CS230

Mask R-CNN Application: Instance Segmentation in Driving Scenes

Xiao Lu
Department of Management & Science Engineering
xiaolu@stanford.edu

Chen Luo
Department of Electrical Engineering
chenl3@stanford.edu

Michelle Zhang
Department of Aeronautics Astronautics Engineering
zhangmx@stanford.edu

Abstract

One of the major challenges in autonomous driving is the ability to understand the environment including surrounding vehicles, traffic signs, and pedestrians at a finer-grained level. In this project, we investigate and evaluate the performance of the state-of-the-art model for instance segmentation, Mask R-CNN, on the newly-released Mapillary dataset, whose images focus specifically on driving scenes. We transfer the learning results from the pre-trained weights from COCO dataset and fine tune the final layers for Mapillary images. The results show significant improvement in precision measurements from the baseline, and surpassed the benchmark in overall mean average precision.

1 Introduction

Deep learning techniques based on CNN backbones have become the top choice for complex computer vision problems, by enabling efficient inferring knowledge in imagery [1]. However, the results are still too simple compared to the complexity and diversity of human-level visual comprehension [2]. The goal of instance segmentation is to provide both object detection and pixel-level classification for different instances of the same class. The finer granularity of instance-level understanding of the images will add critical insights to the perception of autonomous driving. In 2017, the Mask R-CNN network proposes a simple but effective strategy to perform instance classification and segmentation with minimal computation overhead [4].

In this project, we implemented our algorithm on the newly released Mapillary Vistas Dataset [3], which focuses on traffics and road environment. We apply a Mask R-CNN model that was previously trained on the COCO dataset and fine-tune it for Mapillary images to obtain instance-level segmentation outputs.

2 Dataset

The Mapillary Vistas dataset [3] contains 20,000 high-resolution street-level images on multiple locations around the world. 37 object categories are labeled with pixel-wise instance-level annotations.

There is no published paper introducing the usage or implementation on the data quality because of the novelty of the Mapillary dataset. Two other popular object detection and segmentation datasets, CityScapes and COCO, are compared with Mapillary dataset here. Microsoft Common Objects in Context (COCO) is a diverse set of general objects which provides a good baseline for image recognition, segmentation and captioning. In comparison, Mapillary is characterized by its diversity specific to traffic environment.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Compared to CityScapes dataset, the Mapillary dataset is closer to real-world application with a variety of weather, season, and time of day. Moreover, Mapillary has a much better labeling quality and is more fine-grained. So the general contour of the objects are preserved much better. This requires our model to be more robust to learn the fine-grained labeling and tolerate diversity.

2.1 Pre-processing of Mapillary Dataset

The published results are evaluated over MS COCO data set. For reasonable comparison, we take the intersection between the 80 object classes in MS COCO and the 37 instance-specific classes in Mapillary. The intersection contains 11 object classes: Bird, Person, Bicyclist, Motorcyclist, Bench, Car, Person, Fire Hydrant, Traffic Light, Bus, Motorcycle, and Truck, in addition to 1 background class required by default.

To support training multiple images, all images are resized to the fixed size (1024×1024 px by default). The non-square image is padded with all zeros. The smallest side is 800 px and the largest is trimmed at 1000 px.

2.2 Dataset Usage

We split the distribution of the dataset for training, development and testing as shown in Table 1. Because we focus on only 11 classes, we leave out images which contain no instance of the interested classes during training. Because filtering process depends on the mask threshold (explained in methodology), we have different sets of images for run 1 and run 2 as they have different mask threshold, even though their split sizes are identical.

Name	Training	Development	Test
baseline	20	10	-
run 1	4,096	512	-
run 2	4,096	512	-
run 3	16,384	1,024	1,024

Table 1: Train/dev/test set splits

3 Methods

3.1 Mask R-CNN Framework

Mask R-CNN [4], is an efficient and effective algorithm for instance segmentation. As an extension of the Faster R-CNN model. The innovating aspect is that Mask R-CNN decouples class prediction and mask generation. The original Faster R-CNN has two outputs for each candidate object: a class label and a bounding-box offset. Mask R-CNN adds a third branch that outputs the object mask extracting a much finer spatial layout of an object, Fig. 1. As a result, Mask R-CNN is capable of performing object instance segmentation with much higher efficiency. The Mask R-CNN model that we used is based on an open-source implementation by Matterport [6], built on Feature Pyramid Network (FPN) and ResNet-101 as backbone.

