M5 Object detection

Week 2: Introduction to Object Detection and Segmentation

Group 6: José Manuel López Camuñas, Marcos Conde Osorio, Alex Martín Martínez

INDEX

- KITTI-MOTS
- Run inference
- Evaluation with COCO weights
- Fine-tune on KITTI-MOTS

KITTI-MOTS

This dataset contains 21 sequences of images that include the annotations for two classes: cars and pedestrians. The original dataset is divided into a train, validation and test datasets, but for this project we will be using the train dataset for training and validation, and the official validation for test. The sequences of frames used for the training and the validation were the ones specified in the documentation of the dataset.

KITTI MOTS. We performed the aforementioned annotation procedure on the bounding box level annotations from the KITTI tracking dataset [13]. A sample of the annotations is shown in Fig. 2. To facilitate training and evaluation, we divided the 21 training sequences of the KITTI tracking dataset[2] into a training and validation set, respectively[3]. Our split balances the number of occurrences of each class – cars and pedestrians – roughly equally across training and validation set. Statistics are given in Table 1.

[1] The two frames annotated per object are chosen by the annotator based on diversity.

[2] We are currently applying our annotation procedure to the KITTI test set with the goal of creating a publicly accessible MOTS benchmark.

[3] Sequences 2, 6, 7, 8, 10, 13, 14, 16 and 18 were chosen for the validation set, the remaining sequences for the training set.

KITTI-MOTS

The annotations of the dataset used were images that had the segmented objects, with each object differentiated with a pixel value. This format had to be transformed to the COCO format so that we could evaluate the model with the detectron2 tools. We did that following the tutorial on custom datasets that the documentation gives and knowing that the person and car class in the COCO dataset are 0 and 2 while in the KITTI-MOTS they are 1 and 2 respectively.

Run inference

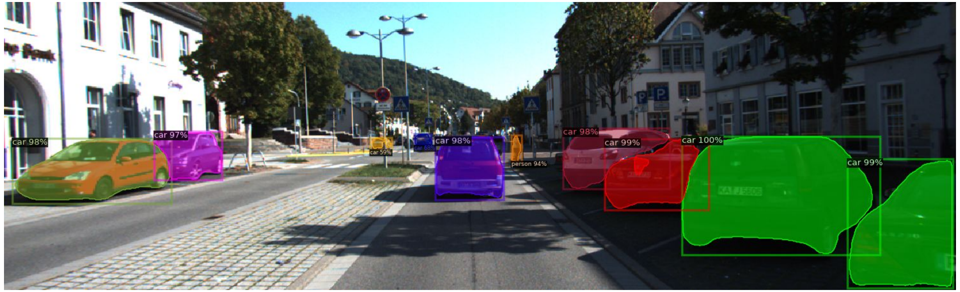
From the model zoo that detectron2, we decide to test the following models and once seen how they performed with the KITTI-MOTS dataset to select the one that we considered that was the best one to perform fine-tuning with.

The models tested for both the detection and segmentation were:

- R50-FPN
- R50-DC5 with lr_schedule 1x
- R50-DC5 with lr_schedule 3x

Examples of inference with Faster R-CNN

Examples of inference with Mask R-CNN



Evaluation with COCO weights

Quantitative results: Faster R-CNN

Models	AP	AP50	AP75	APs	APm	APl	Person AP	Car AP	Inf.time(min:sec)
R_50_FPN_1x	54.83	79.15	62.77	29.27	60.48	70.82	45.37	64.29	3:50
R_50_DC5_1x	51.14	77.22	56.60	24.22	56.25	69.89	41.61	60.68	6:37
R_50_DC5_3x	52.61	77.55	58.68	25.30	57.50	71.09	42.59	62.64	7:44

Evaluation with COCO weights

Quantitative results: Mask R-CNN detection

Models	AP	AP50	AP75	APs	APm	APl	Person AP	Car AP	Inf.time(min:sec)
R_50_FPN_1x	54.81	79.26	62.11	39.70	65.06	62.81	44.48	65.14	4:23
R_50_DC5_1x	52.21	78.00	58.60	34.78	64.22	60.16	42.175	62.255	7:20
R_50_DC5_3x	54.32	79.36	60.28	36.76	66.33	60.97	44.65	63.99	7:56

Evaluation with COCO weights

Quantitative results: Mask R-CNN segmentation

Models	AP	AP50	AP75	APs	APm	APl	Person AP	Car AP
R_50_FPN_1x	43.76	76.15	44.21	26.64	53.19	62.70	28.77	58.75
R_50_DC5_1x	40.31	72.95	38.38	21.44	49.94	62.71	24.81	55.81
R_50_DC5_3x	42.57	75.26	42.44	23.26	52.39	65.79	27.92	57.21

Evaluation with COCO weights

Conclusions from quantitative results

Looking the metrics obtained after evaluating the models, we can observe that the behavior is very similar, although for the R_50_FPN_1x we obtain better values. One of the most remarkable things that we can observe is that for all the models, the AP for the pedestrian class is generally a 20% lower than for the cars.

In order to establish why it is that, we will take a look at the qualitative results.

Evaluation with COCO weights

Qualitative results: Faster R-CNN

From this two examples we can see that the models could predict pretty much correctly cars and persons when they where in the images.

