Instance Segmentation for 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, however 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 and 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 in the light of the positive feedback it has 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]. Instead, 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 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 and 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 approch with a Convolutional Neural Network, that is at the avant-garde 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 ouput, *RoIAlign* is applied to each RoI, which makes it possible to obtain a mask, where the layout of the object is maintained.

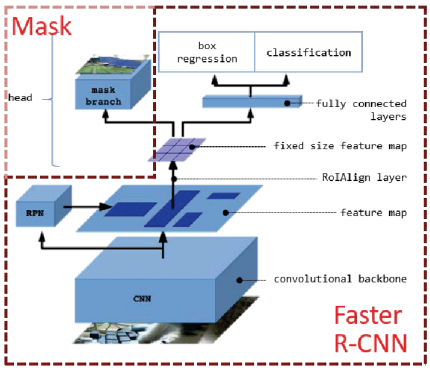

Figure 1. Mask R-CNN architecture. *Source: [11], pag. 3.* The convolutional *backbones* (in the lower part of the figure) proposed in the paper are C4, C5 and FPN [12]. For *heads* (in the upper part of the figure) the backbone must include the 5-th stage of ResNet [7], "res5".

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 cobine top-down and bottom-up search strategies in FCOS [13], 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 in 3 parts: the bottom module that deals with predicting scores maps, called bases; the top layer composed of a single convolution layer and towers, as many as the input features, with the task of predicting attention instances; and a blender module that combines scores with attentions.

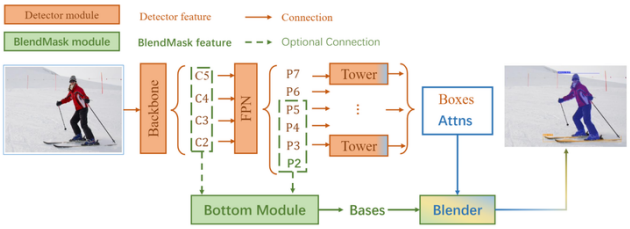

Figure 2. BlendMask architecture. *Source: [2], pag. 3.*
The *bottom module* (to left) uses C4/C5 or FPN [12] and produces the bases. In the *top layer* (to right) a single convolution layer is added above the towers and this allows attention masks to be produced. The *blender module* (lower part of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

SOLOv2 [8] is 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 the use of *Circular Convolution*.

3. Datasets

The datasets of *Urban Street Scenes*, which we used, belong to *Cityscapes* [5] and *WildDash* [6].

3.1. Cityscapes

Cityscapes [5] is a suite of benchmark 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 allow the testing of methods employing large volumes of weakly labelled data. The annotations contained, which are fundamental for the evaluation of a model, are of a polygonal type. The dataset we used, belonging to *Cityscapes*, is partitioned into two units: *gtFine* and *leftImg8*, which we used in pairs. *gtFine* consists of fine annotations for 3 475 train and val images, and 1 525 test set images. *leftImg8* from "row" images of urban traffic, with train set, test set and val set; a total of 5 000 images. In Figure 3, 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 3. *Cityscapes*: definitions of train dataset classes.

However of *Cityscapes* 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* consisting of 4 256 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 3 405 images as train set and 851 images as test set. We did not consider it necessary to make a further partition in val set, in order to keep as many images as possible

in the training; and assuming that our models, using weights already pretrained on *ImageNet* [15], were accurate enough not to require *Model Selection*. In Figure 4, 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	1526	person	7943	rider	1626
car	14440	truck	1807	bus	825
caravan	46	trailer	65	train	74
motorcycle	1555	bicycle	536	pickup	531
van	928				
total	31902				

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

4. Method

In order 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 sub-optimal ideal models, in order to improve their accuracy during 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 [19].

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 [17] 101, a cardinal variant of ResNet 101; which, however, we have excluded from any possible comparison because of the excessive training time, for the means at our disposal. We considered ResNet 101, Figure 5, sufficient for our purposes, defined as a network that also converges very well from the reference paper [7].

