Instance Segmentation of Urban Street Scenes

Francesco Bari
francesco.bari.2@studenti.unipd.it

Eleonora Signor
eleonora.signor@studenti.unipd.it

Abstract

In this report we compared different existing instance segmentation techniques on the specific task of Urban Street Scenes. Our interest in the topic was born out of the fact that the segmentation of instances is one of the fundamental tasks of the vision but at the same time it is still complex and not fully explored.

1. Introduction

Image segmentation is the process of separating an image into several segments in which each pixel is associated with an object type. There are two types of image segmentation: semantic segmentation and instance segmentation. The first marks objects of the same type with the equal class label, the second marks objects of the same type belonging to distinct entities with different class labels. The idea we tried to develop was to compare different instance segmentation methods. The first technique we studied was Mask R-CNN [1], a two-stage approach. We chose this one because of the positive feedback received from the world of vision research, due to its conceptually simple and general framework characterised by efficient image object detection and the simultaneous generation of a high-quality segmentation mask for each instance. The technique we decided to contrast with Mask R-CNN was BlendMask [2]. This is a one-stage technique that has been shown to outperform Mask R-CNN both in terms of mask prediction and training time on the MSCOCO 2017 [3] and LVIS [4] datasets. We were interested in verifying whether this also applied to datasets such as *Cityscapes* [5] and *WildDash* [6] belonging to the specific topic of *Urban Street Scenes*, featuring images from the streets around the world with many difficult scenarios. Some of the aspects we tested involved backbone changes, depth of the ResNet [7] and the number of frozen layers. The results obtained confirmed what had already been announced in previous works, generalising BlendMask as one of the most promising staged approaches. At the end of our work, in order not to limit our analysis, we also made some considerations on other instance segmentation techniques such as SOLOv2 [8] and Deep Snake [9].

2. Related Work

The approach we used in our work was to use the following papers as guidelines [1, 2, 8, 9] extending the analysis also to datasets of *Urban Street Scenes*.

2.1. Stage approch

Mask R-CNN [1] is a box-based two-stage approach with a Convolutional Neural Network, avant-garde in the task of image segmentation. It is a variant of a Deep Neural Network that detects objects in an image and generates a segmentation mask for each instance. Mask R-CNN is the next evolution of Faster R-CNN [10], a Region-based Convolutional Neural Network which produces for each candidate object 3 outputs: the class label, the bounding box offset and the object mask. The architecture of the network, Figure 1, consists of a CNN (backbone) which processes the image and extracts the feature map. After that, thanks to the *Region Proposal Network*, proposals or RoI are presented on which to make recognition of the bounding box and mask prediction (head). In addition, before generating the output, *RoIAlign* is applied to each RoI which makes it possible to obtain a mask where the layout of the object is maintained.

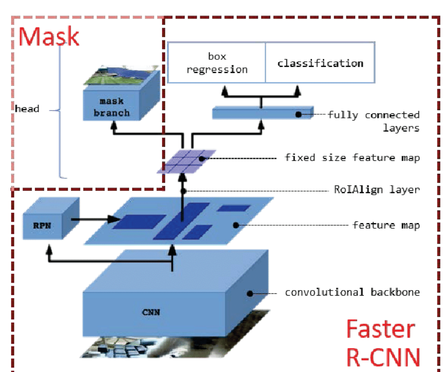

Figure 1. Mask R-CNN architecture. *Source: [11], pag. 3.* The convolutional *backbones* (in the lower part of the figure) proposed in the paper are C4, C5 and FPN [12]. For *heads* (in the upper part of the figure) the backbone must include the 5th stage of ResNet [7], "res5".

BlendMask [2] is derived from the limits of Mask R-CNN. The authors of the paper define how Mask R-CNN strongly constrains the speed and quality of mask generation at heads, thus making it difficult to deal with complicated scenarios and placing a limit on masks resolution. Furthermore, Mask R-CNN presents itself as an inflexible framework for multi-task networks. They thus attempted to combine top-down and bottom-up search strategies in FCOS [13] and anchor box-free one-stage approach which seems able to outperform its two-stage counterparts in terms of accuracy. The architecture of BlendMask, Figure 2, consists of a detector network and a mask branch. The latter is partitioned into 3 parts: the bottom module that deals with predicting scores maps, called bases; the top layer is composed of a single convolution layer, one for each tower as many as the input features with the task of predicting attention instances; and a blender module that combines scores with attentions.