Evaluation with COCO weights

Qualitative results: Faster R-CNN

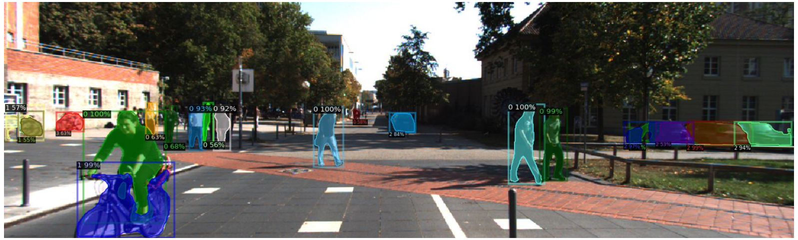
Here we can see how a traffic signal is detected as a pedestrian, which may cause the low AP in the person class as this object appears repeatedly in the different sequences.

Also, we encountered this case where the shape of a person inside the car was detected as a pedestrian, which was not

Evaluation with COCO weights

Qualitative results: Mask R-CNN

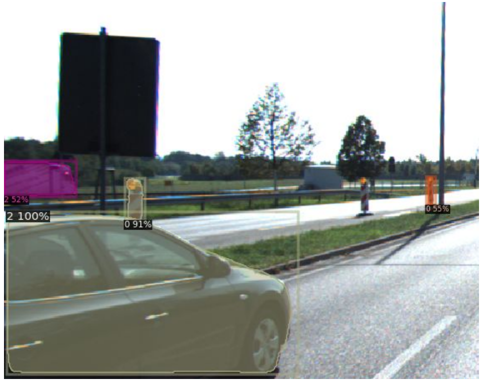
Good examples of detection and segmentation of pedestrians and cars.

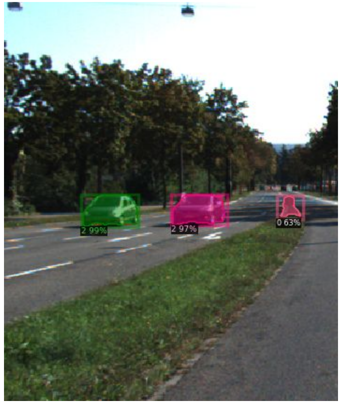


Evaluation with COCO weights

Qualitative results: Mask R-CNN

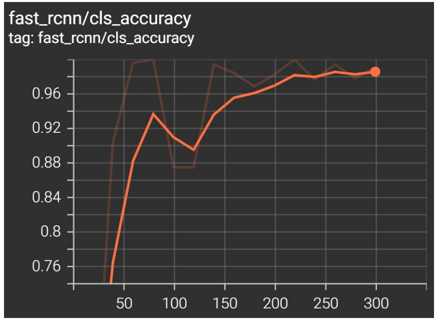
Here we also observe the same behavior as with the detection of the traffic signals and other artifacts in the road are detected as pedestrians

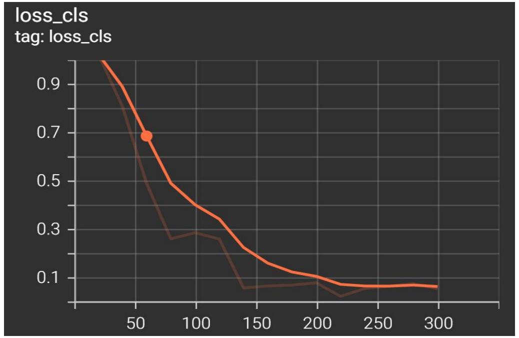




Fine-tune on KITTI-MOTS

This section was intended to be done, but this night the PC where the results were saved and trainings were done crashed, and we couldn't recover any information as there wasn't any push done to the repo. The error was not to save any info, which we should have done. Even though we can't present a fine-tuning we wanted to show that the code uploaded to GitHub ran. Here we deliver some plots from Tensorboard.





Start writing paper

Although the problems with fine-tuning we started to write the paper at overleaf.

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

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. We will not focus on this task, as the model's require to perform it are too 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) [5, 6].

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

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 [9] paper fixes this by using another convolutional network (the RPN) to generate the region proposals.

There other network architectures as CenterNet [1] 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 [4], extends Faster R-CNN [9] 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 [2, 8, 10] which is a set of sequences of frames of urban scenarios with annotations on 2 different

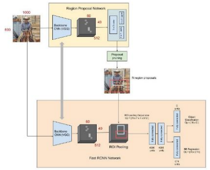


Figure 1. R-CNN architecture [9]

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

4. Experiments

4.1. Evaluation on models with COC0 dataset weights

For the detection task the used architecture was the Faster R-CNN [9] while for instance segmentation the Mask R-CNN [4] architecture was used

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 [9].

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 [4].

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 [4].

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.

4.2. Fine tuning the models

References

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

[2] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel

Final summary

Dataset

The main challenge of this week was to understand the format and how to transform the data in order to achieve the required format for the framework to run correctly.

Fine-tuning

We couldn't perform an analysis of the results from fine-tuning as they were lost, but from the previous plots seen we can observe that the models are able to perform very well once they are trained with the KITTI-MOTS dataset.

Evaluation

With the models pretrained on COCO datasets only, the models don't perform very well (as expected) and it is seen a bias to make wrong detections of pedestrians with other road artifacts. The best model that we found was the R50-FPN.

Models	Person AP	Car AP
R_50_FPN_1x	44.48	65.14
R_50_DC5_1x	42.175	62.255
R_50_DC5_3x	44.65	63.99