Comparative Analysis of Object Detection Models: Faster R-CNN, RetinaNet with Focal Loss, and comparison with SSD

DDA4220/MDS6224/MBI6011 Final Project

Lingyi Yang, Yunfei Ke, and Shijie Shao
School of Data Science
Chinese University of Hong Kong, Shenzhen
{120090269, 120090277, 120090491}@cuhk.edu.cn

Abstract

The purpose of this project is to conduct object detection task. Our study explored multiple models of object detection and conducted a comprehensive comparison between three base models, Faster R-CNN, RetinaNet, and SSD. We also investigate the usefulness of focal loss in both one-stage and two-stage object detection, and explore some potential methods for increasing the detection speed while preserving accuracy. We tried Faster RCNN and RetinaNet to discover the effectiveness of focal loss and utilized Single Shot multibox detector(SSD) method to speed up the detection. Our findings show that focal loss is effective in detecting hard samples in two-stage detection and can successfully dealing with unbalanced samples in one-stage detection, and the two methods can not achieve the goal of real time detection while SSD can achieve reaching fps of 43 with only little drop on accuracy.

1 Workload

Yunfei, Shijie, and Lingyi had the same contributions to the project.

2 Introduction

Object detection refers to the task of accurately identifying and localizing objects within an image, which is essential for achieving a comprehensive understanding of image content. In this project, we are required to implement deep learning algorithms for object detection in 11.5k images from PASCAL VOC2012 dataset [1], where 5.7k images are for training and 5.8k images are for testing. The VOC2012 dataset consists of a set of images in 20 categories; each image has an annotation file giving a bounding box and object class label for each object in one of the 20 classes present in the image. In this project, we decided to investigate both the accuracy and speed of object detection methods.

3 Related Work

In the past two decades, the development of object detection can be divided into two paradigms: machine learning based and deep learning based [2], which is shown in Figure 1. The traditional machine learning methods are based on the features extracted from the images (e.g., gradient orientations[3]) and uses classifiers like SVM for the detection. Starting from 2014, deep learning based methods gradually became the popularity. Deep learning based detection methods include two patterns: two-stage detection represented by the R-CNN series (R-CNN, Fast R-CNN, Faster R-CNN), one-stage detection represented by YOLO series, SSD, RetinaNet. Two-stage methods are based on a Region Proposal Network for getting region proposals, and use CNN on the regions

to do classification. Generally two-stage methods are more accurate but slower due to the proposal generation and RoI pooling procedures. Therefore, one-stage methods are proposed for higher speed by end-to-end regression, which tests the images with higher speed but are generally with lower accuracy.

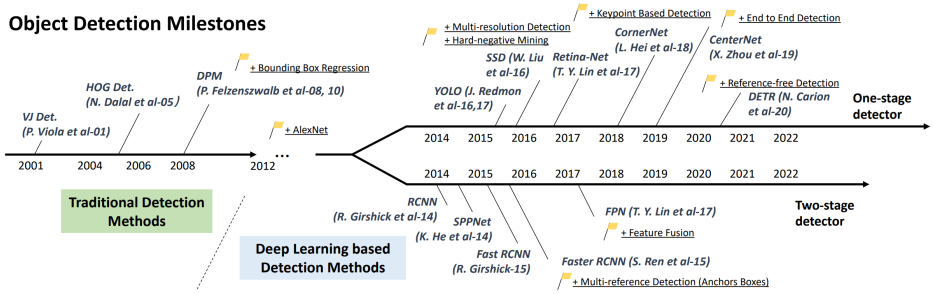

Figure 1: The developing history of object detection[2]

4 Method

4.1 Faster RCNN

4.1.1 Baseline

We decided to utilize the structure of Faster R-CNN [4] as our baseline methods. Faster R-CNN is a popular object detection model which is an extension of is an extension of the earlier R-CNN and Fast R-CNN models. It is a two-stage object detection model that consists of a region proposal network (RPN) and a region-based convolutional neural network (RCNN) classifier, which are trained jointly. The picture 2 is the main structure of our baseline.

4.1.2 Improvement: replace loss function

During the classification stage of the object detection task, imbalanced classes is always a problem. Instead of using traditional cross entropy in the loss of the Faster-RCNN, we decided to use the focal loss[5] that dynamically scales cross entropy loss, where the scaling factor decays to zero as confidence in the correct class increases. In this way, we focus more on hard and misclassified samples, which means harder samples could be detected by us using the focal loss.

