Object detection and Instance Segmentation.

Marcos Conde, Alex Martin, José Manuel López
UPC - UAB. M5-Visual Recognition, Group 6
{ marcos.conde , alex.martin }@uab.cat

Abstract

Object Detection and Instance Segmentation are two of the most challenging tasks in Computer Vision, for this reason, multiple approaches have been proposed in the past years with fast and reliable performance. This work explores the state-of-the-art models, frameworks and applications for these tasks, as well as their difficulties. In particular, we will focus on the Detectron2 framework.

1. Introduction

The three main computer vision tasks well-described and studied in the literature are:

1. **Classification**: for a given image predict to one of a set of predefined categories or classes the image belongs. There might be 1 class (binary classification) or multiple classes (multi-class).

2. **Detection**: this task aims to find the location of a single (or multiple) objects in an image, this task is an extension of the classification. We provide the localization of the object using coordinates related to the image, usually this coordinates conform a box called "bounding box".

3. **Segmentation**: is an extension of the detection task, in this case, we want to know where the objects are and also its particular area, not a wide bounding box. These models are more complex to train and integrate.

By definition, the most important and common task is image classification. All these task have a common approach, Convolutional Neural Networks (CNNs) [6, 7].

In this work, we introduce the main computer vision techniques used to build a Visual Perception system focusing on: detection and instance segmentation of multiple objects in the scene.

2. Related work

The well-known benchmark and dataset to train and test models is MS-COCO [8] which contains annotations for detection and segmentation of common objects in our life (i.e. table, seat, cars). The KITTI Dataset [3] and Benchmark serves as baseline for object detection and segmentation in the autonomous driving field, it contains annotations of real driving scenarios, the purpose of this suite is to develop new models that improve the self-driving car perception layer.

For object detection, the most widely used state-of-the-art models are the R-CNN family: Faster R-CNN [10] and Mask RCNN [5]. These networks usually consist of: (i) A region proposal algorithm to generate "bounding boxes" or locations of possible objects in the image; (ii) A feature generation stage to obtain features of these objects, usually using a CNN; (iii) A classification layer (a network from the previous section) to predict which class this object belongs to; and (iv) A regression layer to make the coordinates of the object bounding box more precise.

The only stand-alone portion of the network left in Fast R-CNN was the region proposal algorithm. The Faster R-CNN [10] paper fixes this by using another convolutional network (the RPN) to generate the region proposals.

There other network architectures as CenterNet [2] that not require the anchors, the bounding box candidates.

These models generate ROIs (Regions of interested) where an object might be localized, usually, authors use a post-processing algorithm to unify boxes, for example, the NMS (Non-Maximum Supression Algorithm) unifies the ROIs generated by the Region Proposal Network (RPN), candidates to bounding box.

Mask R-CNN [5], extends Faster R-CNN [10] by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition, and therefore, this approach efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance.

3. Dataset and metrics

For the experiments that are going to be performed, we use: KITTI-MOTS [3, 9, 11] which is a set of sequences of frames of urban scenarios with annotations on 2 different classes cars and pedestrians. It contains 21 sequences of images. The original dataset is divided into a train, validation

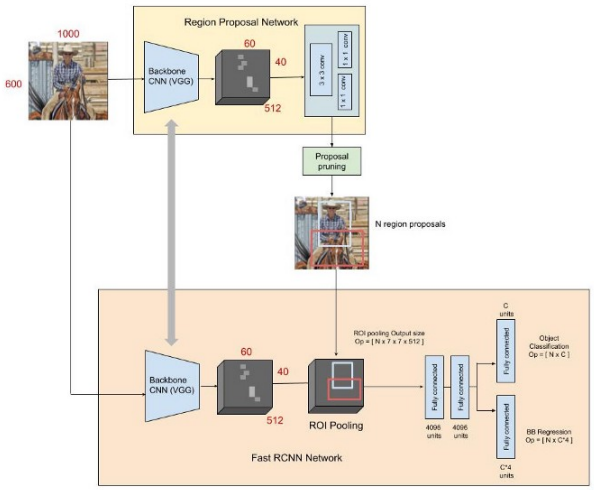

Figure 1. R-CNN architecture [10]

and test sets, but in this work, we will use only train and validation. The sequences of frames used for the training and the validation were the ones specified in the documentation of the dataset.

To evaluate how well the models perform with this dataset, we will use the "CocoEvaluator()" function that the framework detectron2 [4, 12] gives. In order to do that, first we transformed the dataset to the COCO [8] format following the guide to use custom datasets. With this function, we are able to obtain different Average Precision (AP) metrics, from which we can compare better between different models. The metrics are the following:

1. AP: average precision
2. AP50: average precision with 0.5 IOU threshold
3. AP75: average precision with 0.75 IOU threshold
4. APs: average precision with objects that have an area $< 32^2$ pixels
5. APm: average precision with objects that fulfill $32^2 <$ area $< 96^2$ pixels
6. APl: average precision with objects that have an area $> 96^2$ pixels

We also analyze the performance of Faster R-CNN [10] and Mask R-CNN [5] on the Out of Context (OOC) Dataset [1] for object detection. This work explains how the context in the image encapsulates rich information about how natural scenes and objects are related to each other, and therefore, how they are detected by the network.