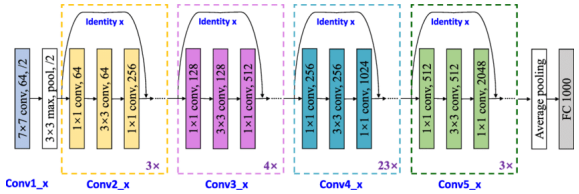

Figure 5. The structure of the ResNet 101. *Source: [16], pag. 6.* The figure shows Resnet 101. It uses skip connections, i.e. arcs over each block, to solve the problems associated with gradient computation.

4.1.2 Feature Pyramid Network

For complete the backbone, ResNet [7] it is not enough. Is necessary a second architecture that extract 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 4-th convolutional stage of ResNet; or FPN [12] with lateral connections that allow to build 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]. In fact, the researchers declare as FPN is performant in both features/bases extraction and execution time. In Figure 6 we show how FPN solves the feature extraction, when it is used as backbone in an instance segmentation task.

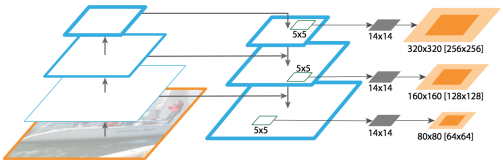

Figure 6. FPN for object segment proposals. *Source: [12], pag. 8.* The feature pyramid is constructed with identical structure as for object detection. Each level of the pyramid tries to detect all instances present in an image. A features map/bases is produced for each of these layers, and each will produce a mask. These latter will be combined into a single final mask.

4.1.3 ProtoNet decoder

In *AdelaiDet* there are two types of bottom modules, which receive input from the backbone: ProtoNet [19] and DeepLabv3+ [18]. 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. DeepLabv3+ is an encoder-decoder structure, that is able to control the resolution of feature extracts thanks to the *atrous convolution*, which has the effect of increasing the kernel's field of view. In the default configuration of *AdelaiDet*, ProtoNet is defined as a module. We have decided to remain in line with this choice for our sub-optimal model.

4.2. Configuration hyperparameters

Hyperparameters of configurationHyperparameters of configurationHyperparameters of configurationHyperparameters of configurationHyperparameters of configurationHyperparameters of configurationHyperparameters of configurationHyperparameters of configurationHyper-

parameters of configurationHyperparameters of configuration

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 resolutiom 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.

With regard to *BlendMask*, we decided to refer to the semantic loss [20], 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] not to make the annotations for the test set public, and this makes it 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, which we 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 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.

4.4.2 Sub-optimal ideal models results

Figure 7. *Models ideal results*: mask image (to left) and blend image (to right). First row is *Cityscapes* dataset and second row is *WildDash* dataset.

Method	*Cityscapes*	*WildDash*
Mask R-CNN	box AP: 28.278 mask AP: 23.793 training time:0:26:43	box AP: 18.815 mask AP: 17.738 training time: 1:14:57
BlendMask		

Table 1. AP and training time ideal models.

5. Experiments

In questa sezione descriviamo gli esperimenti che abbiamo eseguito per testare e valutare le tecniche oggetto di questo lavoro. Tali esperimenti gli abbiamo eseguiti al termine delle fasi di studio e codifica.

5.1. Backbone

La seconda serie di esperimenti, che abbiamo compiuto, riguarda la definizione della backbone. Tutte le tecniche a stadi, oggetto del confronto, sono dotate del suddetto modulo inferiore, per cui ci è risultato semplice uniformare le scelte architetturali in modo da poter compiere una valutazione oggettiva. Le tecniche che abbiamo confrontato sono state Mask R-CNN e BlendMask.
Le configurazioni constanti delle reti sono image size ..., numero massimo di iterazioni, learning rate ..., step size a ... e fine-tuning esclusivamente agli ultimi 2 livelli.

Method and architecture	*Cityscapes* AP	*WildDash* AP
Mask R-CNN + ResNet50 + C4 + Base-RCNN-C4		
Mask R-CNN + ResNet50 + DC5 + Base-RCNN-DilatedC5		
Mask R-CNN + ResNet50 + FPN + Base-RCNN-FPN		

Table 2. Backbone Mask R-CNN result.