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

4.2 RetinaNet

To get a balanced performance in both accuracy and speed, RetinaNet is proposed. RetinaNet is a one-stage detector that adopts ResNet as the backbone, feature pyramid network (FPN) as the neck (shown in Figure 3), and most importantly its loss function is replaced by focal loss. The formula of focal loss in shown in Formula 1. Focal loss prevents the model from pay overwhelming attention

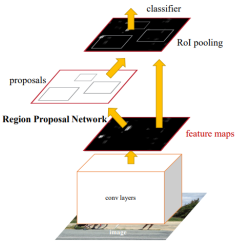

Figure 2: Structure of Faster-RCNN

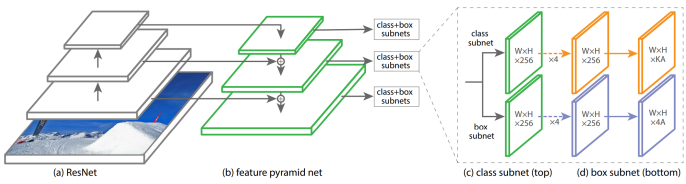

Figure 3: Structure of RetinaNet

on the easily detected samples. Finally, RetinaNet is able to match the speed of previous one-stage detectors while surpassing the accuracy of two-stage detectors like Faster RCNN.

4.3 SSD

SSD (Single Shot MultiBox Detector) is a popular one-stage object detection algorithm that can simultaneously detect multiple objects of different categories in an image, and can perform detection in a single forward pass. SSD was initially proposed by Liu et al [5]. in 2016 with the goal of improving the efficiency and accuracy of object detection algorithms.

The basic idea of the SSD model is to combine deep convolutional neural networks (CNNs) with multiple prior boxes to detect objects at different scales, which adopts vgg16 as the backbone . By applying multiple scales of prior boxes on convolutional feature maps at different levels, SSD can detect objects of different sizes. Two most impotant features of ssd is its high efficiency and model simplicity because of its on-shot-detection method and the relatively small architecture.

5 Experiments

5.1 Evaluation method

The evaluation methods we use to determine our accuracy of our model are average precision and average recall under different settings. Average precision is the area under the Precision-Recall curve and average recall denotes the average of maximum of recall under different IoU. In the following passage, **AP** and **AR** denote the average precision and average recall over all classes. AP/AR @k refers to the AP or AR is calculated in the IoU thresholds k. Specially AP/AR @50:95 refers to the range of IoU thresholds from 0.5 to 0.95 with a step size of 0.05 and then take an average.

The evaluation method we use for detection speed is Frames Per Second (**fps**), which refers to the number of images our model can analyze and identify in per second.

5.2 Experimental details

During the training, we all used the pre-trained model from torchvision and finetue on our own dataset since our traning data only include 5k images. Optimizer, learning rate and epoches are chosen empirically. The detailed parameter settings are listed below.

methods	epochs	optimizer (SGD)			lr_scheduler	
		lr	momentum	weight_decay	step_size	gamma
Faster RCNN-moblie	10	0.005	0.9	0.0005	3	0.1
Faster RCNN-res	10	0.005	0.9	0.0005	3	0.1
Faster RCNN-moblielarge	10	0.005	0.9	0.0005	3	0.1
Faster RCNN-res + FL	10	0.005	0.9	0.0005	3	0.1
SSD	10	0.005	–	0.0005	1	0.9
RetinaNet	20	0.005	0.9	0.0001	15	0.2
RetinaNet + CE	20	0.005	0.9	0.0001	15	0.2

Table 1: Hyperparameter settings

5.2.1 Focal loss v.s Cross Entropy in Faster RCNN

We decided to compare the performance of focal loss with cross entropy loss mentioned in 4.1.2. Pretrained models are utilized from the torchvision. The backbone is chosen as the Resnet-50. The comparison is listed in table 2. Besides that, we also change different backbones of Faster RCNN (including moblie-net, mobile-net-large320 and resnet-50) to discover their performances. The results show that backbone resnet performs the best. The detailed results are in Appendix. The following experiments related to Faster-RCNN are based on resnet-50.

	AP	AP(s)	AP(m)	AP(l)	AR	AR(s)	AR(m)	AR(l)
Faster RCNN with CE	**0.468**	0.176	**0.37**	**0.533**	0.588	0.291	0.495	0.649
Faster RCNN with FL	0.454	**0.185**	0.367	0.515	**0.618**	**0.355**	**0.548**	**0.667**

Table 2: Focal loss v.s Cross Entropy in Faster RCNN based on backbone Resnet-50. The AP and AR are @50:95. s, m, l denote small, medium and large respectively.

	AP	AP(s)	AP(m)	AP(l)	AR	AR(s)	AR(m)	AR(l)
RetinaNet with CE	0.235	0.063	0.134	0.391	**0.601**	**0.285**	**0.460**	0.607
RetinaNet with FL	**0.454**	**0.135**	**0.317**	**0.521**	0.581	0.247	0.458	**0.658**

Table 3: Focal loss v.s Cross Entropy in RetinaNet based on backbone Resnet-50. The AP and AR are @50:95. s, m, l denote small, medium and large respectively.

