ELL793 Computer Vision
Object Detection: Faster RCNN & RetinaNet

Abhinava Sikdar 2017MT10724
Yashank Singh 2017MT10756

December 31, 2020

1 Introduction

Object detection is a core and vital task in the field of computer vision and a natural extension of object classification. Herein, in addition to classifying the objects, we expect the model to localise the object within the image using a bounding box. Another challenge in object detection is the presence of multiple instances of the same of different classes withing the same image. Varying scale and aspect ratios of object instances amongst different images is also a problem. Crowd counting, self-driving cars, video surveillance, face detection and anomaly detection are some of the domains wherein object detection is heavily deployed.

2 Metrics

mAP or mean average precision is the most popular metric for object detection. Precision is defined as

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

A proposed bounding box is considered to be a true positive if it has an Intersection over Union(IoU) with a ground truth bounding box greater than some threshold. This threshold is usually set to a 0.5 value. Similar to precision, recall is defined as

$$Recall = \frac{True\ Positives}{Total\ Positives}$$

Note that while the value of precision keeps fluctuating while parsing through the data, the value of recall in a non-decreasing function since the denominator i.e. the *Total Positives* remain constant throughout. The Area under the Curve(AuC) of the smoothed Precision vs Recall graph is defined to be the Average Precision(AP). The mean of the Average Precisions over all the classes is defined to be the mAP.

Figure 1: The PASCAL VOC 2007 Dataset

3 PASCAL VOC 2007

The goal of the PASCAL Visual Object Classes Detection Challenge 2017 was to detect objects from a number of visual object classes in realistic scenes. There were a total of twenty classes present

- Person: person
- Animal: bird, cat, cow, dog, horse, sheep
- Vehicle: aeroplane, bicycle, boat, bus, car, motorbike, train
- Indoor: bottle, chair, dining table, potted plant, sofa, tv/monitor

The train data consisted of about 2501 images whereas the test data consisted of about 5011 images. Some images from the dataset are shown in Fig.(1)

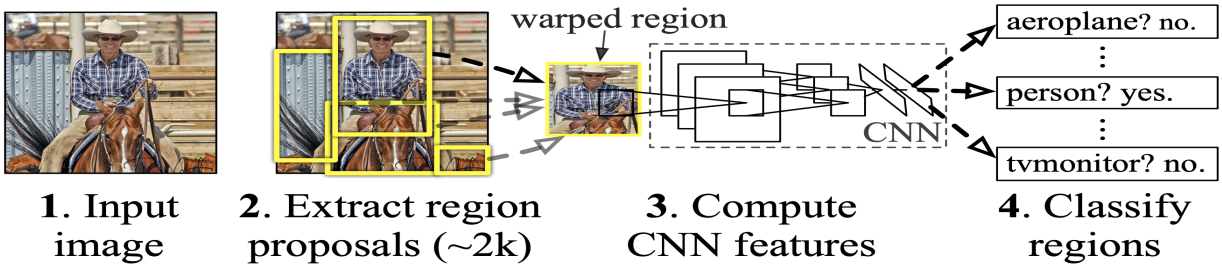


Figure 2: Object detection schematic of RCNN

4 RCNN

Before 2014, most of the object detection techniques were based on SIFT and HOG representations. The progress on the PASCAL VOC object detection was especially slow from 2010-2012. The paper on RCNN [2] was a firm attempt to replicated the success of CNNs in classification at the ILSVRC 2012. Their approach included using selective search to generate about 2000 category-independent region proposals for each image. Each of these proposals are then warped into a 227×227 pixel size and passed through a CNN to obtain a feature vector of size 4096. Since for image, there are 2000 region proposals, we end up with a 2000×4096 vector. This is then fed into a linear SVM to generate class-wise scores for each of the region proposals. Given, all scored regions in an image, greedy non-maximal suppression (for each class independently) is applied which rejects a region if it has an IoU