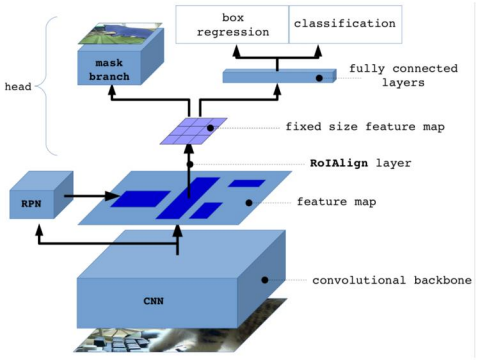


Figure 1: Mask R-CNN framework for instance segmentation [8]

3.2 Loss Function

Mask R-CNN adopts the same two-stage as Faster R-CNN:

The first stage, RPN (Regional Proposal Network) is left unchanged. RPN outputs k potential bounding boxes with certain aspect ratios (anchors) and evaluate how good each of these anchors is expected to be. The second stage outputs a binary mask for each RoI in parallel to predict the class and box offset. Specifically, the mask is selected from the label of each RoI classification in parallel by the dedicated classification branch.

Thus, the loss is defined for the three tasks on each sampled ROI: class prediction, bounding box refinement and mask generation. Class prediction loss and bounding box regression loss are collected from both RPN and mask generation stages, whereas mask loss is taken only from mask generation stage. Mask loss is only defined per each individual class to avoid competition among other mask outputs.

$$L_{RPN} = L_{r,class} + L_{r,box}, \quad L_{mask} = L_{m,class} + L_{m,box} + L_{mask}, \quad L = L_{RPN} + L_{mask},$$

3.3 Hyperparameters Tuning

We first set up a baseline on run 1 with all default hyper-parameters and select learning rate (LR) and mask threshold (MT) as hyperparameters of interests. We introduce the concept of mask threshold to filter out image sizes below than the threshold value. This concept stems from two practical considerations. First, when our Mask-RCNN model scales all input images down to a predetermined size of 1024 by 1024, small objects from the original image (typically 4000 by 3000) will shrink to even smaller objects in output, which become indistinguishable even by human eyes. Secondly, small masks refer to objects either so small or so far away that they are less critical in driving decision making.

We train the model on different mask thresholds but keeps at a learning rate fixed at 10^{-4} (run 1 and 2). As shown in Fig. 2b, higher threshold turns out to result in higher loss in the initial stage of training, but later converges to a similar level to that of run 2 with lower MT. For the remainder of the project we train on MT = 32^2. We assume that objects smaller than this size appears small enough to be safely ignored (see Fig. 2d).

As we are applying a pre-trained network to a new dataset whose classes and image resolutions are different from the original COCO dataset, the learning rate must be tuned for the new dataset. We train the model on run 2 with different learning rates for 8 epochs. As shown in Fig. 2a, LR = 10^{-3} yields lower loss than LR = 10^{-4} and similar loss as LR = 10^{-2}. Thus, we select LR = 10^{-3} for the remainder of the project. We keep weight decay of 0.0001 and momentum of 0.9 as default in the given code.

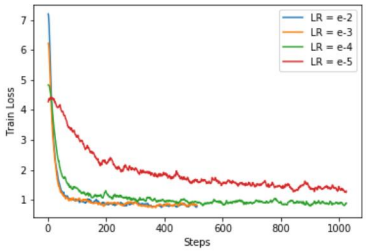
Finally, we train the model on run 3, with 16,384 images in training set and 1,024 images in development and test sets. We set LR to 10^{-3} and MT to 32^2, and leave the rest of the hyper-parameters to default. We train the model on 1 GPU and each mini-batch with 8 images for 8 epochs. We also run 2 experiments: one for head classifier layers only (RPN, classifier and mask heads of the network), i.e, we fix all parameters from earlier layers and only train the head classifier layers; another one experiment for ResNet stages 5 and above after the top-layer classifiers have been trained. Lastly, we repeat the procedure on run 3 with all 37-instance classes from the Mapillary dataset. The training process is summarized in Table 2.

4 Results and Evaluation

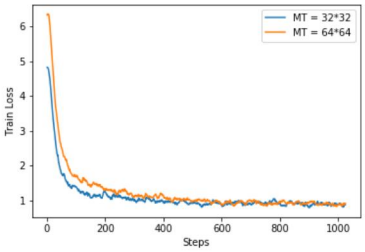
The instance segmentation results on four randomly chosen Mapillary images from our test set is shown in Fig. 3. The network achieves high confident-level on each detected object (white labels in the image), in different lighting and road conditions.

4.1 Evaluation Metrics

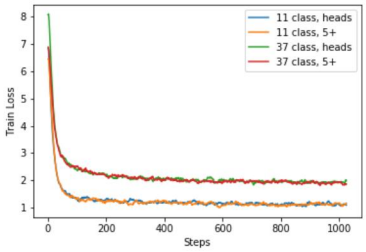
One common approach to evaluate instance segmentation performance is average precision (AP) over all possible classes under a certain IoU threshold. In this application, we define a positive example with mask IoU greater than some IoU threshold.