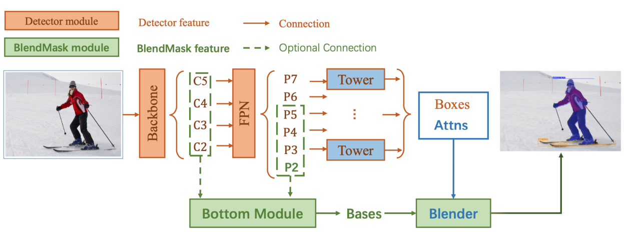

Figure 2. BlendMask architecture. *Source: [2], pag. 3.* The *bottom module* (to left) uses C4/C5 or FPN [12] and produces the bases. In the *top layer* (to right) a single convolution layer is added above the towers and this allows attention masks to be produced. The *blender module* (lower part of the figure), for each instance, combining bases linearly with the learned attention maps.

SOLOv2 [8] is a box-free one-stage approach, a successor of SOLO [14]. In this case each instance of an image is segmented dynamically without detection of the bounding box. The mask generation, unlike Mask R-CNN, is decoupled into mask kernel prediction and mask feature learning. These two elements are responsible for generating convolution kernels and feature maps. SOLOv2 also manages to achieve promising results through the use of the matrix *non-maximum suppression* (NMS) technique proposed by the authors of the paper which reduces duplicate predictions while gaining less inference overhead.

2.2. Contour-based approach

Deep Snake [9] is a contour-based approach implementing the idea of snake algorithms with a learning-based approach. Deep Snake consists of a two-stage pipeline: in a first instance there is an initial contour proposal on the object of an image followed by the use of a Neural Network which iteratively deforms this proposal until it exactly matches the object's boundaries. For learning the structure of contour features the authors of the paper propose *Circular Convolution*.

3. Datasets

The datasets of *Urban Street Scenes* which we used have been *Cityscapes* [5] and *WildDash* [6].

3.1. Cityscapes

Cityscapes [5] is a suite of benchmarks and a large-scale dataset for semantic urban scene understanding. It is suitable for learning and testing pixel-level and instance-level semantic labelling methods. The images of *Cityscapes* were created from a large and diverse set of video sequences, recorded in 50 different cities. These may be inclusive of high-quality annotations, or/and coarse annotations. The latter allows testing of methods employing large volumes of weakly labelled data. The annotations contained are of a polygonal type and they are fundamental for the evaluation of the model. The dataset we used is partitioned into two units: *gtFine* and *leftImg8*, Figure 3, which we used in pairs. *gtFine* consists of fine annotations for 3475 train and val images and 1525 test set images. *leftImg8* composed with "raw" images of urban traffic split in train, test and val set with a total of 5000 images.

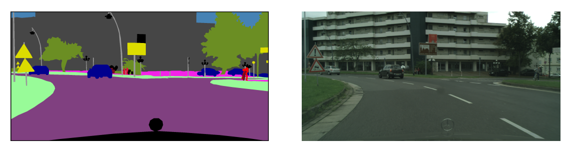

Figure 3. *Cityscapes* [5] images: *gtFine* to left and *leftImg8* to right.

In Figure 4 we report the classes and the number of occurrences in *gtFine train* and *leftImg8 train*. The total number of classes is 8.

category	#instances	category	#instances	category	#instances
person	17918	rider	1781	car	26963
truck	484	bus	380	train	168
motorcycle	737	bicycle	3675		
total	52106				

Figure 4. *Cityscapes*: definitions of train dataset classes.

Although the datasets include several months and seasons, they are always images taken in good weather conditions. This aspect prompted us to analyse the behaviour of our instance segmentation techniques on the *WildDash* [6] dataset.

3.2. WildDash

WildDash [6] is a benchmark suite and dataset for semantic and instance segmentation for the automotive domain. The images contained in the datasets come from different sources from all over the world. In addition they present scenarios such as rain, darkness and road cover which are real challenges for image recognition. This highlights the shortcomings of any instance segmentation technique. The dataset we used is *public gt package*, Figure 5, consisting of 4256 images aimed specifically at solving instance segmentation tasks. However was not divided into train, val and test set. Consequently we decided to divide it up manually reserving 3405 images as train set and 851 images as test set. We did not consider it necessary to make a further partition in val set to keep as many images as possible in the training and assuming that our models that use weights already pre-trained on *ImageNet* [15] were accurate enough not to require *Model Selection*.

