitc (right).

significantly better than the combined method on all metrics except SQ. We show the visual examples of panoptic segmentation on this dataset in the last two rows of Fig. 4. From the examples, similar observations as COCO and Cityscapes are found.

4.4. Run Time Comparison

We compare the run time of inference on all three datasets in Table 5. We use a single NVIDIA GeForce GTX 1080 Ti GPU and an Intel Xeon E5-2687W CPU (3.00GHz). All entries are averaged over 100 runs on the same image with single scale test. For COCO, the PSPNet within the combined method uses the original scale. We also list the image size on each dataset. It is clearly seen in the table that as the image size increases, our UPSNet is significantly faster in run time. For example, the combined method takes about $3\times$ time than ours.

4.5. Ablation Study

We perform extensive ablation studies on COCO dataset to verify our design choices as listed in Table 6. Empty cells in the table indicate the corresponding component is not used. All evaluation metrics are computed over the output of the panoptic head on the validation set.

Panoptic Head: Since the inference of our panoptic head is applicable as long as we have outputs from both semantic and instance segmentation heads, we can first train these two heads simultaneously and then directly evaluate the panoptic head. We compare the results with the ones obtained by training all three heads. By doing so, we can verify the gain of training the panoptic head over treating it as a post processing procedure. From the first two rows of Table 6, we can see that training the panoptic head does improve the PQ metric.

Instance Class Assignment: Here, we focus on different alternatives of assigning instance class. We compare our heuristic as described in section 3.1 with the one which only trusts the predicted class given by the instance segmentation head. As you can see from the 2-nd and 3-rd rows of the Table 6, our instance class assignment is better.

Loss Balance: We investigate the weighting scheme of loss functions. Recall that without the proposed RoI loss, our UPSNet contains 7 loss functions. In order to weight them, we follow the principle of loss balance, *i.e.*, making sure their values are roughly on the same order of magnitude. In particular, with loss

Pano.	Our ICA	Loss Bal.	RoI Loss	Unk.	GT Box	GT ICA	GT Seg.	PQ	PQTh	PQSt
								41.2	47.6	31.6
✓								41.6	47.6	32.5
✓	✓							41.7	47.7	32.8
✓	✓	✓						42.3	48.4	33.1
✓	✓	✓	✓					42.3	48.3	33.2
✓	✓	✓	✓	✓				42.5	48.5	33.4
✓	✓	✓	✓	✓	✓			46.7	55.3	33.6
✓		✓	✓		✓	✓		53.0	64.6	35.5
✓	✓	✓	✓	✓			✓	72.0	58.6	92.3

Table 6: Ablation study on COCO dataset. 'Pano.', 'Loss Bal.', 'Unk.' and 'ICA' stand for training with panoptic loss, loss balance, *unknown* prediction and instance class assignment respectively.

balance, we set the weights of semantic and panoptic losses as 0.2 and 0.1 and the rest ones as 1.0. Without loss balance, we set the weights of both semantic and panoptic losses as 0.1 and the rest ones as 1.0. The 3-rd and 4-th rows of Table 6 show that introducing the loss balance improves the performance.

RoI Loss & Unknown Prediction: Here, we investigate the effectiveness of our RoI loss function over the semantic head and the unknown prediction. From 4-th and 5-th rows of Table 6, one can conclude that adding such a new loss function does slightly boost the PQSt. From 5-th and 6-th rows of Table 6, along with the RoI loss, predicting unknown class improves the metrics significantly.

ORACLE Results: We also explore the room for improvement of the current system by replacing some inference results with the ground truth (GT) ones. Specifically, we study the box, instance class assignment and semantic segmentation results which correspond to GT Box, GT ICA and GT Seg. columns in Table 6. It is clear from the table that using GT boxes along with our predicted class probabilities improves PQ which indicates that better region proposals are required to achieve higher recall. On top of the GT boxes, using the GT class assignment greatly improves the PQ, *e.g.*, $\approx +7.0$. The imperfect PQTh indicates that our mask segmentation is not good enough. Moreover, using the GT semantic segmentation gives the largest gain of PQ, *i.e.*, $+29.5$, which highlights the importance of improving semantic segmentation. PQSt is imperfect since we resize images during inference which causes the misalignment with labels. It is worth noticing that increasing semantic segmentation also boosts PQTh for 10 points. This is because our model leverages semantic segments while producing instance segments. However, it is not the case for the combined method as its predicted instance segments only relies on the instance segmentation network.