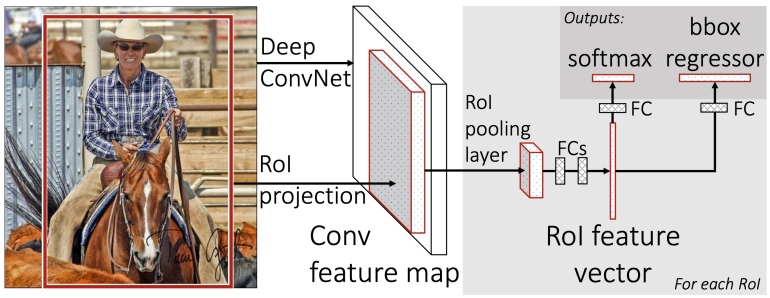


Figure 3: Object detection schematic of Fast RCNN

overlap with a higher scoring region larger than a learned threshold. Before the actual training, they also performed supervised pre-training through classification on the ILSVR2012 dataset. Through this, they were able to achive a mAP of 53.7%, outperforming their nearest competitor [6] at 35.1%. The schematic of RCNN can be seen in Fig.(2). The model was fine tuned on top of weights learned from ImageNet with minibatches of 128 RoIs, 64 from each image. These were sampled to ensure that at least 25% of the RoIs had an IoU of more than 0.5 with some ground truth bounding box. For these RoIs, $u \geq 1$, for all other RoIs, u was set to 0. While testing, the forward pass assigns each RoI a distribution p and a set of bounding boxes r for each of the K classes. Non-maximal suppression is then applied independently for each class as done in [2].

5 Fast RCNN

Despite the success of RCNN, it had some drawbacks. These included multi-stage pipelined training of the ConvNet, the SVM classifiers and the bounding-box regressors. The training is computationally very expensive since for training the later two, features for each proposal have to be extracted and stored in the disk. Even during testing, the RCNN is slow because it performs a separate forward pass for each region proposal without sharing any computation. To tackle all these challenges, Fast RCNN [1] was proposed. Not only did Fast RCNN performed better with respect to mAP than RCNN, it employs a single-stage training via a multi-task loss. This ensures that no disk storage is required for caching features. The Fast RCNN takes as an input the entire image and a set of object proposals. This entire image is passed through a serious of convolutional and pooling layers to obtain the final convolutional feature map. The rectangular window in this final feature map corresponding to a region proposal is called the Region of Interest(RoI). On this RoI, an RoI pooling layer is applied to obtain a feature map of fixed size. This feature map is fed into a fully connected network which ends with two sibling modules- one for classification of object class, the other for regression on the bounding box. This can be seen in Fig.(3). For each RoI with a ground-truth class u and a bounding-box regression target v, the multi-task loss L is given by

$$L(p, u, t^u, v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(t^u, v)$$

where $L_{cls}(p, u) = -\log p_u$ is the log loss for the true class u, λ is a hyperparameter for weighing the two losses, set to 1 in [1] the L_{loc} is the loss for the bounding box regressor.

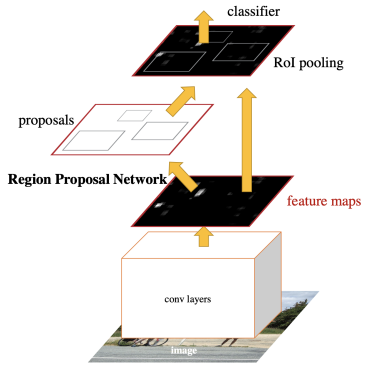


Figure 4: Schematic of Faster RCNN

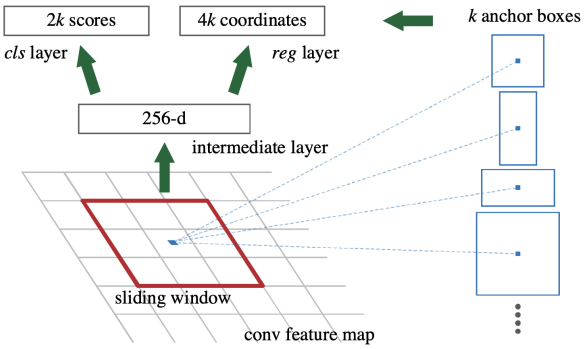


Figure 5: RPNs and anchors

6 Faster RCNN

Earlier state of the art models such as [1] and [3] achieved real time detection given the region proposals, revealing the bottleneck to be the regional proposal system. In [5], a new region proposal network(RPN) is proposed which works in conjunction with the Fast RCNN of [1] to achieve detection rates of upto 5 frames per second, including generation of region proposals. The overview can be seen in Fig.(4) The RPN and the Fast RCNN the starting few convolution layers(13 in VGG-16). To generate region proposals, a small network is made to slide over the last shared convolutional feature map. This small network takes as input a small spatial window of size $n \times n$ as seen in Fig.(5). The resultant feature is fed into two sibling fully connected layers- a box-regressor and a box-classifier. At each sliding window position, upto 9 multiple region proposals are predicted simultaneously. These have been called anchors. An anchor can have 3 possible sizes and 3 possible aspect ratios, hence the 9 possibilities. Note that these anchors are translation-invariant when it comes to object proposals. The design of multi- scale anchors is a key component for sharing features without extra cost for addressing scales. In addition to the loss function of [1], the loss term for the RPN is