Per BlendMask, oltre a settare le configurazioni costanti, avvalendoci dei risultati presentati in ... abbiamo settato R = 56, M = 14, K = 4, sampling method for bottom bases bilinear pooling, interpolation method for top-level attentions bilinear upsampling and semantic loss. Inoltre abbiamo deciso di testare vari tipi di decoder: ProtoNet and DeepLabv3+.

Method and architecture	*Cityscapes* AP	*WildDash* AP
BlendMask with decoder ProtoNet + ResNet50 + FPN + Base-550		
BlendMask with decoder ProtoNet + ResNet50 + deformable convolution + FPN + Base-550		
BlendMask with decoder DeepLabv3+ + ResNet50 + FPN + Base-550		
BlendMask with decoder DeepLabv3+ + ResNet50 + deformable convolution + FPN + Base-550		

Table 3. Backbone BlendMask result.

5.2. Deepness

Una terza serie di esperimenti ha riguardato lo studio della profondità delle reti ResNet.
I parametri di configurazione non definiti in modo esplicito, sono le medesime di quelle riportate nella sezione §5.1.

Method and architecture	*Cityscapes* AP	*WildDash* AP
BlendMask with decoder ProtoNet + ResNet101 + FPN + Base-BlendMask		
BlendMask with decoder ProtoNet + ResNet101 + deformable convolution + FPN + Base-BlendMask		

Table 4. Deepness BlendMask result.

5.3. Freeze levels

Per la quarta serie di esperimenti ci siamo voluti concentrare sul numero di layers da "scongelare" di ResNet durante il re-training dei pesi.
I parametri di configurazione non definiti in modo esplicito, sono le medesime di quelle riportate nella sezione §5.1.

Method and architecture	*Cityscapes* AP	*WildDash* AP
Mask R-CNN + ResNet101 + FPN 1 layers freeze		
Mask R-CNN + ResNet101 + FPN 3 layers freeze		
BlendMask with decoder ProtoNet + ResNet101 + FPN + Base-BlendMask 1 layers freeze		
BlendMask with decoder ProtoNet + ResNet101 + FPN + Base-BlendMask 3 layers freeze		

Table 5. Freeze layers result.

5.4. Own best models

Come ultima serie di esperimenti abbiamo cercato di individuare i modelli migliori, per ciascuna le due tecniche di instance segmentation in esame in questa sezione; tenedo conto della possibilità di allenare ciascun modello solo su una singolo macchina e 1 GPU.

I parametri di configurazione non definiti in modo esplicito, sono le medesime di quelle riportate nella sezione §5.1.

Dataset	method and architecture	AP

Table 6. Own best models result.

6. Conclusion

From our experiments we are able to 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]), both in terms of AP precision and GPU 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 8, prove that a box-free stage approach combined with matrix NMS, such as SOLOv2 [8], is demonstrated to be very competitive against BlendMask [2].

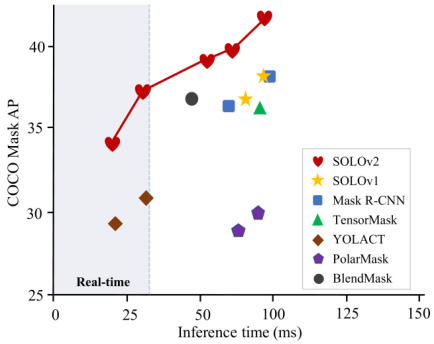

Figure 8. Comparison between SOLOv2 and other stage approches. *Source: [8], pag. 2.*
From top to bottom: SOLOv2[8] and SOLO[14] are box-free one-stage approch; Mask R-CNN is box-based two-stage approch; TensorMask [21] is dense sliding-window one-stage approch; YOLACT [22] is one-stage approch predecessor of *BlendMask* [2]; PolarMask [23] anchor box-free, single shot instance segmentation one-stage approch; BlendMask [2] anchor box-free one-stage approch.

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 [1] both in terms of inference speed and AP precision, as shown by the table in Figure 9, and has the potential to be a good competitor to SOLOv2 [8].

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

A possible future extension, in line with the results and considerations we have made, could consist in search to further improve the performance of BlendMask [2] by trying to use Deep Snake [9] contour-based approach for detecting object bounding boxes of a 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] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff and Hartwig Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. CoRR, 2018.

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

[20] 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.

[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.