Figure 5. *WildDash* [6] images: *rain scenario* to left and *road in the desert* to right.

In Figure 6 we report the classes and the number of occurrences in *public gt package train*. The total number of classes is 13.

category	#instances	category	#instances	category	#instances
ego vehicle	504	person	1479	rider	589
car	4045	truck	439	bus	128
caravan	16	trailer	20	train	6
motorcycle	536	bicycle	74	pickup	266
van	201				
total	8303				

Figure 6. *WildDash*: definitions of train dataset classes.

4. Method

To be able to operationally compare the instance segmentation techniques that are the subject of our work, we have referred to already existing libraries developed by the researchers of [1, 2]. These are *Detectron2* for Mask R-CNN [1] and *AdelaiDet* for BlendMask [2]. These libraries have allowed us to define *ideal models* found during the experiments. Furthermore making explicit the evaluation set and the metrics needed to estimate performance allowed us to concretise the concept of comparison.

4.1. Architecture

We decided to use ResNet-101-FPN as the backbone in both Mask R-CNN [1] and BlendMask [2]. In addition, as the bottom module of BlendMask, we have instantiated ProtoNet [18].

4.1.1 ResNet 101

Both *Detectron2* and *AdelaiDet* offer ResNet [7] as a backbone network in which it is possible to choose between 50 and 101 layers. In addition *Detectron2* also includes ResNeXt-101 [17], a cardinal variant of ResNet 101 that we have excluded from any possible comparison a priori to the benefit of the relationship resources at our disposal - forecasting accuracy - computation effort. This because ResNeXt, as can be seen [1, 17], has a longer inference and training time than ResNet 101. We considered ResNet 101, Figure 7, sufficient for our purposes, defined as a network that also converges very well from the reference paper [7].

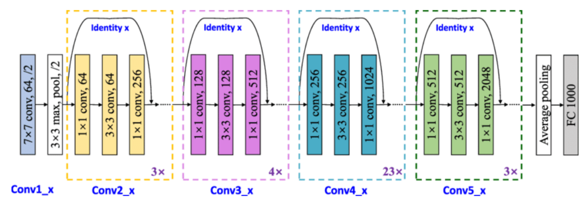

Figure 7. The structure of the ResNet 101. *Source: [16], pag. 6.* The figure shows ResNet 101. It uses skip connections i.e. arcs over each block to solve the problems associated with gradient computation.

4.1.2 Feature Pyramid Network

For complete the backbone ResNet [7] it is not enough. Is necessary a second architecture that extracts the features in Mask R-CNN [1] or the basis for BlendMask [2]. In this case, *Detectron2* includes several already configured approaches, e.g. C4 where features are extracted from the 4th convolutional stage of ResNet, or FPN [12] with lateral connections that allow building a pyramid of functionalities within the network from single-scale input. *AdelaiDet* on the other hand exclusively implements FPN. Our choice fell on FPN in order to maintain consistency between the two models under analysis and to be once again in line with what was defined by the papers [1, 2]. The researchers declare as FPN is performant in both features/bases extraction and execution time. In Figure 8 we show how FPN solves the feature extraction when it is used as a backbone in an instance segmentation task.

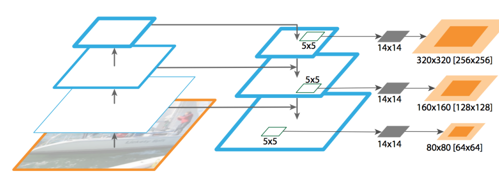

Figure 8. FPN for object segment proposals. *Source: [12], pag. 8.* Each level of the pyramid tries to detect all instances present in an image. A features map/bases are produced for each of these layers and each will produce a mask. The latter will be combined into a single final mask.

