Decouple ROI pooling for object detection
CS 260 Final Project Report

Pengrui Quan (805227042) Weixi Feng (105220789) Miaoqing Chen (505219312)
Department of Electrical and Computer Engineering
University of California, Los Angeles

March 20, 2019

1 Introduction

In this project, we mainly focus on region-based object detection [1] frameworks which are composed of network proposal module on top of feature extraction as well as classification and localization module based on the proposals.

Based on the work of current region-based CNN detection, we introduce an approach to enhance the adaptive receptive capability of CNN, namely decouple ROI pooling. The module can enable the network to select the best receptive field to improve its classification accuracy without sacrificing its ability of bounding box regression.

Our experiments shows that the method can benefit object classification module of the network with our designed structure. However, we also observed that even though decouple operation can render better receptive field, the overall performance of the deep CNN (including both classification and bounding box regression) is dropped by 0.3% on Pascal VOC dataset [4] due to the separate branches for classification and localization, and this method will be under further investigation.

2 Related work

2.1 Object detection

The Recent CNN based object detection framework can be generally separated into two categories: single-stage [2] and two-stage detector [1]. One of the most widely used two stage framework is Faster RCNN [1]. The Region Proposal Network (RPN), can not only provide more accurate region of interest (ROI), but also wipe out most of the negative samples corresponding to the inaccurate localization or background candidate in the classification and regression operations, which eliminates the trend that the network will bias to negative samples.

2.2 Adaptive receptive field

Deformable ConvNet (DCN) [3] tries to incorporate deformable convolution and deformable ROI pooling that use learned offsets to adjust convolution and sampling bin operations. However, even though the module can learn a deformable geometry of an object, the operations still cannot adapt to the scale of a ROI.

3 Decouple ROI pooling

3.1 Problems with Faster RCNN

After investigating on the Faster RCNN structure, we conjecture that there are still defects in it:

- **Feature sharing between classification and localization** This approach is suboptimal due to mismatch characteristic of the two task: Classification is transition invariant, with the pretrained backbone ResNet trained with flipped, rotated and randomly cropped images on ImageNet dataset, while regression is designed to be transition sensitive to estimate offset to ground truth location.

- **Fixed receptive field** In the Faster RCNN, the ROI is estimated by RPN and its corresponding feature is cropped to a fixed size representation for the following classification, which leads to the problem of fixed receptive field. Since the ROI is predicted by RPN, once the ROI is too large, the network might be unable to focus on the object, while if the ROI is too small, the network can only look at part of the object, under the risk of missing important information.

3.2 Naïve decouple ROI pooling

The feature corresponding to the ROI is cropped into a fixed size ($C \times 7 \times 7$). We therefore design a Gaussian kernel for feature pooling. The idea is that the bin of the cropped feature is exponentially decreased w.r.t. the distance to the central position of the kernel.

$$I(u,v) = \sum_{(x,y)\in ROI} I(u+x, v+y) f(x,y)$$

$$f(x,y) = e^{-(\frac{(\mu_x - x)^2}{\delta_x^2} + \frac{(\mu_y - y)^2}{\delta_y^2})}$$

where μ_x, μ_y are the central of the ROI. However, this kind of method is subjected to heavy computation and usually hundreds of time slower: each grid in pooled feature of a ROI needs an element-wise product and most of the computation outside the central position are redundant.

3.3 Optimized decouple ROI pooling

We therefore propose another decouple ROI pooling method with higher speed and much less computation:

- $A \times S$ predefined Gaussian filter with size $C \times H \times W$ is applied to feature map, producing $A \times S$ distinct feature map. Each filter corresponds to a specific aspect ratio and scale of a predefined anchor box in Faster RCNN architecture.

