Faster R-CNN
<<Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks>>

摘要

RPN:Region Proposal Network （ mimick中也有 ）：跟检测网络共享整个图片的卷积特征：预测物体的边界和在每个位置的得分：end-to-end产生高质量的候选域
把RPN和Fast-RCNN和并成一个网络，加入attention机制

1.介绍

fast R-CNN忽略了在候选域产生上的时间消耗,是fast RNN的时间瓶颈 候选域的产生花费大量的时间
selective search:每张图片2秒
EdgeBoxes:每张图片0.2秒，候选域质量差
R-CNN候选域产生是在CPU上实现的，而不是利用GPU
用深度网络计算候选域，在已知监测网络的计算时，计算候选域几乎是免费的
RPN和检测网络共享卷积层
卷积feature map可以被用于检测器，也可以用来产生候选域
在卷积特征后面，加上RPN(一次额外的卷积层)去回归边界和给物体在每个位置上打分用规则网格*（？？？？什么是规则网格）*
RPN是全卷积层（ FCN ），可以被end-to-end 训练
anchor box:多尺度和多纵横比，避免枚举图片或滤波器 ，这种方法当使用单尺度的图片训练和测试时性能很好，速度也快

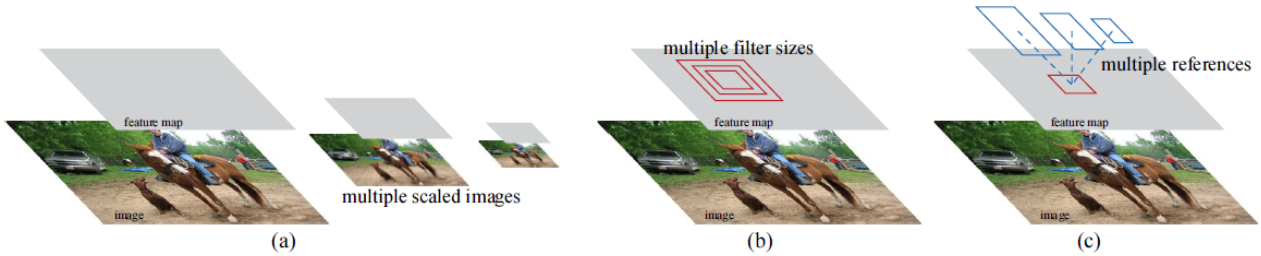

Figure 1: Different schemes for addressing multiple scales and sizes. (a) Pyramids of images and feature maps are built, and the classifier is run at all scales. (b) Pyramids of filters with multiple scales/sizes are run on the feature map. (c) We use pyramids of reference boxes in the regression functions.

训练：保持候选不变，在微调候选域任务和检测任务之间交替进行
代码位置：https://github.com/rbgirshick/py-faster-rcnn
速度快，而且提高了物体检测的精度

2.相关工作

物体候选产生方法：
基于grouping super-pixels：Selective Search,CPMC,MCG
基于滑动窗口：objectness in windows,EdgeBoxes
物体候选产生是一个独立的模块
用于检测的深度网络：
R-CNN只是一个分类器，不能预测物体边界（除了使用bounding-box 回归） OverFeat:训练了一个全连接层去

预测物体的位置
全连接层被转化成卷基层去检测多个类别的物体
MultiBox:不能在候选域和检测网络中共享特征
共享卷积计算

3.Faster R-CNN

由两部分组成：产生候选域的深度卷积网络，用于检测的Fast R-CNN
attention机制：RPN告诉Fast R-CNN去看那里

候选域产生网络RPN

输入：一张图片
输出：一系列矩形候选域和是物体类的得分
ZF有五个共享卷基层
VGG16有13个共享卷基层
在卷积feature map上滑过一个小网络
小网络的输入为：$n \times n$的空间feature map n=3
每个滑过的窗口的feature map被映射到一个低维的特征（ZF:256,VGG:512）
这些特征被输入到bounding-box回归的全连接层（reg）和box分类的全连接层(cls)
（？？？我认为回归和分类用到的特征应该是不同的，输入相同的特征不好吧）

Anchors

在每个滑动窗口位置同时预测多个候选域，可能的候选域的最大数目为k
所以reg层有4k个输出 ,cls层有2k个，表示是物体的概率和不是物体的概率 *（？？？为什么不是k,是物体的概率和不是物体的概率之和不为1吗？）*
k个box参数化后成为k个候选框叫做anchors
anchor是滑动窗的中心，并且和尺度，纵横比有关
3个尺度和3个纵横比，k=9