Model	AP	AP50	AP75	APs	APm	APl
R50-FPN	54.83	79.15	62.77	29.27	60.48	70.82
R50-DC5	51.14	77.22	56.60	24.22	56.25	69.89
R50-DC5x3	52.61	77.55	58.68	25.30	57.50	71.09

Table 1. Average precisions for detection of the models on KITTI-MOTS for Faster R-CNN [10].

Model	AP	AP50	AP75	APs	APm	APl
R50-FPN	54.81	79.26	62.11	39.70	65.06	62.81
R50-DC5	52.21	78.00	58.60	34.78	64.22	60.16
R50-DC5x3	54.32	79.36	60.28	36.76	66.33	60.97

Table 2. Average precisions for detection of the models on KITTI-MOTS for Mask R-CNN [5].

Model	AP	AP50	AP75	APs	APm	APl
R50-FPN	43.76	76.15	44.21	26.64	53.19	62.70
R50-DC5	40.31	72.95	38.38	21.44	49.94	62.71
R50-DC5x3	42.57	75.26	42.44	23.26	52.39	65.79

Table 3. Average precisions for segmentation of the models on KITTI-MOTS for Mask R-CNN [5].

4. Experiments

4.1. Evaluation on models with COCO dataset weights

For the detection task the used architecture was the Faster R-CNN [10] while for instance segmentation the Mask R-CNN [5] architecture was used.

First, we selected 3 models from the model zoo that detectron2 [12] for the two tasks, detection and segmentation which are named R50-FPN, R50-DC5 and R50-DC5x3. All the 3 tested models had as backbone ResNet [6] with 50 layers and the heads were fully convolutional layers, but they had some slight variances in the backbone. For the R50-FPN the ResNet [6] had a standard conv, for the R50-DC5s it had dilations in conv5 and the R50-DC5x3 is the same, but the weights were trained with a learning rate 3 times larger.

With the evaluation method described in the previous section, we obtained the following results for detection on Table 1 and 2, and for instance segmentation on Table 3.

From these results, we can observe a low performance in object which are small, which normally correspond to the pedestrians class. This can be observed if an inference is made, and we can review that road artifacts such as signals or others may produce false positives of pedestrians, as it can be seen in Figure 1.

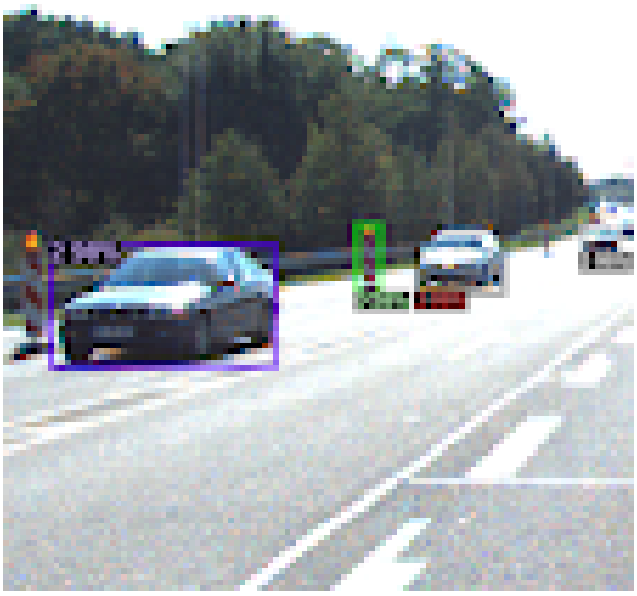

Figure 2. Example of a traffic artifact being detected as a pedestrian [3].

Learning Rate	F mAP Detect.	M mAP Detect.	M mAP Seg.
1e-3	58.038	58.384	43.370
1e-4	55.942	45.572	27.39
1e-5	14.856	52.938	26.838

Table 4. Learning rate effect on (F) Faster R-CNN [10] and (M) Mask R-CNN [5] performance.

4.2. Fine tuning the models

We selected the best model tested from the "model zoo" of detectron2, R_50_FPN_1x, to perform the fine-tuning experiments. Different learning rates and batch size of the ROI head were tested for the detection and segmentation task as they are the ones with most impact to the performance, this ablation study is shown in Tables 4 and 5. We trained models for 2000 iteration and we can observe that a higher learning rate makes the models perform better. We also tried with an even higher learning rate (i.e. 0.005)but in this case the model mAP decreases which indicates that 0.001 is the optimal learning rate. To conclude the fine-tuning, we show in Tables 6 and 7 a comparison between pre-trained and fine-tuned models with the optimal found parameters, as we expected, after fine-tuning models performance improves drastically.

