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

Technical Report: Fast Panoptic Segmentation Network

Daan de Geus Panagiotis Meletis Gijs Dubbelman
Eindhoven University of Technology
{d.c.d.geus, p.c.meletis, g.dubbelman}@tue.nl

Abstract

In this technical report, we present the results for our Fast Panoptic Segmentation Network (FPSNet) on the COCO dataset. FPSNet is an end-to-end method for fast panoptic segmentation, that does not require computationally costly instance mask predictions and merging heuristics. This is achieved by introducing a novel convolutional neural network module, which casts the panoptic segmentation task into a custom dense pixel-wise classification task, that assigns a class label or an instance id to each pixel. Our method achieves a Panoptic Quality score of 29.8% on the COCO test-dev split (29.3% on the validation set), at prediction speeds of 24 frames per second on a single GPU. For PQ scores of 29.1% and 25.1% on the validation set, we achieve prediction speeds of 34 and 45 frames per second, respectively. The results have been submitted to the Panoptic Segmentation Challenge of the Joint COCO and Mapillary Workshop at ICCV 2019.

1. Introduction

When the task of panoptic segmentation was introduced, a baseline method was proposed [8]. This baseline method combined the outputs of state-of-the-art semantic segmentation and instance segmentation networks using heuristics. In terms of required computational time and resources, this is not very efficient since it requires multiple networks. To improve on this, several methods have been introduced that perform panoptic segmentation with a single deep neural network [3, 7, 14, 13, 9]. These approaches make instance segmentation and semantic segmentation predictions in a single network, and then fuse these predictions using heuristics to generate a single panoptic output. Although this is faster than using multiple separate networks, these methods still rely on computationally costly merging heuristics and need to make expensive instance mask predictions.

To achieve fast panoptic segmentation, we introduce the Fast Panoptic Segmentation Network (FPSNet) [4]. This network makes panoptic segmentation predictions without relying on expensive merging heuristics or instance mask predictions. FPSNet achieves this by learning the panoptic task in an end-to-end fashion, and treating it as a dense segmentation problem. In Section 2, we will explain how FPSNet is able to do this. Thereafter, we will discuss the experiments and our results on the COCO dataset [12] in Section 3, followed by conclusions in Section 4.

This technical report presents a summary of [4], in which FPSNet is introduced and described in more detail.

2. FPSNet

To achieve fast panoptic segmentation, we aim for a method that does not require making computationally costly instance mask predictions or merging heuristics. In current methods, these heuristics are needed because there are are pixels for which the semantic segmentation and instance segmentation predictions are conflicting. Instead of relying on heuristics to solve these conflicts, we treat panoptic segmentation as a dense pixel-wise classification task, since there can only be one prediction for each pixel.

We achieve this by introducing a novel convolutional neural network module, which we call the *panoptic head*. This head has two inputs: 1) a feature map on which we can perform dense segmentation, and 2) attention masks indicating the presence of *things* instances, and the classes corresponding to those instances, which we obtain from a regular bounding box object detector. From this, the model is trained to simultaneously 1) perform semantic segmentation for *stuff* classes, 2) 'morph' the attention masks into complete pixel-wise instance masks for *things* instances, and 3) output the predictions for both the *stuff* classes and *things* instances in a single map, on which we can do pixel-wise classification. This module is trained end-to-end in a single network, together with the required feature extractor and bounding box object detector.

We use the single-stage RetinaNet bounding box object

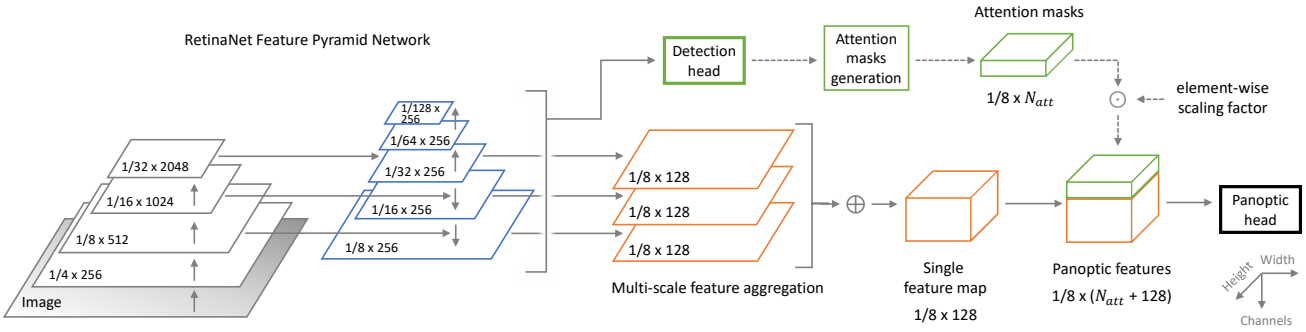

Figure 1: Overview of the FPSNet architecture. The dimensions indicate spatial stride on the input image (e.g. $1/8$), and the feature depth (e.g. 128). $\oplus$ denotes element-wise addition. During training, losses are applied only at the two emphasized blocks (detection head and panoptic head). The dotted lines denote that there is no gradient flow in this path during training.