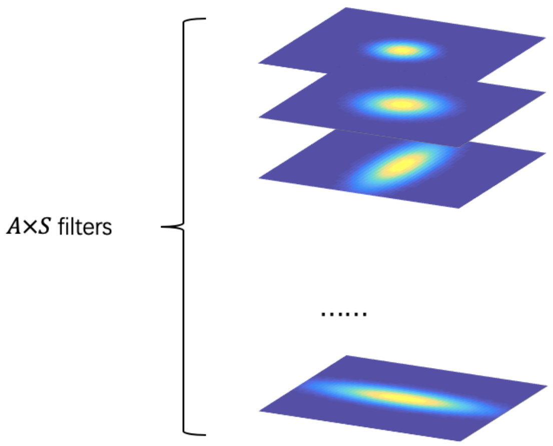

Figure 1: Gaussian filters

- The element of each filter is computed by

$$f(x,y) = e^{-(\frac{(\mu_x - x)^2}{\gamma_x \delta_x^2} + \frac{(\mu_y - y)^2}{\gamma_y \delta_y^2})}$$

  where δ_x and δ_y are proportional to the width and length of anchor boxes respectively, μ_x, μ_y are the central of the filters, and γ_x, γ_y are scaling hyper parameters.

- After getting $A \times S$ feature map, a trainable $1 \times (A \times S)$ indicator will select which of the feature map is the most informative and ultimately combine them into a ROI feature.

3.4 Overall detection framework

Since the decouple ROI pooling is designed for adaptive receptive field, which is inconsistent with the regression branch, we also add a separate branch solely for CNN classification.

3.5 Other trials

To deal with the fixed receptive field problem, we have also tried out some other simple modification based on Faster RCNN. We applied filters with different sizes after ROI pooling to learn regional information from broader ranges compared to original 3×3 filter. We also