平移不变anchors
平移不变性减少了模型的大小，而且减少了overfiting的风险
多尺度anchors作为回归参考
两种流行的多尺度预测的方法：
1. 图像或特征金字塔，图像的大小调整成多个尺度，计算每个尺度的特征，时间开销大
2. 在feature map上使用多尺度的滑动窗口，滤波器的多尺度

这两种方法经常联合起来使用
这篇文章采用的方法是anchors金字塔

损失函数

给anchor一个正标签：
1. 跟一个真实box的IoU是最高的

2. 跟任意一个真实box的IoU大于0.7

一个真实的box可能会分配正标签给多个anchor
采用第一个条件的原因是：一些稀有的情况下，用第二种条件找不到正样本
指派负标签给anchor:
跟所有真实box的IoU都小于0.3
其他的不是正也不是负的anchor不参与训练
(????这样是不是把一些比较难的训练样本没有加入进去，为什么要这样做？)
跟Fast-RCNN一样使用多任务损失的目标函数：

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

i:mini-batch中 anchor的索引
p_i:预测到的第i个anchor是物体的概率
p_i^*:第i个anchor的真是标签是正则为1，否则为0
t_i:4维的向量，表示预测到的bounding-box的位置
t_i^*:正标签的anchor对应的真实box的位置
L_{cls}:两类的分类损失，是物体和不是物体，log损失
\(L*{reg}=R(t_i -t_i^*)\),R:光滑的L1损失，跟fast rcnn中定义的一样
cls层输出p_i,reg层输出t_i
\(N {cls},N_ {reg}\)是用来正则化的，$N_{cls} = 256, N_{reg} = 2400$，$\lambda = 10$
后面会讲简化的正则化
四个坐标的参数化：

$$\begin{aligned} t_{\mathrm{x}} &= (x - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}} = (y - y_{\mathrm{a}})/h_{\mathrm{a}}, \\ t_{\mathrm{w}} &= \log(w/w_{\mathrm{a}}), \quad t_{\mathrm{h}} = \log(h/h_{\mathrm{a}}), \\ t_{\mathrm{x}}^* &= (x^* - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}}^* = (y^* - y_{\mathrm{a}})/h_{\mathrm{a}}, \\ t_{\mathrm{w}}^* &= \log(w^*/w_{\mathrm{a}}), \quad t_{\mathrm{h}}^* = \log(h^*/h_{\mathrm{a}}), \end{aligned} \quad (2)$$

x:预测的box
x_a:anchor的box
x^*:真实的box
从anchor box到真实box的回归
跟SSP ,Fast RCNN不同的地方：bounding box 回归 用于回归的feature map的大小是相同的3×3
为了考虑到不同大小，需要学习k个bounding-box回归器
每个回归器为一个尺度和纵横比负责，
k个回归器不共享权重

训练RPN

RPN可以用反向传播和随机梯度下降，end-to-end训练出来
image-centria采样策略？
每个min-batch来自一张图片的包含正负样本的anchor
因为负样本多，所以优化损失函数时可能会偏向负样本
每张图片次啊样256个anchor，正负样本数近似相等

RPN和Fast R-CNN共享特征

学习一个由RPN和Fast R-CNN构成的统一的一个网络：
1. 交替训练：
- 训练RPN,用imagenet预训练的模型初始化，end-to-end微调做候选框产生任务
- 用RPN产生的候选框训练Fast R-CNN作为检测网络,也是用imagenet与训练网络做初始化
- 然后用Fast R-CNN监测网络初始化RPN，固定共享的卷积层，微调只跟RPN有关的层 - 固定共享卷积层，微调只跟Fast RCNN有关的层
2. 近似联合训练：
后向传播把RPN和Fast RCNN的损失结合在一起，忽略了候选框坐标 ，得到跟上面近似的结果，但训练时间减少25-50%
3. 非近似联合训练：
需要RoI池化层，可以区分候选框的坐标 RoI warping 层
（？？？？这种方法的效果怎样，应该会比较好吧？）