(a) Tuning learning rate



(b) Tuning mask threshold



(c) training 11 and 37 classes

(d) Mask threshold

Figure 2: Hyperparameters tuning

Data	LR	MT	Epoch	class	layers
baseline	10^{-3}	5000	1	11	heads
baseline	10^{-3}	5000	1	37	heads
run 1	10^{-4}	64^2	8	11	heads
run 2	10^{-4}	32^2	8	11	heads
run 2	10^{-5}	32^2	8	11	heads
run 2	10^{-2}	32^2	4	11	heads
run 2	10^{-3}	32^2	4	11	heads
run 3	10^{-3}	32^2	8	11	heads
run 3	10^{-3}	32^2	8	11	5+
run 3	10^{-3}	32^2	8	37	heads
run 3	10^{-3}	32^2	8	37	5+

Table 2: Training summary

The value for IoU threshold is referenced from MS COCO dataset evaluation metrics [4] with fixed IoU threshold as 0.50 ($AP_{.50}$), 0.75 ($AP_{.75}$), as well as the average AP value over [0.50, 0.95] with an increment of 0.05 (AP).

We consult results from Facebook and Matterport [6] evaluated on MS COCO dataset as benchmarks. Baselines are evaluated on Mapillary dataset with pre-trained weights. The fine-tuned network results is named as DreamNet, with heads/5+ layers corresponding to two experiments. For each network, we assess multiple MT and number of classes. The results is organized in Table 3.

	class	MT	AP	$AP_{.50}$	$AP_{.75}$
Facebook	80	-	35.7	58.0	37.8
Matterport	80	-	35.1	59.4	36.5
Baseline	11	-	29.6	47.1	36.2
DreamNet-heads	11	32^2	37.3	57.9	45.8
DreamNet-5+	11	32^2	36.8	58.1	45.2
DreamNet-heads	11	64^2	49.6	72.4	61.2
Baseline	37	-	5.6	12.1	6.3
DreamNet-heads	37	32^2	15.1	25.4	18.5
DreamNet-5+	37	32^2	16.0	26.6	19.9

Table 3: Evaluation results of Facebook, Matterport, and DreamNet

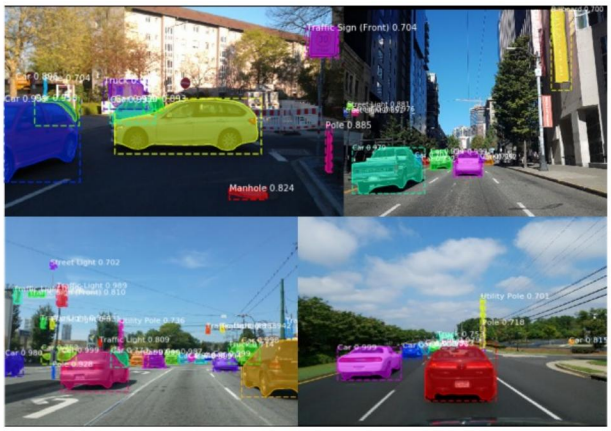


Figure 3: Sample instance segmentation results in 38 classes

Our results show that the 11-class model achieves comparable $AP_{.50}$ and AP performances as benchmarks, and yields a superior $AP_{.75}$ of 45%. Specifically, the $AP_{.50}$ reaches 72.4% on roadside images. The improvement in $AP_{.75}$ proves that the 11-class model performs better in pixel-level classification with large IoU, i.e. when predicted bounding box significantly overlaps with ground truth bounding box.

The results for 37-class model are not as promising as the 11-class one, failing to match neither benchmark by a wide margin in AP, $AP_{.50}$ or $AP_{.75}$. Nevertheless, it shows an great improvement compared with its baseline, with 200% increase in AP and $AP_{.75}$. Since the 26 newly added classes have not been seen by the pre-trained network, 8-epoch training duration may not be enough to learn the features of the object. We believe longer training steps would lead to comparable results as the 11-class model.

4.2 Discussion

Scaling down images from 3000 by 4000 to 1024 by 1024 means many details in the instance labels are lost. It may result in failures of recognizing small objects, where each pixel may be crucial on detecting tasks. To fully utilize the Mapillary dataset, we suggest that the input dimension be set to the original level.

We have make an assumption that all objects smaller than 32^2 (relative the the original image) can be safely ignored. This may be true for vehicles far away in a safety distance, but a red traffic light of 32^2 size may be important enough for decision making in self-driving. Ideally, we recommend varying mask threshold for different classes. For example, once the model decides that the object can be safely ignored, it can skip the segmentation step and allocate resources to other objects required for finer-grained understanding.

Our low AP scores in 37-class model suggests that in order for the model to learn the 26 new classes, more training data with targeted objects are necessary. Interpolated linearly from the AP improvement and our dataset size, we suggest that at least 35,000 images be acquired to train the 26 new classes.

5 Conclusion

In the project, we fine-tuned a pre-trained Mask R-CNN network on Mapillary dataset that focuses on traffic environment. The backbone we use in this model is ResNet-101 and FPN. Results show that a 11-class model surpasses benchmarks in terms of $AP_{.75}$, and achieves comparable result in $AP_{.50}$ and AP as the benchmark. On the other hand, we see a huge improvement on precisions of the 37-class model, but it still far from practical usage. We attribute this under-performance to the insufficient quantity of data, and suggest further training on larger dataset.