4.1.3 ProtoNet decoder

In *AdelaiDet* there are one type of bottom modules that receive input from the backbone: ProtoNet [18], Figure 9. ProtoNet is a prototype generation branch, i.e. it predicts a set of k prototypes masks for the whole image. It is implemented as a Fully Connected Network where the last layer of which has as many channels as there are prototypes. We have decided to remain in line with this choice for our model.

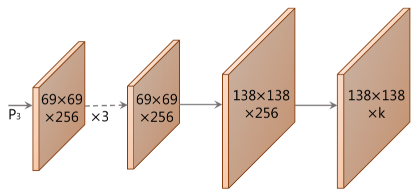

Figure 9. ProtoNet architecture. *Source: [18], pag. 4.* The labels denote feature size and channels for an image size of 550 × 550. Arrows indicate 3 × 3 conv layers except for the final conv which is 1 × 1. The increase in size is an upsample followed by a conv. As can be seen the last layer has as many channels as k prototypes.

4.2. Hyperparameters configuration

For the choice of models hyperparameters we tried to keep the default ones as defined in the libraries *Detectron2* and *AdelaiDet* which are the same as those defined in the papers [1, 2]. We only modified those that did not allow us to run on Colab, namely the number of iterations to 4000 for Mask R-CNN and 2000 for BlendMask and the number of batch sizes per image to 128.

4.3. Loss function

The loss we used during the training of Mask R-CNN, defined in [1], is as follows:

$$L = L_{cls} + L_{box} + L_{mask} \tag{1}$$

Where:

- L_{cls} is the classification loss;
- L_{box} is the bounding-box loss;
- L_{mask} is the average binary *cross-entropy* loss. The mask branch has Km^2 dimensional output for each RoI where K is cardinality binary mask with resolution $m \times m$, one for each of the K classes. L_{mask} is defined only on k-th mask. This allows the network to generate masks for every class without competition among classes.

Concerning *BlendMask* we decided to refer to the semantic loss [19] as also done in part of the experiments in [2]:

$$L^s(\alpha, p) \propto -log \sum_{x \models \alpha} \prod_{i:x \models X_i} p_i \prod_{i:x \models \neg X_i} (1 - p_i) \tag{2}$$

Where:

- α is a sentence in propositional logic defined on variables $X_1, ..X_n$;
- p is a probability vector for each variable X_i;
- $L^s(\alpha, p)$ is the semantic loss between α and p.

Intuitively the semantic loss is proportional to the logarithm of generating a state that satisfies the constraint when values according to p are sampled. This loss function captures how close the Neural Network is to satisfying the constraints on its output. We decided to add the semantic loss to the loss defined in 1, driven by the promising results of the paper [2] and because it was defined by its authors as useful to achieve state-of-the-art results on semi-supervised multi-class classification.
The objective we tried to pursue both during the definition of the *ideal models* and during the experiments was the identification of a trade-off between the minimisation of the loss function, the training time and the accuracy box/mask.

4.4. Evalutation

As far as the evaluation is concerned we had to settle for the val test dataset of *Cityscapes*. This is because one of the choices made by the creators of *Cityscapes* [5] was to make the annotations for the test set not public and this make impossible to perform any calculations and evaluation on masks and boxes. Instead, as already reported in the section §3.2, for the *WildDash* dataset we used the test set partitioned from the train set at 80%-20%.

4.4.1 Metrics

As a metric for evaluating the results obtained from our models we decided to use *Average Precision* (AP). The AP calculates the average accuracy value for the recall value from 0 to 1. In formula that means:

$$\int_0^1 p(r)dr \tag{3}$$

Where $p(r)$ is the area under the curve of maximum intersection between *Precision* and *Recall*:

- *Recall* measures "how well" all positives are found

$$\frac{\text{true positive}}{\text{true positive + false negative}} \tag{4}$$

- *Precision* measures the accuracy of predictions

$$\frac{\text{true positive}}{\text{true positive + false positive}} \tag{5}$$

For our evaluations we decided to consider the AP as is defined metric by COCO, i.e. the AP is averaged over several *Intersection over Union* (IoU) values. Specifically 10 IoU thresholds from the range .50:.05:.95 are used and from these are extracted the 100 detections with the highest score.

5. Experiments

In this section we present the most significant experiments that led us to define the *ideal models*.