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

where i is the index of the anchor in a mini-batch and p_i is the predicted probability of the anchor i being an object. p_i^* is the ground truth of objectness. t_i is the predicted bounding box and t_i^* is the ground truth. L_{cls} is the log loss and L_{reg} is the smooth L_1 loss used in [1]. N_{cls} and N_{reg} are regularisation terms and λ is a hyperparameter for balancing the two losses but doesn't make much of a difference in reality as claimed by the paper. The RPN was trained end to end through mini batches arising out of a single image containing positive and negative anchors in a fixed ratio (1:1 in the original work) as done in [1]. Another novelty is how the RPN and the Fast RCNN were trained using an alternating strategy. This included first training the RPN and then use the proposals to train the Fast RCNN. The network tuned by Fast RCNN is then used to initialise the RPN and this process is iterated over and over.

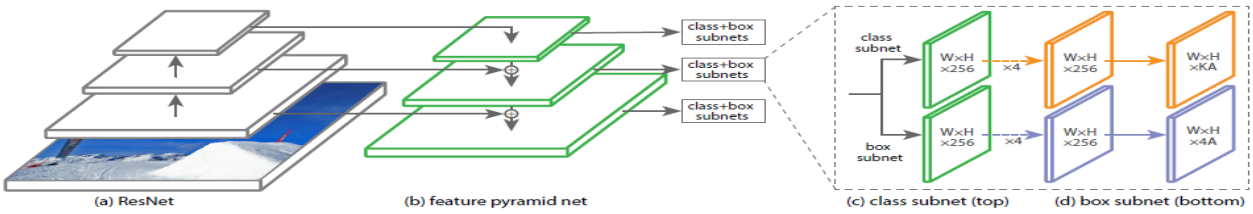


Figure 6: Object detection schematic of RetinaNet

7 RetinaNet

RetinaNet developed by Facebook AI Research was a breakthrough in one stage object detection models. Contrary to their two-stage proposal mechanism driven counterparts one-stage detection networks performed dense sampling of regions in an image to be processed. In the two stage models sparse region proposals are given by the first stage that are then processed by the second stage. The one-stage models are faster but less accurate due to dense-sampling resulting in a lot of information deficient background regions being sampled that inflate the Cross Entropy loss as shown in Fig.(7). This problem is referred to as class imbalance in [4] and pointed out as the reason why these models fall short in performance. This is alleviated by the proposed focal loss given by :

$$FL(p_t) = -(1-p_t)^{\gamma}\log(p_t)$$

This is contrary to cross entropy loss, down-weighs the easy examples and hence improves learning. For example, consider a background region that is correctly being predicted with probability $p_t = 0.9$ then for a value of $\gamma = 2$ this will be down-weighed $100\times$ than in cross entropy loss. Thus, this helps reduce the loss inflation due to easy background examples and focuses more on hard-positive examples. The architecture uses a CNN backbone and a FPN on top of it as shiwn in Fig.(6). Anchor boxes with different area sizes and aspect ratios are used for region proposals obtained from multilpe levels of FPN to be fed into classification and regression subnets. This classification subnet predicts the probability of object presence at each spatial position for each of the A anchors and K object classes. The subnet is a FCN which applies four 3×3 conv layers, each with C filters and each followed by ReLU activations, followed by a 3×3 conv layer with KA filters. (K classes, A=9 anchors, and C = 256 filters). The Box-regression subnet is identical to the classification subnet except that it terminates in 4A linear outputs per spatial location. It is used for the purpose of regressing the offset from each anchor box to a nearby ground-truth object, if one exists. It is a class-agnostic bounding box regressor which uses fewer parameters, which is found to be equally effective. During inference the network only decodes box predictions from at most 1k top-scoring predictions per FPN level, after thresholding detector confidence at 0.05.

Figure 7: Class imbalance in one-stage models due to CE loss

Table 1: mAP on test and validation splits

model/split	Train mAP		Test mAP	
	IoU=0.5	IoU=0.9	IoU=0.5	IoU=0.9
Faster RCNN	0.9837	0.1315	**0.8039**	0.0588
RetinaNet	0.9618	0.5947	0.7549	**0.2655**

8 Results on Faster RCNN and RetinaNet

For both Faster RCNN and RetinaNet, we use the torchvision library. It consists of implementations of both the models on a ResNet-50 backbone. We use a mini batch size of two images along with the Adam Optimizer and train both the models for 20 epochs. The mAP scores for Faster RCNN and RetinaNet on the train and test splits of the dataset are shown in Table 1. Clearly, Faster RCNN performs better when the IoU threshold is set to be lower i.e 0.5 whereas it is the RetinaNet which performs better when the IoU threshold is set o be larger i.e. 0.9.