5.2.2 Focal loss v.s Cross Entropy in RetinaNet

After conducting experiments on differences of focal loss and cross entropy for the one-stage method, we make a comparison for them on the two-stage method — RetinaNet. The result is listed in Table 3

5.3 Results and analysis

After training and fine-tuning the models, we obtain the accuracy and detection speed of these models. Moreover, we found that, during training, the **training speed** of these models is like: $SSD > RetinaNet > Faster - RCNN$, which indicates one of the key advantages of SSD model. Main results are in the table4. The following are the details and analysis.

5.3.1 Use of focal loss

In the experiment, we discovered the use of focal loss in both one-stage and two-stage methods.

In the two-stage Faster RCNN, from the Table2 we can see that compared with the Cross Entropy loss, the use of focal loss increases the recall of our detection which means harder objects can be detected by us although precision decreases. This result is also in accordance with the motivation of focal loss: harder samples could be detected by us using the focal loss. The example image4 below also demonstrates our numerical results.

(a) Detection using Faster RCNN with CE (b) Detection using Faster RCNN with FL

Figure 4: Example of CE v.s FL in Faster RCNN

In the one-stage RetinaNet, we can conclude that focal loss is very important for the unbalanced positive and negative data from the Table3. In terms of the precision, it outperforms cross entropy a lot, which proves the effectiveness of focal loss in one stage method when positive and negative samples are relatively unbalanced.

	AP @50	AP @75	AP @50:95	AP(s) @50:95	AP(m) @50:95	AP(l) @50:95	FPS
Faster-RCNN -resnet50	**0.468**	**0.727**	**0.515**	**0.176**	**0.37**	**0.533**	21
RetinaNet -resnet50	0.446	0.68	0.476	0.135	0.317	0.521	24
SSD -vgg16	0.367	0.613	0.347	0.021	0.193	0.425	**43**

Table 4: Detection precsion and speed on different models. s, m, l denote small, medium and large respectively. Inference speed for different models is tested on GPU RTX A5000

5.3.2 Comparison among different models

As the results in Tabel4, we can see that Faster-RCNN have a relatively high accuracy over other models, as the average precision of Faster-RCNN is about 0.73. However, we observed that during training, the training speed of Faster R-CNN was relatively slow, taking up to 8 minutes per epoch. The detection speed of Faster-RCNN is also the lowest. This is may becuase its two-stage-detection method, which needs to generate region proposals first. Subsequently, we explored two one-stage detection models, RetinaNet and SSD, and found that the SSD model was the fastest with a frame rate of 43 FPS, nearly double the value of Faster R-CNN and RetinaNet. RetinaNet, on the other hand, demonstrated a relatively faster speed than Faster R-CNN, while maintaining a higher accuracy compared to SSD.

In summary, we can conclude that Faster-RCNN achieves the highest average precision owing to its two-stage model and relatively large architecture; SSD achieves the fastest training speed and detection speed due to its model simplicity and one-shot-detection method; and RetinaNet generally offers a trade-off between training speed, detection speed and average precision. The example images below are results of our detection:

(a) Example 1 using Faster RCNN (b) Example 2 using RetinaNet (c) Example 3 using SSD

Figure 5: Example detection results of our methods

6 Conclusion

In conclusion, our study explored multiple models of object detection and conducted a comprehensive comparison between three base models, Faster R-CNN, RetinaNet, and SSD. Our results revealed that Faster R-CNN exhibited the highest average precision, SSD demonstrated the fastest detection and training speed, while RetinaNet offered a balanced trade-off between accuracy and efficiency. Given the strengths and limitations of each model, it remains challenging to determine which model is the best.

Furthermore, we investigate when focal loss is useful in terms of both one-stage detection and two-stage detection in this project . We find that focal loss can detect hard samples in two-stage detection and can deal with unbalanced samples in one-stage detection. Moreover, we also tried one stage SSD method to increase the speed of our detection while preserving the accuracy. However, due to the limited sources and training samples, our method have some gaps with the SOTA performance. In addition, we only focus on the classification loss in this project. More attention could be paid to box regression loss in our future work to discover its performance on object detection.

References

[1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[2] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 2023.

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

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

[5] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

A Appendix

A.1 Performance of Faster RCNN with different backbones

	AP @50	AP @75	AP @50:95	AP(s) @50:95	AP(m) @50:95	AP(l) @50:95
Backbone -mobile	0.462	0.7	0.507	0.112	0.304	**0.552**
Backbone -resnet50	**0.468**	**0.727**	**0.515**	**0.176**	**0.37**	0.533
Backbone -mobile_large320	0.395	0.619	0.426	0.024	0.178	0.513

Table 5: Performance of Faster RCNN with different backbones: Backbone resnet50 is the best one.