实现细节

训练RPN和检测网络都使用的是单尺度，短边s=600像素
在调整图片大小时，在ZF和VGG网络最后一个卷积层上的跨度是16像素，这么大的跨度效果也比较好，通过减小跨度，可以进一步提高精确率
anchors使用3个尺度3个纵横比，没有根据特定数据集仔细选择
预测的候选框可以比底层的感受野更大，原因是物体可能会有扩展，但只有中心部分在图片中是可见的
anchor box跟图片边界相交的情况：训练阶段，忽略这些anchor，最后每张图片大概有6000个anchor参与训练，测试阶段修剪anchor边界到图片边界
对候选框基于他们的cls得分采用NMS（非极大值抑制），去掉IoU大于0.7的候选域，最后每张图片剩余2000个候选域，之后使用得分为前N的候选域用于检测
训练Fast RCNN使用2000个RPN候选框，测试阶段使用不同的数字

4.实验

在VOC上的实验

用到的数据集：
VOC2007数据集：5k个训练数据，5k个测试数据，20个物体类别
VOC2012数据集：
用到的网络结构：
fast ZF网络：5个卷积层，3个全连接层
VGG-16:13个卷积层，3个全连接层 实验不同的候选域产生方法对mAP的影响：

Table 2: Detection results on **PASCAL VOC 2007 test set** (trained on VOC 2007 trainval). The detectors are Fast R-CNN with ZF, but using various proposal methods for training and testing.

train-time region proposals		test-time region proposals		
method	# boxes	method	# proposals	mAP (%)
SS	2000	SS	2000	58.7
EB	2000	EB	2000	58.6
RPN+ZF, shared	2000	RPN+ZF, shared	300	**59.9**
ablation experiments follow below				
RPN+ZF, unshared	2000	RPN+ZF, unshared	300	58.7
SS	2000	RPN+ZF	100	55.1
SS	2000	RPN+ZF	300	56.8
SS	2000	RPN+ZF	1000	56.3
SS	2000	RPN+ZF (no NMS)	6000	55.2
SS	2000	RPN+ZF (no *cls*)	100	44.6
SS	2000	RPN+ZF (no *cls*)	300	51.4
SS	2000	RPN+ZF (no *cls*)	1000	55.8
SS	2000	RPN+ZF (no *reg*)	300	52.1
SS	2000	RPN+ZF (no *reg*)	1000	51.3
SS	2000	RPN+VGG	300	59.2

RPN+ZF 共享卷积层的方法产生的候选域最好

RPN和Fast RCNN共享卷积层的影响：4步训练阶段中去掉后两个阶段

RPN对训练Fast RCNN的影响

分别评估了cls和reg层对mAP的影响

评估了使用VGG16去训练RPN产生的候选框质量更好，mAP从56.8%提升到59.2%

候选框网络和检测网络都用VGG16在VOC07上的实验结果：

Table 3: Detection results on **PASCAL VOC 2007 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07+12": union set of VOC 2007 trainval and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 2000. †: this number was reported in [2]; using the repository provided by this paper, this result is higher (68.1).

method	# proposals	data	mAP (%)
SS	2000	07	66.9†
SS	2000	07+12	70.0
RPN+VGG, unshared	300	07	68.5
RPN+VGG, shared	300	07	69.9
RPN+VGG, shared	300	07+12	**73.2**
RPN+VGG, shared	300	COCO+07+12	**78.8**

候选框网络和检测网络都用VGG16在VOC2012上的实验结果：

Table 4: Detection results on **PASCAL VOC 2012 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07++12": union set of VOC 2007 trainval+test and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 2000. †: http://host.robots.ox.ac.uk:8080/anonymous/HZJTQA.html. ‡: http://host.robots.ox.ac.uk:8080/anonymous/YNPLXB.html. §: http://host.robots.ox.ac.uk:8080/anonymous/XEDH10.html.

method	# proposals	data	mAP (%)
SS	2000	12	65.7
SS	2000	07++12	68.4
RPN+VGG, shared†	300	12	67.0
RPN+VGG, shared‡	300	07++12	**70.4**
RPN+VGG, shared§	300	COCO+07++12	**75.9**

测试时间：

Table 5: **Timing** (ms) on a K40 GPU, except SS proposal is evaluated in a CPU. "Region-wise" includes NMS, pooling, fully-connected, and softmax layers. See our released code for the profiling of running time.

model	system	conv	proposal	region-wise	total	rate
VGG	SS + Fast R-CNN	146	1510	174	1830	0.5 fps
VGG	RPN + Fast R-CNN	141	**10**	47	198	**5 fps**
ZF	RPN + Fast R-CNN	31	3	25	**59**	**17 fps**