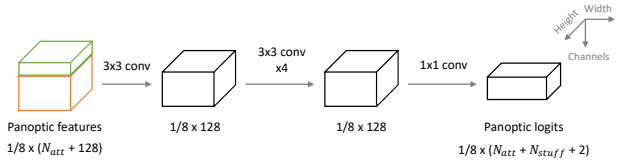

Figure 2: The panoptic head architecture.

detector [11], with the ResNet-50-FPN backbone [6, 10].

An overview of the FPSNet architecture is depicted in Figure 1.

2.1. Panoptic module

In our novel panoptic module for fast panoptic segmentation, we assume that we have bounding box object detections from a regular object detector, as well as a single feature map to apply dense image segmentation. The bounding boxes are used to generate attention masks that indicate the location of *things* in the image, and determine the order of the *things* in the output. The attention masks are first shuffled, then concatenated to the feature map, and finally applied to a fully convolutional network, i.e. the panoptic head. The panoptic head architecture is shown in Figure 2.

This panoptic head outputs, for each pixel, a value (i.e. a score) for all N_{stuff} *stuff* classes and N_{att} attention masks indicating *things* instances. The panoptic head is trained to morph the attention masks into coherent *things* instance masks, and apply regular semantic segmentation for the *stuff* classes. In essence, in the output features of the panoptic head, the *stuff* classes and the *things* instance *ids* are treated the same.

During training, we construct the ground-truth for *things* instances such that the order of the instances is the same as the order of the attention masks. As a result, the network learns to output the *things* instances in the same order as the input attention masks. The *stuff* classes are trained as in a regular semantic segmentation network.

During inference, we simply pick the output with the highest score for each pixel, i.e. applying *argmax*. We then have a single *stuff* class or *things id* prediction for each pixel. Since each *things id* can be related back to a unique attention mask and therefore a *things* class, we have a *things* class for each predicted *things id*, and we have an output in the panoptic segmentation format.

Note that our method does not make explicit instance mask predictions, and does not rely on merging heuristics to generate a final panoptic output. In [4], our method is described in more detail.

3. Experiments

3.1. Implementation details

We train FPSNet on the COCO train set [12]. The backbone of the network is initialized with weights from a network pretrained on ImageNet [5]. We resize the images so that the height is equal to the width, and then take crops of 512 x 512 pixels during training. During inference, we resize all images to 512 x 512 pixels. Because of limited time and resources, we train our network on a single Nvidia Titan RTX GPU, with a batch size of 8, for 400k steps.

We use a polynomial decay learning rate schedule (as in [1]) with an initial learning rate of 0.01 and a decay of 0.9. Weight decay is 0.0001. As described in [4], we have two losses, i.e. the RetinaNet detection loss, L_{det}, and the softmax cross-entropy panoptic loss L_{pan}. Both losses are weighted equally. We set the number of attention masks, N_{att}, to 50.

When applying FPSNet to the COCO dataset, we find that certain small changes to the method lead to improved accuracy and reduced prediction time:

1. Instead of applying a regular Non-Maximum Suppression (NMS) algorithm to the detections by RetinaNet, which is applied to all classes separately, we use a

Method	PQ	SQ	RQ	PQ^{Th}	SQ^{Th}	RQ^{Th}	PQ^{St}	SQ^{St}	RQ^{St}	Prediction time (ms)
$FPSNet_{SR}$ (ours)	29.8	74.3	37.9	33.1	76.1	41.6	24.9	71.5	32.4	42

Table 1: The best overall results on the COCO test-dev split. St = *stuff* classes; Th = *things* classes. SR indicates stuff removal.

			Accuracy	*Prediction time*		
Method	Backbone	Resolution	PQ (%) ↑	Inference (ms) ↓	Post-processing (ms) ↓	Total (ms) ↓
DeeperLab [15]	LW-MNV2	320 x 320	17.5	29	8	37
FPSNet (ours)	ResNet-50-FPN	320 x 320	23.7	33	*n/a*	33
$FPSNet_{SR+FN}$ (ours)	ResNet-50-FPN	320 x 320	**25.1**	**21**	1	**22**
DeeperLab [15]	LW-MNV2	640 x 640	24.1	48	25	73
DeeperLab [15]	W-MNV2	640 x 640	27.9	58	25	83
DeeperLab [15]	Xception-71	640 x 640	33.8	94	25	119
UPSNet [14]	ResNet-50-FPN	800 x 1300	**42.5**	167	*n/a*	167
FPSNet (ours)	ResNet-50-FPN	512 x 512	27.3	41	*n/a*	41
$FPSNet_{SR}$ (ours)	ResNet-50-FPN	512 x 512	29.3	41	1	42
$FPSNet_{SR+FN}$ (ours)	ResNet-50-FPN	512 x 512	29.1	**28**	1	**29**