Table 2: Class-wise AP on test and train splits for IoU thresholds 0.5 & 0.9

class/ model	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv monitor	**mAP**
Test Set Results																					
IOU=0.9																					
RetinaNet	0.377	0.225	0.324	0.280	0.194	0.314	0.295	0.357	0.192	0.249	0.061	0.345	0.236	0.218	0.162	0.267	0.204	0.251	0.377	0.379	**0.265**
Faster-RCNN	0.027	0.042	0.059	0.037	0.027	0.027	0.154	0.016	0.022	0.059	0.067	0.081	0.083	0.048	0.099	0.044	0.050	0.098	0.036	0.097	0.059
IOU=0.5																					
RetinaNet	0.827	0.760	0.782	0.685	0.703	0.732	0.831	0.884	0.652	0.735	0.534	0.878	0.839	0.822	0.828	0.598	0.686	0.716	0.830	0.768	0.755
Faster-RCNN	0.823	0.842	0.768	0.724	0.749	0.784	0.854	0.894	0.705	0.852	0.682	0.902	0.891	0.866	0.858	0.663	0.801	0.7786	0.814	0.819	**0.804**
Train Set Results																					
IOU=0.9																					
RetinaNet	0.898	0.569	0.743	0.576	0.423	0.675	0.650	0.800	0.467	0.545	0.116	0.761	0.513	0.528	0.442	0.589	0.494	0.506	0.851	0.792	**0.595**
Faster-RCNN	0.069	0.089	0.143	0.077	0.070	0.046	0.309	0.023	0.047	0.130	0.174	0.165	0.224	0.120	0.244	0.104	0.099	0.202	0.088	0.198	0.132
IOU=0.5																					
RetinaNet	0.999	0.979	0.982	0.966	0.941	0.967	0.976	0.985	0.944	0.965	0.790	0.993	0.980	0.993	0.970	0.963	0.899	0.950	1.0	0.994	0.962
Faster-RCNN	0.996	0.990	0.985	0.974	0.966	0.994	0.993	0.997	0.981	0.977	0.947	0.999	0.987	0.992	0.987	0.987	0.945	0.984	0.997	0.996	**0.984**

Table 2 shows the class-wise average precisions on the test and validation splits for IoU thresholds 0.5 and 0.9. Bar graph visualizations of these results can be seen in Fig.(11), Fig.(12), Fig.(13) and Fig.(14).

8.1 mAP vs IoU

Here we investigate the effect of varying the IoU on the mAP score for both test and train splits. Once again, Faster-RCNN does better on lower IoU upto 0.7. AS intuition says, increasing the threshold decreases the mAP for all the models Fig,(8). This is because the predictions with bounding boxes offset by even a small amount are discarded at higher IoU's. However it is noteworthy to see that the decrease is sharp on going from 0.7 to 0.9 in Faster-RCNN and RetinaNet with CEloss as compared to RetinaNet with focal loss. This is in line with claim of [4] that the model with focal loss is able to focus more on the hard positive examples and hence result in better precision on higher IoU thresholds.