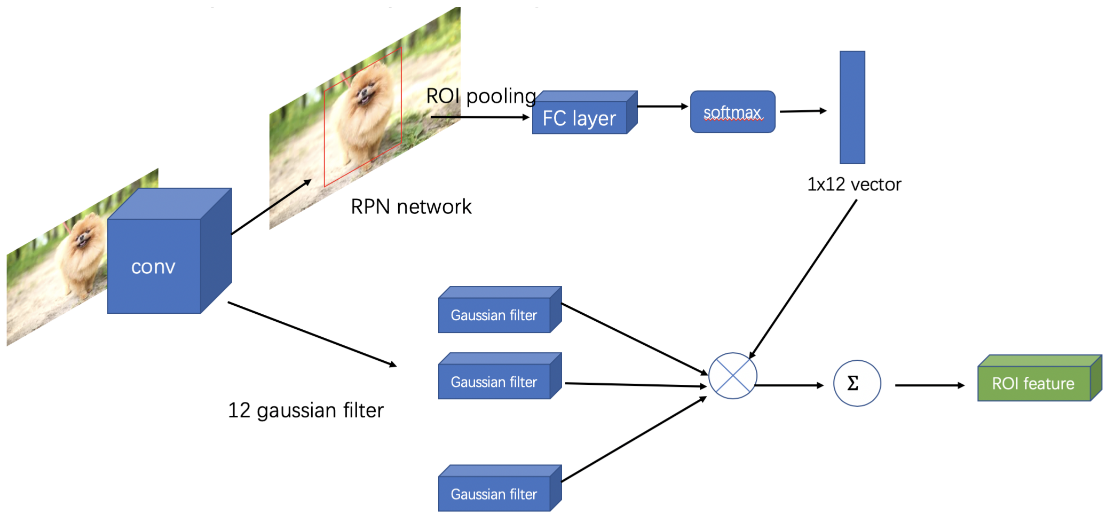

Figure 2: decouple ROI pooling module

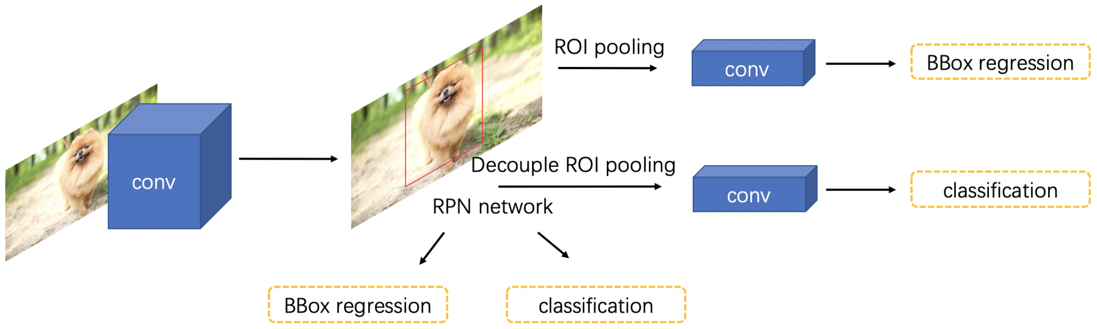

Figure 3: overall detection framework

modified the loss function to bias it towards objects that has extreme shapes. Objects like airplanes can be extremely narrow in height and long in width, looking much different from any of the anchor boxes. Boundary information of these objects are likely to get lost. Therefore we modified the original regression loss as $\tilde{L}_{reg}(t_i, t_i^*) = \frac{w(t_h^*)+w(t_w^*)}{2} L_{reg}(t_i, t_i^*)$ where weight $w(t_i^*) = 1 + |t_i^*|$. t_i, t_i^* follows the definition from [1] and L_{reg} uses the smooth-L1 loss.

4 Experiment results

Experiments are conducted on 2 NVIDIA GeForce GTX 1080 Ti graph cards, with backbone net ResNet18, 30 epoch, 0.0005 weight decay, learning rate decaying 0.1 per 10 epoch, 0.01 learning rate, and SGD optimizer with 0.09 momentum. Models are trained on Pascal VOC [4] 2007+2012 train+val dataset (15k images) and evaluated on Pascal VOC 2007 test dataset (5k images). Codes are available on github `https://github.com/quanpr/decouple-roi-pooling`.

We evaluate our network under different network structure and various γ_x, γ_y through experiments:

Models	**Accuracy (mAP)**	**Time per epochs (hr)**
Baseline	**73.0%**	0.67
Separate branches	72.4%	0.69
Separate branches + naïve decouple ROI pooling	-	59
Same branch + Optimized decouple ROI pooling	70.2%	0.68
Separate branches + Optimized decouple ROI pooling	**72.7%**	0.72
Baseline*	69.6%	0.67
Multiple kernels + Biased loss*	68.8%	0.67

Table 1: Performance of different network structure(* with VGG16 backbone, others with ResNet18)

γ_x	γ_y	**Accuracy (mAP)**
1	1	**72.7%**
0.5	0.5	72.6%
0.25	0.25	72.6%

Table 2: Performance of Separate branches + Optimized decouple ROI pooling with respect to γ_x, γ_y

5 Experiment Analysis

From the experiments, we can have following observations and discussions:

- Our decouple pooling method can improve the classification accuracy by 0.3% comparing the results from separate branches model and separate branches + decouple ROI pooling.
- Without using the separate branches, network performance will drop dramatically using decouple ROI pooling, which agrees with our assumption that the Gaussian kernel will incorporate background information and blur the original ROI boundary.
- The overall performance decrease 0.3% simply because using separate branches for classification and localization. As is shown in the experiment, the accuracy of using separate branches only drop 0.6%, which is too large for our improvement to alleviate.

- The network performance is insensitive to the scaling hyper parameter γ_x and γ_y.

References

[1] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[C]//Advances in neural information processing systems. 2015: 91-99.

[2] Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector[C]//European conference on computer vision. Springer, Cham, 2016: 21-37.

[3] Dai J, Qi H, Xiong Y, et al. Deformable convolutional networks[C]//Proceedings of the IEEE international conference on computer vision. 2017: 764-773.

[4] Everingham M, Van Gool L, Williams C K I, et al. The pascal visual object classes (voc) challenge[J]. International journal of computer vision, 2010, 88(2): 303-338.