5. Conclusion

In this paper, we proposed the UPSNet which provides a unified framework for panoptic segmentation. It exploits a single
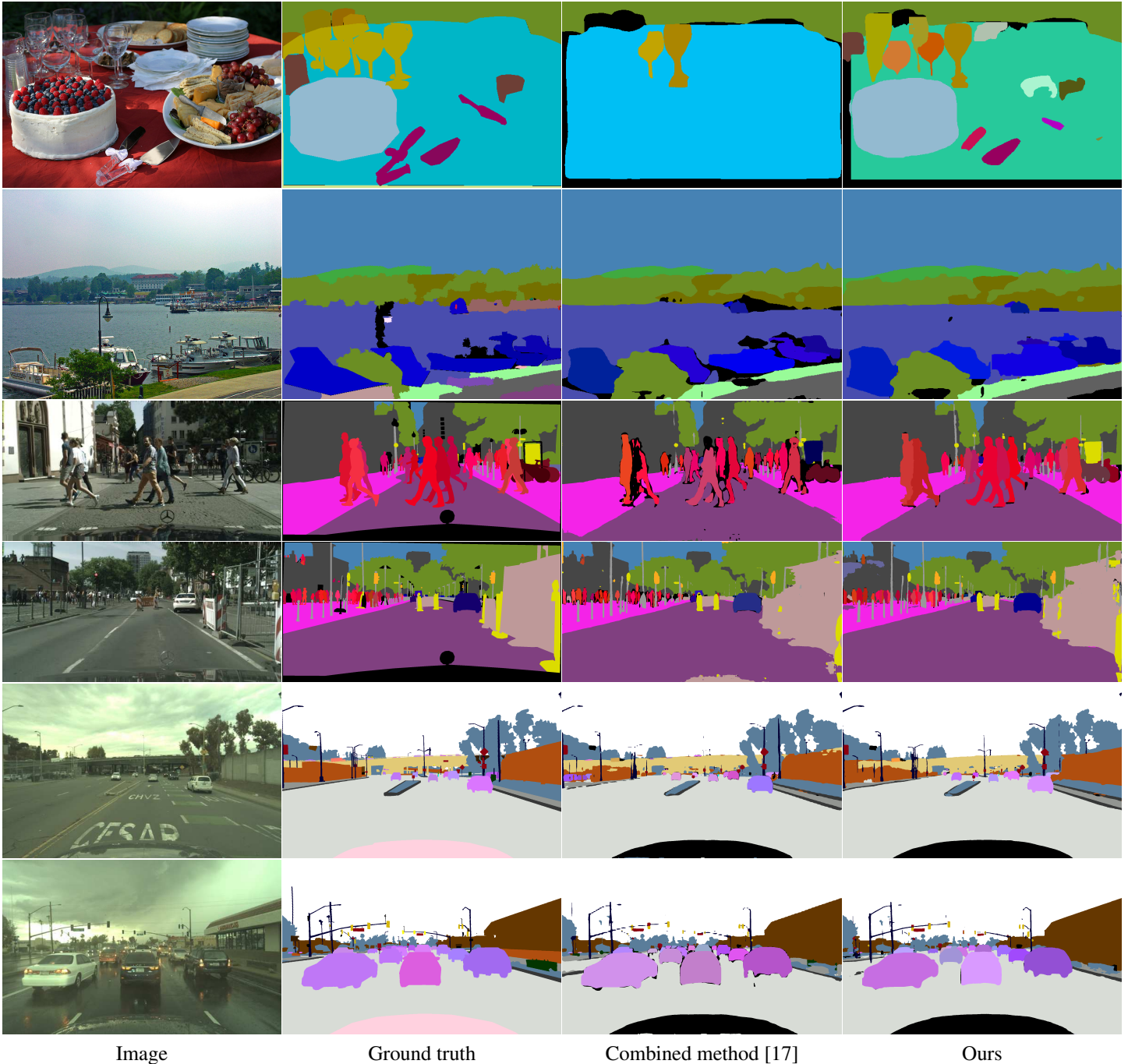
Figure 4: Visual examples of panoptic segmentation. 1-st and 2-nd rows are from COCO. 3-rd and 4-th rows are from Cityscapes. 5-th and 6-th rows are from our internal dataset.

| Image | Ground truth | Combined method [17] | Ours |

backbone network and two lightweight heads to predict semantic and instance segmentation in one shot. More importantly, our parameter-free panoptic head leverages the logits from the above two heads and has the flexibility to predict an extra *unknown* class. It handles the varying number of classes per image and enables back propagation for the bottom representation learning. Empirical results on three large datasets show that our UPSNet achieves state-of-the-art performance with significantly faster in-

ference compared to other methods. In the future, we would like to explore more powerful backbone networks and smarter parameterization of panoptic head.

References

[1] Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, and Philip HS Torr. Higher order conditional random fields in deep neural networks. In *ECCV*, 2016.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 2018.

[4] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[5] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *CVPR*, 2016.

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[7] Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016.

[8] Jifeng Dai, Kaiming He, and Jian Sun. Convolutional feature masking for joint object and stuff segmentation. In *CVPR*, 2015.

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

[10] Daan de Geus, Panagiotis Meletis, and Gijs Dubbelman. Panoptic segmentation with a joint semantic and instance segmentation network. *arXiv preprint arXiv:1809.02110*, 2018.

[11] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[12] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[13] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.

[14] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. *arXiv preprint arXiv:1801.00868*, 2018.

[18] Jie Li, Allan Raventos, Arjun Bhargava, Takaaki Tagawa, and Adrien Gaidon. Learning to fuse things and stuff. *arXiv preprint arXiv:1812.01192*, 2018.

[19] Qizhu Li, Anurag Arnab, and Philip HS Torr. Weakly-and semi-supervised panoptic segmentation. In *ECCV*, 2018.

[20] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-guided unified network for panoptic segmentation. *arXiv preprint arXiv:1812.03904*, 2018.

[21] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, 2017.

[22] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[24] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018.

[25] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *ICCV*, 2015.

[26] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[27] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS Workshop*, 2017.

[29] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. Megdet: A large mini-batch object detector. *arXiv preprint arXiv:1711.07240*, 2017.

[30] Mengye Ren and Richard S. Zemel. End-to-end instance segmentation with recurrent attention. In *CVPR*, 2017.

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[32] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in TensorFlow. *arXiv preprint arXiv:1802.05799*, 2018.

[33] Zhuowen Tu, Xiangrong Chen, Alan L Yuille, and Song-Chun Zhu. Image parsing: Unifying segmentation, detection, and recognition. *IJCV*, 2005.

[34] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *IJCV*, 2013.

[35] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018.

[36] Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *CVPR*, 2012.

[37] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[38] Fisher Yu, Vladlen Koltun, and Thomas A Funkhouser. Dilated residual networks. In *CVPR*, 2017.

[39] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *ECCV*, 2018.

[40] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.

[41] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015.