5.1. Backbone

The first set of experiments we carried out on the backbone, an element present in both Mask R-CNN and BlendMask although different stage architectures. We have decided to test depths and feature extractor.

Architecture	*Cityscapes*	*WildDash*
Mask R-CNN: ResNet-50-C4	box AP: 25.935 mask AP: 20.005	box AP: 18.730 mask AP: 17.589
Mask R-CNN: ResNet-50-DC5	box AP: 25.591 mask AP: 20.285	box AP: 17.871 mask AP: 16.543
Mask R-CNN: ResNet-50-FPN	box AP: 24.437 mask AP: 20.327	box AP: 16.865 mask AP: 16.017
Mask R-CNN: ResNet-101-C4	box AP: 30.888 mask AP: 24.562	box AP: 20.459 mask AP: 19.331
Mask R-CNN: ResNet-101-DC5	box AP: 29.125 mask AP: 24.062	box AP: 19.391 mask AP: 18.529
Mask R-CNN: ResNet-101-FPN	**box AP: 28.563 mask AP: 24.676**	**box AP: 19.024 mask AP: 17.994**

Table 1. Experiments on *backbone* Mask R-CNN.

The feature extractors of Mask R-CNN that we decided to test were C4, DC5 and FPN. From the results shown in Table 1 appears that the best performing is the ResNet-101-C4 architecture. However this runs counter to what is defined in Mask R-CNN paper [1] which shows that 4-stage features extraction is less accurate than the bottom-up and top-down approach of FPN [12]. We explained this inconsistency by the computational limitations of Colab which constrained us to work with only 4000 iterations. Probably this small number did not allow us to grasp the real potential and limitations of each feature tested. For this reason we have decided to consider ResNet-101-FPN as the best performing architecture for Mask R-CNN.

Architecture	*Cityscapes*	*WildDash*
BlendMask: ResNet-50-FPN	box AP: 30.129 mask AP: 25.325	box AP: 20.225 mask AP: 19.810
BlendMask: ResNet-101-FPN	**box AP: 31.017 mask AP: 26.411**	**box AP: 20.347 mask AP: 20.162**
BlendMask: ResNet-101-FPN + dcni	box AP: 32.196 mask AP: 27.212	box AP: 20.497 mask AP: 20.236

Table 2. Experiments on *backbone* BlendMask.

For BlendMask the features we have considered have always been extracted with FPN. However we thought it interesting to also test a ResNet backbone with interval 3 convolution (dcni). As can be seen from the results which we have reported in Table 2 the application of deformation achieves even better results. In this case the low number of iterations allowed us to obtain sufficiently clear results, probably because the feature extractor we tested was always FPN. We still decided to continue the BlendMask experiments with ResNet-101-FPN for uniformity of comparison with Mask R-CNN. In conclusion we can say that even if the backbones are integrated into different architectures the expected behaviours are similar. In fact both in Mask R-CNN and BlendMask as the depth of the network increases, the accuracy of the network increases as well. Moreover, even if not visible in our experiments, FPN in both methods performs a very good features extraction.

5.2. Frozen layers

Another set of experiments we performed were on the number of frozen layers from the ResNet during the training. The choice of experimenting with the effects of fine-tuning was made to understand how many convolutional layers are worth training when already working with pre-trained models. For the experiments on the number of frozen layers we have decided to omit the single layer case due to the size of our individual datasets. In this scenario, in fact, the advantages of using a pre-trained network would disappear.

Architecture	*Cityscapes*	*WildDash*
Mask R-CNN: ResNet-101-FPN + freeze at 2nd	**box AP: 28.563 mask AP: 24.676**	**box AP: 19.024 mask AP: 17.994**
Mask R-CNN: ResNet-101-FPN + freeze at 3rd	box AP: 26.297 mask AP: 22.179	box AP: 19.132 mask AP: 18.224
Mask R-CNN: ResNet-101-FPN + freeze at 4th	box AP: 25.053 mask AP: 21.349	box AP: 19.824 mask AP: 18.448

Table 3. Experiments on *frozen layers* Mask R-CNN.

