Faster R-CNN pre-report

Taehun Kim[1]
[1] Department of Computer Science and Engineering, Pusan National University.

1 Introduction

Region-based CNNs makes good performance(mAP) for object detection, and achieves near real-time rates when ignoring the time spent on computing region proposals. Computing region proposals using CPU like Selective Search[3] or EdgeBoxes[4] is the bottleneck of Region-based CNNs.

Faster R-CNN[2], however, computes proposals with a deep convoluional neural network(Region Proposal Networks, RPNs). RPNs share convolutional layers that extract features from image with object detection network like Fast R-CNN[1]. The paper[2] says this make the cost for computing proposals small(10ms per image).

these advances make Faster R-CNN[2] won the 1st-place entries in the track of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation in ILSVRC and COCO 2015 competitions.

this report summarize the Fast R-CNN paper[2] focusing on the architecture of the RPNs.

2 Faster R-CNN

Faster R-CNN[2] has two modules. The first is Region Proposal Networks, and the second is the Fast R-CNN[1] detector that classify the object from the proposed regions. The entire system is a unified network using attention mechanisms. It means that the RPN tells the detector where to look.

2.1 Region Proposal Networks

A Region Proposal Network can takes an image of any size as input because It is fully convolutional network. Also, this network share convolution layers (that extract features from image) with a Fast R-CNN.

The paper[2] add a small network over the shared feature maps. This small network takes as input a fixed $n \times n$ windows of the outputs of the last shared conv layer. The small network maps windows to a lower dimensional feature(implemented with a $n \times n$ conv layer, which maps $n \times n \times d$ to $1 \times 1 \times d$), and this features is fed into two fully connected layers(implemented with a 1×1 conv layer, which change number of channels). one is a box coordinates regression layer, and another is box scores classification layer. Note that box scores indicates probability that there is an object in the box or not an object in the box.

2.1.1 Anchors

At each sliding-window location, The network simultaneously predict multiple region proposals that has different scale and aspect ratio each other. If k is denoted the number of maximum possible proposals for each sliding windows, then output size of the box coordinates regression layer is $4k$, and the output size of the box scores classification layer is $2k$. The k proposals is generated from the k reference box(*anchors*) that has different size and aspect ratio each other, and centered at the sliding window.

If the convolutional feature map is $H \times W \times C$, this feature map is mapped to $H \times W \times C$ by $n \times n$ conv layer with padding, then the output of box regression layer is $H \times W \times 4k$ and the output of the box classification layer is $H \times W \times 2k$.

these method relies on a single images, feature maps and sliding window. Also it enable that RPNs share features with detector without rescaling.

2.1.2 Loss Function

For training RPNs, We should label box scores and box coordinates to each anchor of each sliding windows. For box scores(that represents whether there is an object or not), the paper[2] assign a positive label to two kind of anchors. One is the anchor(s) with the highest IoU overlap with a ground-truth box, another is an anchor(s) that has an IoU overlap higher than 0.7 with any ground-truth box. First condition is adopted because it is possible that there isn't anchor that has an IoU higher than 0.7 with the ground-truth box. The paper[2] assign a negative label to a anchor that has IoU lower than 0.3 for all ground-truth boxes. Note that only anchor that assigned as positive or negative is used for loss function.

For box coordinate regression, the paper[2] calculate coordinate as follow:

$$t_x = (x - x_a)/w_a, t_y = (y - y_a)/h_a, \tag{1}$$

$$t_w = log(w/w_a), t_h = log(h/h_a), \tag{2}$$

$$t_x^* = (x^* - x_a)/w_a, t_y^* = (y^* - y_a)/h_a, \tag{3}$$

$$t_w^* = log(w^*/w_a), t_h^* = log(h*/h_a) \tag{4}$$

where x,y is center coordinates of the box's center, w and h denotes width and height of the box. Variable $\square, \square_a, \square^*$ are for predicted box, anchor box, and ground-truth box respectively. t is a vector representing the 4 coordinates of the predicted bounding box.

With these definition, Loss function is defined as:

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

i is the index of an anchor, p_i is the predicted probability of anchor i being an object. p_i^* is ground-truth label, 1 for positive and 0 for negative. The p_i^* of the $p_i^* L_{reg}$ means regression loss is applied only for positive anchor because It is useless to applying regression loss for negative anchor.

$N_{cls}, N_{reg}, \lambda$ is the normalizing and balancing parameter for equally weighting the both loss(classification and regression).

In implementation, the anchor boxes that cross image boundaries do not affect the loss. Also, the paper[2] randomly sample 256 anchors with 128 positive and 128 negative to compute the loss function. this is because the number of negative anchors is much more than positive ones.

2.2 Sharing Features for RPN and detector

For sharing conv layers between RPN and detector, the paper[2] uses *Alternating training*. this method has 4-step:

1. training the RPN initalized with an ImageNet pre-trained model.
2. training the separate detection network using the proposals generated by step 1. this detection network is also pre-trained with Imagenet.
3. the paper[2] use the convolution layer(that extract features from image) of detector network to fine-tune the layers unique to RPN with fixed the shared conv layers.
4. the paper[2] fine-tune the layers unique to detection network with fixed the shared conv layers.

3 conclusion

By getting region proposals from convolution network called RPN that sharing convolution layers with detector, the region proposal step is nearly cost-free and improve the proposals quality. Therefore, Fast R-CNN is not only more accurate but also faster than previous object detection network that uses Selective Search[3] or EdgeBoxes[4] to get region proposals.

[1] Ross Girshick. Fast r-cnn, 2015. URL https://arxiv.org/abs/1504.08083.

[2] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016. URL https://arxiv.org/abs/1506.01497.

[3] Jasper Uijlings, K. Sande, T. Gevers, and A.W.M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104:154–171, 09 2013. doi: 10.1007/s11263-013-0620-5.

[4] Charles Zitnick and Piotr Dollar. Edge boxes : Locating object proposals from edges. volume 8693, 09 2014. ISBN 978-3-319-10601-4. doi: 10.1007/978-3-319-10602-1_26.