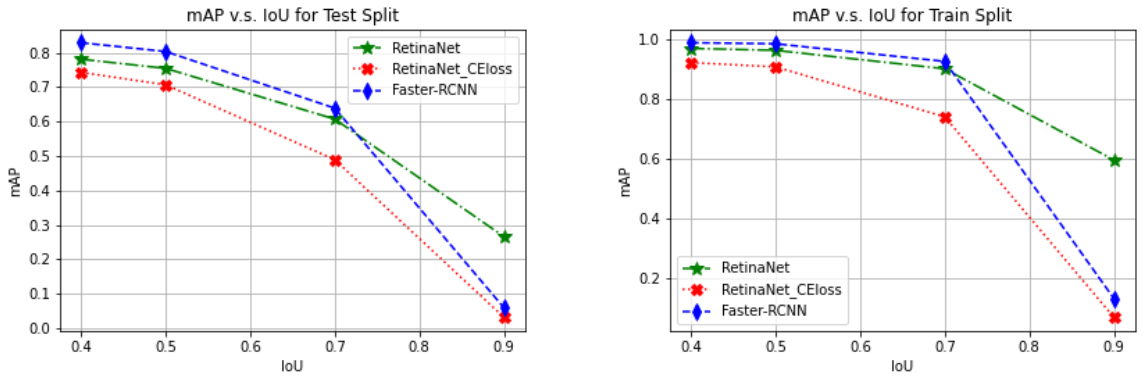


Figure 8: mAP with varying IoU thresholds

8.2 Replacing Focal Loss in RetinaNet with Cross Entropy Loss

In this section we investigate the performance of RetinaNet with Vanila Cross Entropy Loss. Being a one stage mechanism, RetinaNet lacks in performance with CE loss as compared to Faster-RCNN as well as RetinaNet with focal loss as shown in Table(3). As highlighted in the paper the class imbalance hampers performance. With CE loss RetinaNet achieves an mAP of 0.7071 on test split at IoU of 0.5 which is 5% less than that with focal loss. Also, at higher IoU of 0.9 the test and train mAP drop drastically with test mAP of only 0.0309. These results corroborate the claim of [4]. Hence, focal loss indeed improves training of one-stage mechanism by focusing on learning from hard examples and thus alleviating the problem of class imbalance caused by large number of easy background examples.

Table 3: RetinaNet with Cross Entropy mAP

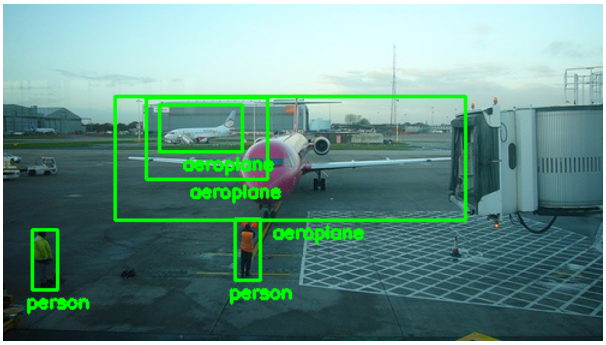
model/split	Train mAP		Test mAP	
	IoU=0.5	IoU=0.9	IoU=0.5	IoU=0.9
RetinaNet with CE loss	0.9064	0.0686	0.7071	0.0309
RetinaNet with focal loss	0.9618	0.5947	**0.7549**	**0.2655**

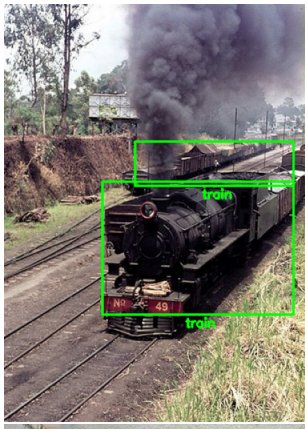
8.3 Which is better for the PASCAL VOC 2007 challenge?

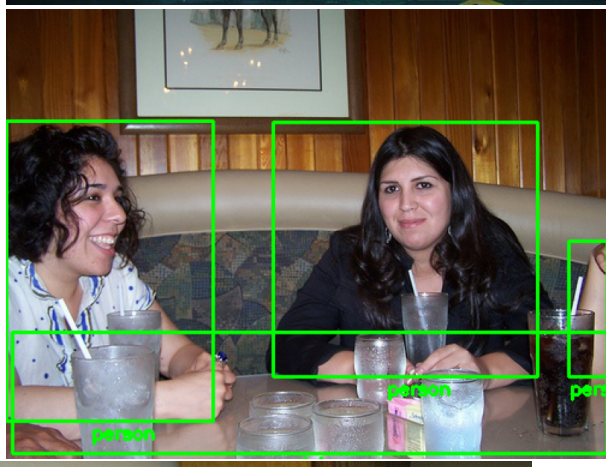
We have seen from our results that Faster RCNN achieves a higher mAP than RetinaNet at lower IoU thresholds whereas it is the RetinaNet which scores a higher mAP at higher IoU thresholds. The PASCAL VOC 2007 challenge for object detection considered the mAP score at a threshold of IoU = 0.5 and hence Faster RCNN would be better positioned to win against RetinaNet despite the later being a more advanced model. However had the mAP been considered at a higher threshold in the competition, say an IoU = 0.9, then RetinaNet would have been our preference.

8.4 Visualizations

Here are some of the visualizations of predictions done by the Faster RCNN on the test split.