Architecture	*Cityscapes*	*WildDash*
BlendMask: ResNet-101-FPN + freeze at 2nd	**box AP: 31.017 mask AP: 26.411**	**box AP: 20.347 mask AP: 20.162**
BlendMask: ResNet-101-FPN + freeze at 3rd	box AP: 28.596 mask AP: 23.649	box AP: 18.993 mask AP: 17.802
BlendMask: ResNet-101-FPN + freeze at 4th	box AP: 31.379 mask AP: 26.936	box AP: 21.597 mask AP: 20.537

Table 4. Experiments on *frozen layers* BlendMask.

From Table 4 it appears that BlendMask frozen at 4th seems to be the most performing architecture. This is not so evident for Mask R-CNN, Table 3. These inconsistencies, once again, we related to the low number of iterations. As a consequence, we have guessed that there is a behaviour that seems to be more promising, freezing 4th layer (which may depend on the large size of conv4). However we could not obtain indisputable results to prove this phenomenon. Therefore a good choice when working with fine-tuning is to freeze the first two convolutional stages of the networks.

5.3. Ideal models

At the end of our experiments we selected the following *ideal models* to solve the *Urban Street View* task:

Method	*Cityscapes*	*WildDash*
Mask R-CNN:	box AP: 28.563 mask AP: 24.676 train-time: 1:08:28 infer-time: 0:05:13	box AP: 19.024 mask AP: 17.994 train-time: 1:13:20 infer-time: 0:08:03
BlendMask:	box AP: 31.017 (+2.45 AP) mask AP: 26.411 (+1.74 AP, +1.4 AP [2]) train-time: 0:43:01 (−37%, −20% [2]) infer-time: 0:12:05 (+131%, −80% [2])	box AP: 20.347 (+1.32 AP) mask AP: 20.162 (+2.17 AP, +1.4 AP [2]) train- time: 0:47:19 (−36%, −20% [2]) infer-time: 0:15:45 (+96%, −80% [2])

Table 5. Results *ideal models*: the architecture is ResNet-101-FPN with freeze at 2nd. In brackets are the improvements of *BlendMask ideal model* compared to *Mask R-CNN ideal model* and follow the improvements in the reference paper [2] when exist.

Figure 10. *Models ideal results*: Mask image (to left) and Blend image (to right). 1st and 2nd row is *Cityscapes* dataset and 3rd and 4th row is *WildDash* dataset. In the figure we see that Mask R-CNN tends to have a higher instance recognition score than BlendMask despite having a lower AP. This is because it is more confident than BlendMask but this also leads it to be more penalised.

As confirmed also by Table 5 BlendMask is more accurate than Mask R-CNN and faster in training time. However the benefits obtained by our *ideal model* are significantly lower than those expressed in [2]. We attribute this to the fact that we used 1 single GPU and only 2000 iterations for our ablations. In Figure 10 we report some inference tests we obtained from ours *ideal models*.

6. Conclusion

From our experiments we can say that one-stage and anchor box-free techniques, suitably modified, perform better than box-based two-stage methods (of which king is Mask R-CNN [1]) least in terms of *Average Precision* and time. These positive effects are probably caused by the hybridisation of top-down and bottom-up methods and the detection of unanchored objects (as is the case for BlendMask [2]). In addition, recent studies reported in [8] and partly shown in Figure 11 prove that a box-free stage approach combined with matrix NMS, such as SOLOv2 [8], is demonstrated to be very competitive against BlendMask.

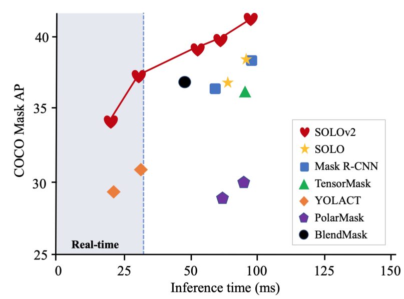

Figure 11. Comparison between SOLOv2 and others stage approaches. *Source: [8], pag. 2.* Mask R-CNN is box-based two-stage approach, the others are box-free or anchor box-free one-stage approach.

