# Faster R-CNN Summary

Roger Marí [1], Joan Sintes [2], Àlex Palomo [3], Àlex Vicente [4]
Universitat Pompeu Fabra, Image Processing and Computer Vision Group
{ [1]roger.mari01, [2]joan.sintes01, [3]alex.palomo01, [4]alex.vicente01 }@estudiant.upf.edu

***Abstract*—This is a summary of the original paper by Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun [1].**

## I. Introduction

The authors of Faster R-CNN, propose a faster version of the R-CNN that reduces the running time of the detection networks, by exposing region proposal computation as a bottleneck. For this purpose they introduce a Region Proposal Network (RPN) that shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. By mergin RPN and Fast R-CNN into a single network by sharing their convolutional features, the model achieve a frame rate of 5 fps.

## II. The model

The object detection system proposed, called Faster R-CNN (Fig. 1), is composed of two modules. The first one is a deep fully convolutional network that proposes regions, and the second module is the Fast R-CNN network that uses the proposed regions. The RPN module tells the Fast R-CNN module where to look, using attention mechanisms.

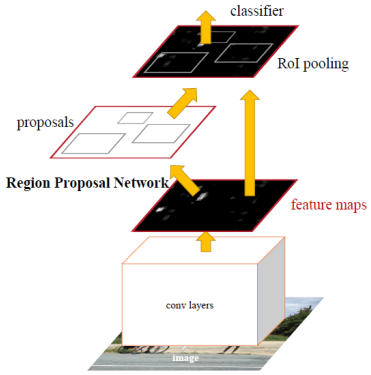


Fig. 1: Faster R-CNN is a single, unified network for object detection.

### A. Region Proposal Networks

An RPN (Fig. 2) is a fully convolutional network that takes an image (of any size) as input and simultaneously predicts object bounds and objectness scores at each position. The RPN is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. The architecture of RPNs follows a fully convolutional network approach. The first part of the RPNs are convolutional layers shared with the Fast R-CNN. In the second part, a small convolutional network is slid over the feature map output by the last shared convolutional layer. The network is fully connected to an N x N spatial window of the feature map. Each sliding window is mapped to a lower dimensional vector of 256-d (for 5 shared convolutional layers (Zeiler and Fergus model)), or 512-d (for 13 shared layers (Simonyan and Zisserman model)). The vector is fed to two 1 x 1 convolutional fully-connected layers: a box-regression layer (reg) and a box-classification layer (cls).

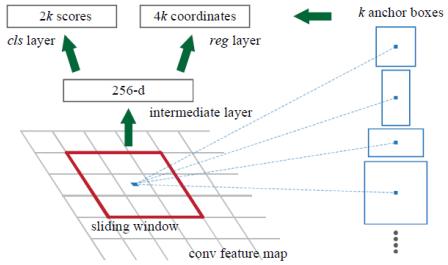


Fig. 2: Region Proposal Network procedure.

At each sliding window location, K region proposals are predicted. The reg layer has 4K outputs, encoding the coordinates of the K boxes). The cls layer has 2K outputs, encoding the probabilities of object/non-object for each box. Then, 3 different scales and 3 different aspect ratios are used, resulting in K = 9 possible boxes evaluated at each window location. Therefore, for a feature map of W * H there are W * K * K boxes. The k proposals are parameterized relative to k reference boxes, which are called *anchors*.

### B. Training an RPN

A binary class label (object or not) is assigned to each anchor. Positive label to anchors with the highest Intersection-over-Union (IoU) overlap with a ground-truth box all anchors with IoU overlap higher than 0.7 with any ground-truth box.On the other hand, negative label to anchors with IoU lower than 0.3 for all ground-truth boxes. Anchors that are not positive neither negative do not contribute to training. Finally, with this constraints, they minimize an objective function following the multi-task loss in Fast R-CNN. The loss function for an image is defined as:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_cls}\sum_i L_cls(p_i,p_i^*) + \lambda\frac{1}{N_reg}\sum_i p_i^* L_reg(t_i,t_i^*) \quad (1)$$

, where *Lcls* is the classification loss (log loss over two classes) and * denotes ground-truth, *Lreg* is the regression loss as defined for Fast R-CNN (only for positive anchors), *Ncls* and

*Nreg* are the normalization parameters and *lambda* is a weight parameter. Finally, stochastic gradient descent based on back-propagation is used for optimization. All weights of new layers are initialized from zero-mean Gaussian distribution with standard deviation 0.01. All other layers (shared convolutional layers) are initialized with the pretrained ImageNet weights. Learning rate 0.001. Momentum 0.9. Weight decay 0.0005.

### *C. Sharing Features for RPN and Fast R-CNN*

A training technique that allows to learn the two goals (region proposal and object detection) is needed. For this purpose they used the called "alternating method". The first step is to train the RPN as described before. The second step is to train a separate detection network Fast R-CNN using the proposals generated by step 1. The third step is to use the detection network to initialize the shared layers of the RPN and fine tune the rest of layers. And the step four is to keep shared layers fixed and fine tune the rest of layers of the Fast R-CNN. Steps 3 and 4 are repeated until convergence.

## III. Conclusion

In this paper, the authors present RPNs for efficient and accurate region proposal generation. The method enables a unified, deep-learning-based object detection system to run at near real-time frame rates. The learned RPN also improves region proposal quality and thus the overall object detection accuracy.

## References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.