Table 2: Speed and accuracy results for FPSNet and other methods on the COCO validation set. Values are as reported in the respective papers, unless indicated otherwise. (L)W-MNV2 is (Light) Wider MobileNetV2 [15]. SR indicates stuff removal; FN indicates Faster NMS.

faster NMS algorithm that considers bounding boxes of all classes at once, to reduce prediction time.

2. As is common practice [14], we remove *stuff* predictions if the total pixel count for that specific *stuff* class is below a certain threshold, to remove potentially wrong and unlikely *stuff* predictions. We set this threshold to 8192.

We provide ablations in Section 3.4.

3.2. Metrics

We evaluate FPSNet in terms of both accuracy and speed. For accuracy, we use the standard Panoptic Quality metric for panoptic segmentation [8]. For speed, we report single image inference time on an Nvidia Titan RTX GPU, averaged over all images in the validation set.

3.3. Main results

In Table 1, we provide detailed results for our method on the COCO test-dev split. To compare FPSNet with existing methods, we present PQ scores and prediction times for FPSNet and existing methods that report prediction times in Table 2, for the COCO validation set. All scores and prediction times are as reported in the respective papers.

From Table 2, it follows that FPSNet is faster than existing panoptic segmentation methods. Moreover, with a total prediction time of 29 ms and a PQ score of 29.1%, we outperform the versions of DeeperLab [15] that take more than $2\times$ as long to make predictions, and have higher input resolutions. FPSNet is still significantly outperformed by UPSNet [14] in terms of Panoptic Quality. However, UPSNet has a much higher input resolution. Because of this and the fact that the method makes costly instance mask predictions, the total prediction time is more than $4\times$ higher than that of FPSNet.

At input resolutions of 320 x 320 pixels, we outperform DeeperLab [15] with over 7 points on the PQ metric, while being 15 milliseconds faster.

It is likely that FPSNet achieves higher accuracy if it is trained with higher input resolutions and for a longer time, as this was the case for training on the Cityscapes dataset [2]. This was currently not done, because of limited time and resources.

Qualitative results on the COCO validation set are shown in Figure 3.

3.4. Ablations

In Table 2, we report scores for different versions of our method. At input resolutions of 512 x 512 pixels, $FPSNet_{SR}$, which uses *stuff* removal post-processing, clearly outperforms the *regular* FPSNet by 2.0 points, while only adding 1 ms in post-processing. Moreover, introducing the faster NMS version reduces the prediction time by 13 ms, while only slightly impacting the PQ score. For input resolutions of 320 x 320 pixels, we see the same effect.

4. Conclusion

In this work, we presented a brief overview of our Fast Panoptic Segmentation Network, a fast end-to-end method for panoptic segmentation. We have shown that our method is faster than current panoptic segmentation methods, and

Figure 3: Example predictions by FPSNet on the COCO validation set. Each color indicates a different *things* instance or *stuff* class.

that we achieve a Panoptic Quality score of 29.8% on the test-dev set and 29.3% on the validation set, at prediction speeds of 24 frames per second. For slightly lower Panoptic Quality scores, we achieve speeds of up to 45 frames per second. It is highly likely that the accuracy of the network can be improved further by training longer and using higher resolution images. A more detailed explanation of FPSNet can be found in [4].

References

[1] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, Apr. 2018. 2

[2] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, June 2016. 3

[3] D. de Geus, P. Meletis, and G. Dubbelman. Panoptic Segmentation with a Joint Semantic and Instance Segmentation Network. *arXiv preprint arXiv:1809.02110*, Sept. 2018. 1

[4] D. de Geus, P. Meletis, and G. Dubbelman. Fast Panoptic Segmentation Network. *arXiv preprint arXiv:1910.03892*, Oct. 2019. 1, 2, 4

[5] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. 2

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. 2

[7] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic Feature Pyramid Networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1

[8] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 3

[9] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-Guided Unified Network for Panoptic Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1

[10] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature Pyramid Networks for Object Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, July 2017. 2

[11] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal Loss for Dense Object Detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, Oct. 2017. 2

[12] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV 2014*, pages 740–755, 2014. 1, 2

[13] Lorenzo Porzi, Samuel Rota Bulo, Aleksander Colovic, and Peter Kontschieder. Seamless Scene Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1

[14] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. UPSNet: A Unified Panoptic Segmentation Network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 3

[15] 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 preprint arXiv:1902.05093*, 2019. 3