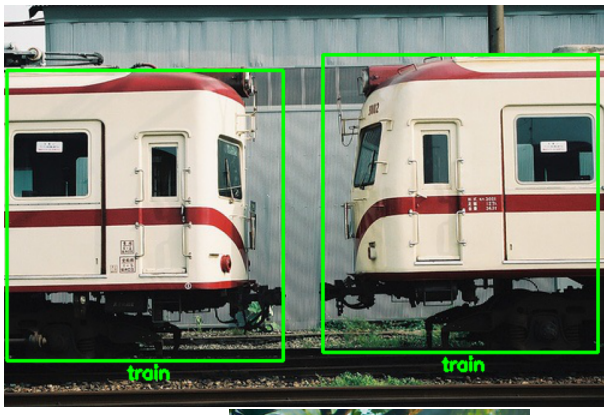

aeroplane
aeroplane
aeroplane
person
person

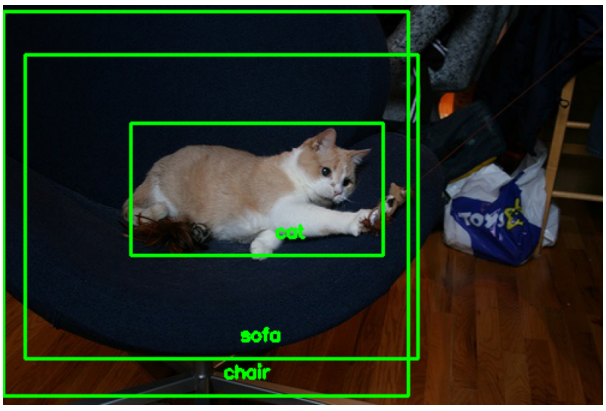

train
train

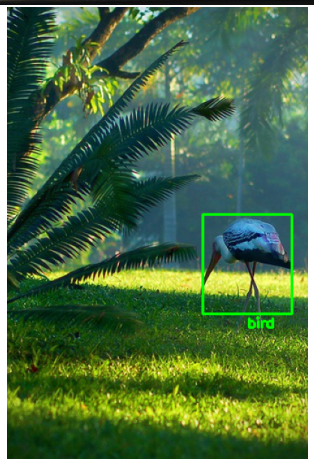

person
person

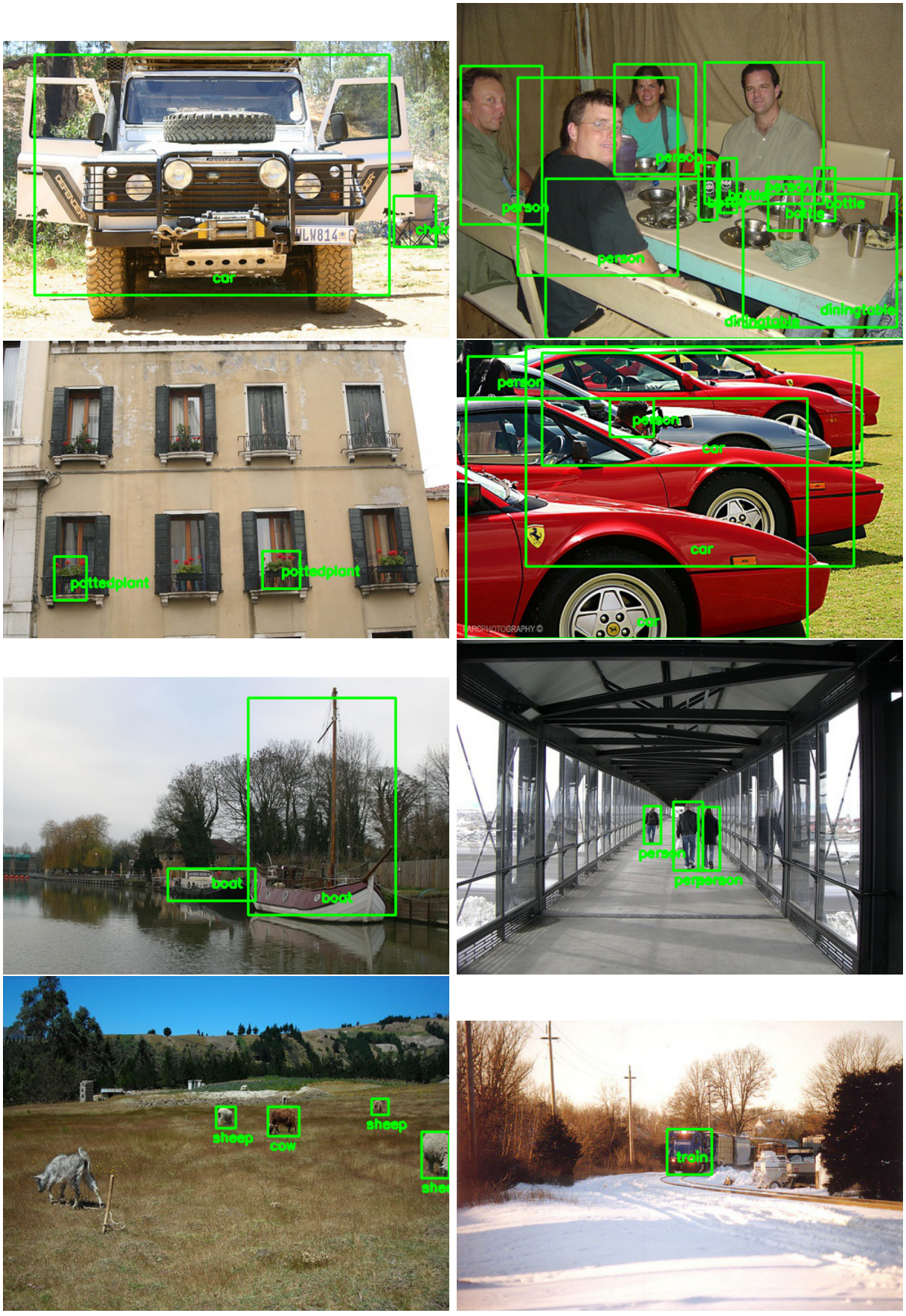

dog


tvmonitor


train
train


cat
sofa
chair


bird


car
person