Moreover, stage approaches are not the only possible methods for solving instance segmentation tasks. For example there is Deep Snake [9] a counter-based approach that outperforms Mask R-CNN both in terms of inference speed and *Average Precision*, as shown by table in Figure 12. It has the potential to be a good competitor to SOLOv2.

	training data	fps	AP [val]	AP	AP_{50}
SGN [26]	fine + coarse	0.6	29.2	25.0	44.9
PolygonRNN++ [1]	fine	-	-	25.5	45.5
Mask R-CNN [18]	fine	2.2	31.5	26.2	49.9
GMIS [28]	fine + coarse	-	-	27.6	49.6
Spatial [31]	fine	11	-	27.6	50.9
PANet [27]	fine	<1	36.5	**31.8**	57.1
Deep snake	fine	4.6	**37.4**	31.7	**58.4**

Figure 12. Results on *Cityscapes* val (AP [val] column) and test (remaining columns) sets. *Source: [9], pag. 7.*

A possible future extension, in line with the experiments and considerations we have made, could consist in search to further improve the performance of BlendMask by trying to use Deep Snake contour-based approach for detecting object bounding boxes of an image.

References

[1] Kaiming He and Georgia Gkioxari and Piotr Dollár and Ross Girshick. Mask R-CNN. CoRR, 2018.

[2] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang and Youliang Yan. BlendMask: Top-Down Meets Bottom-Up for Instance Segmentation. CoRR, 2020.

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

[4] Agrim Gupta, Piotr Dollár and Ross B. Girshick. LVIS: A Dataset for Large Vocabulary Instance Segmentation. CoRR, 2019.

[5] 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. CoRR, 2016.

[6] Zendel, Oliver and Honauer, Katrin and Murschitz, Markus and Steininger, Daniel and Dominguez, Gustavo Fernandez. WildDash - Creating Hazard-Aware Benchmarks. Proceedings of the European Conference on Computer Vision, (ECCV), 2018.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition. CoRR, 2015.

[8] Xinlong Wang,Rufeng Zhang,Tao Kong, Lei Li and Chunhua Shen. SOLOv2: Dynamic, Faster and Stronger. CoRR, 2020.

[9] Sida Peng, Wen Jiang, Huaijin Pi, Hujun Bao and Xiaowei Zhou. Deep Snake for Real-Time Instance Segmentation. CoRR, 2020.

[10] Shaoqing Ren, Kaiming He, Ross B. Girshick, Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. CoRR, 2015.

[11] Bienias, Lukasz & n, Juanjo & Nielsen, Line & Alstrøm, Tommy. Insights Into The Behaviour Of Multi-Task Deep Neural Networks For Medical Image Segmentation. 2019.

[12] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan and Serge J. Belongie. Feature Pyramid Networks for Object Detection. CoRR, 2016.

[13] Zhi Tian, Chunhua Shen, Hao Chen and Tong He. FCOS: Fully Convolutional One-Stage Object Detection. CoRR, 2019.

[14] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang and Lei Li SOLO: Segmenting Objects by Locations. CoRR, 2019.

[15] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei. ”ImageNet: A large-scale hierarchical image database,” 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[16] Chen, Jiayao & Zhou, Mingliang & Zhang, Dongming & Huang, H. & Zhang, Fengshou. (2021). Quantification of water inflow in rock tunnel faces via convolutional neural network approach. Automation in Construction. 123. 103526. 10.1016/j.autcon.2020.103526.

[17] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks. CoRR, 2016.

[18] Daniel Bolya, Chong Zhou, Fanyi Xiao and Yong Jae Lee. YOLACT: Real-time Instance Segmentation. CoRR, 2019.

[19] Xu, Jingyi and Zhang, Zilu and Friedman, Tal and Liang, Yitao and Van den Broeck, Guy. A Semantic Loss Function for Deep Learning with Symbolic Knowledge. Proceedings of the 35th International Conference on Machine Learning, 2018.

[20] X. Yin, W. Chen, X. Wu and H. Yue, ”Fine-tuning and visualization of convolutional neural networks,” 2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2017, pp. 1310-1315, doi: 10.1109/ICIEA.2017.8283041.

[21] Xinlei Chen, Ross B. Girshick, Kaiming He and Piotr Dollár. TensorMask: A Foundation for Dense Object Segmentation. CoRR, 2019.

[22] Daniel Bolya, Chong Zhou, Fanyi Xiao and Yong Jae Lee. YOLACT: Real-time Instance Segmentation. CoRR, 2019.

[23] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen and Ping Luo. PolarMask: Single Shot Instance Segmentation with Polar Representation. CoRR, 2019.