4.3. Qualitative Results

In Figures 3, 4, 5, 5, 6 and 7 we can observe some qualitative results of our models on different scenarios.

ROI BS	F mAP Detect.	M mAP Detect.	M mAP Seg.
64		58.384	45.572
124		57.239	46.363
256	57.160	56.064	44.988
512	58.038		
1024	55.470		

Table 5. ROI Batch size (BS) effect on: (F) Faster R-CNN [10] and (M) Mask R-CNN [5] performance.

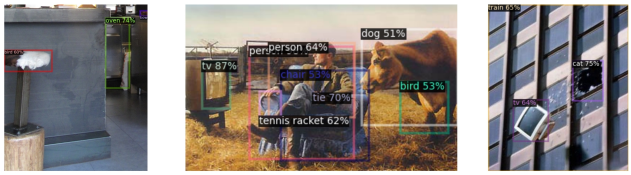

Figure 3. Task A: Out of context with Faster R-CNN [10].

Figure 4. Task A: Out of context with Mask R-CNN [5].

Figure 5. Task C: Qualitatively transplant with Faster R-CNN [10].

- Task A: Images from the out of context split.
- Task B and C: Transplanting randomly new objects to the scene.
- Task D: Leave 1 object with different noisy backgrounds.

From the results we can see that models get confused when objects present similar shapes/features to others (e.g confusing a chair with a bird as both present feathers) and they lower the confidence when the detected object lacked of sense with the whole context of the scene.

5. Conclusion

Object Detection and Instance Segmentation models require longer and more complex training than CNNs for

	AP	AP50	AP75	APs	APm	APl	Person AP	Car AP
Faster R-CNN Pre-trained	43.76	76.15	44.21	26.64	53.19	62.70	28.77	58.75
Faster R-CNN Fine-tuned	58.04	82.24	67.44	32.38	64.00	71.04	52.18	63.69
Mask R-CNN Pre-trained	54.81	79.26	62.11	39.70	65.06	62.81	44.48	65.14
Mask R-CNN Fine-tuned	58.38	80.52	67.88	41.78	68.85	74.94	51.14	65.63

Table 6. Detection performance comparison using: (F) Faster R-CNN [10] and (M) Mask R-CNN [5]. For both models we can observe a significant improvement after fine-tuning in the AP as well as in the person class AP which was the one that gave more troubles.

	AP	AP50	AP75	APs	APm	APl	Person AP	Car AP
Mask R-CNN Pre-trained	43.76	76.15	44.21	26.64	53.19	62.70	28.77	58.75
Mask R-CNN Fine-tuned	45.5	77.03	47.42	28.58	55.85	69.56	31.11	60.02

Table 7. Segmentation performance of Mask R-CNN [5] pre-trained and fine-tuned. For the segmentation task we can see that the impact is not as noticeable as in the detection but the fine-tuning still enhances the model performance.

Figure 6. Task C: Qualitatively transplant with Mask R-CNN [5].

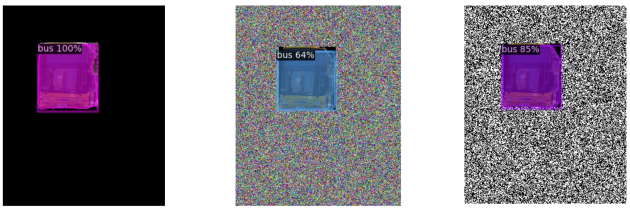

Figure 7. Task D: Feature Inference with Mask R-CNN [5].

other tasks, this means the hyper-parameters (i.e. learning rate, batch size) are very sensitive. fine-tuning on the particular dataset is a must to obtain the greatest performance. We have explored the performance of these models using out-of-context images, forcing the model to have unexpected behaviours, yet, overall, models could perform better than expected. Context, co-occurrence of objects and texture, affect the performance of the network when detecting and classifying objects.

References

[1] Myung Jin Choi, Antonio Torralba, and Alan S. Willsky. Context models and out-of-context objects. *Pattern Recognition Letters*, 33(7):853–862, 2012. Special Issue on Awards from ICPR 2010. 2

[2] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection, 2019. 1

[3] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 1, 3

[4] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron. https://github.com/facebookresearch/detectron, 2018. 2

[5] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1, 2, 3, 4

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1, 2

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. 1

[8] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 1, 2

[9] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixe, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision (IJCV)*, 2020. 1

[10] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016. 1, 2, 3, 4

[11] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. Mots: Multi-object tracking and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7942–7951, 2019. 1

[12] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 2