person
person
person
diningtable
diningtable
pottedplant
pottedplant
person
person
car
car
car
boat
boat
person
person
person
sheep
sheep
cow
train

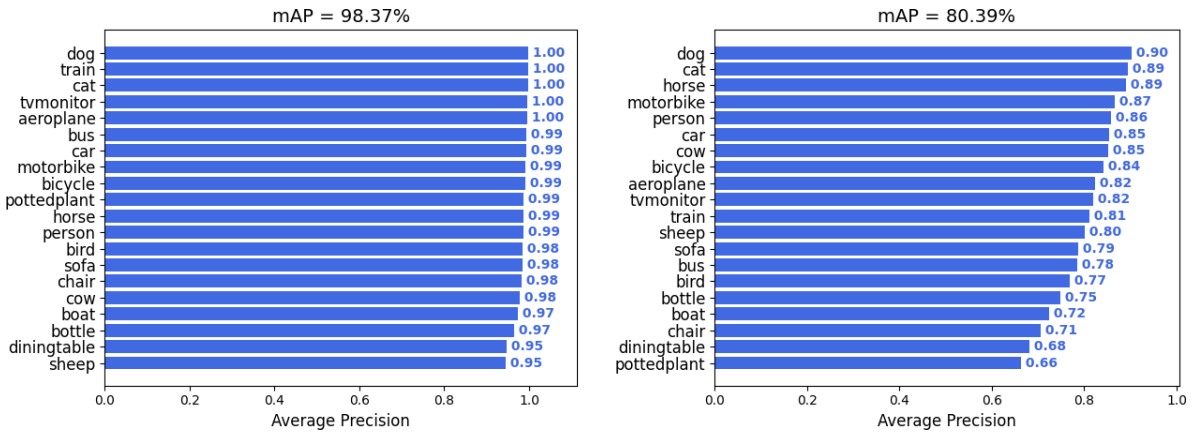


Figure 11: Class-wise average precision for train(left) and test(right) splits on Faster RCNN at IoU = 0.5 threshold

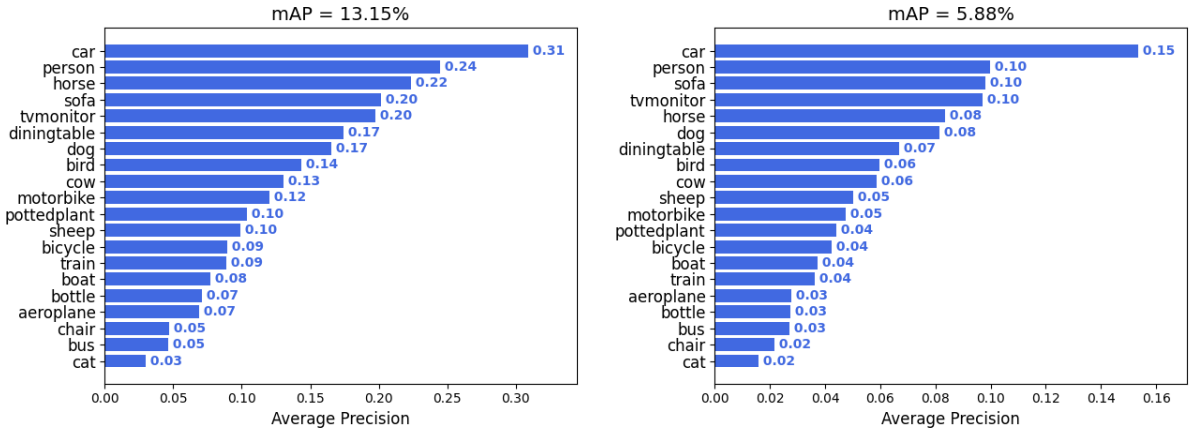


Figure 12: Class-wise average precision for train(left) and test(right) splits on Faster RCNN at IoU = 0.9 threshold

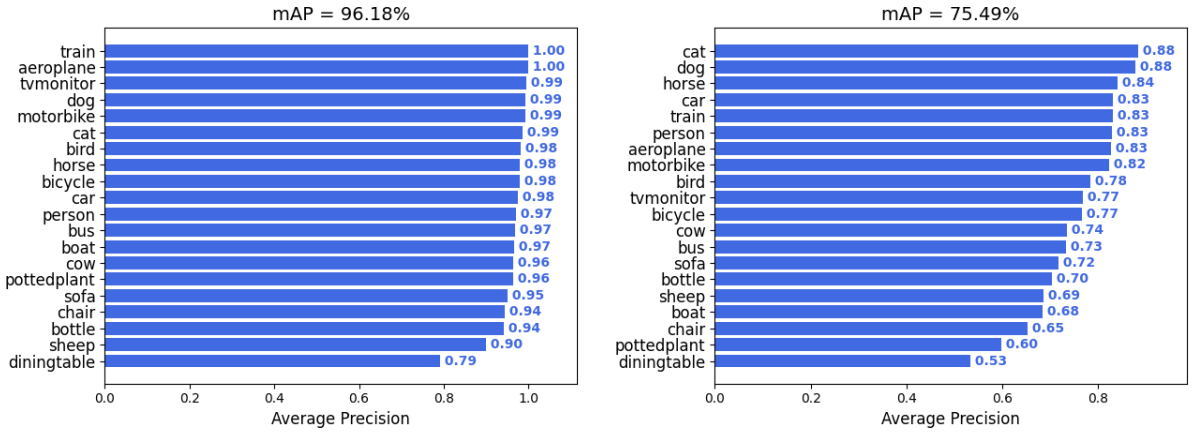


Figure 13: Class-wise average precision for train(left) and test(right) splits on RetinaNet at IoU = 0.5 threshold

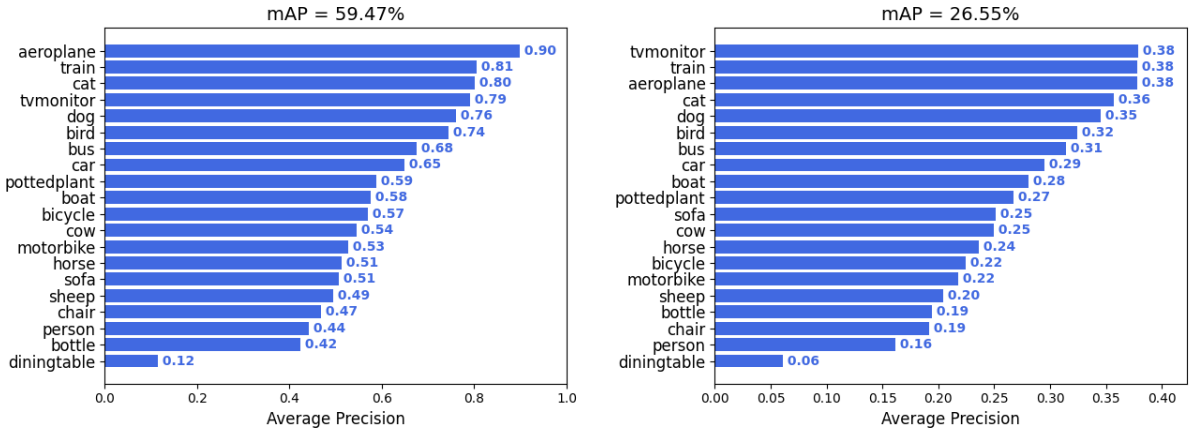


Figure 14: Class-wise average precision for train(left) and test(right) splits on RetinaNet at IoU = 0.9 threshold

References

[1] R. Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[4] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017.

[5] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[6] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 2013.