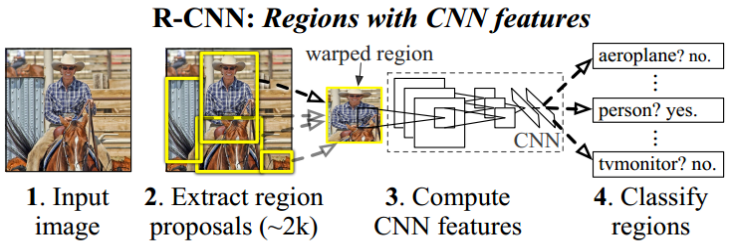
R-CNN,SPP-NET, Fast-R-CNN,Faster-R-CNN, YOLO, SSD系列深度学习检测方法梳理

注：1.本博文持续更新中，文章较长，可以收藏方便下次阅读。2.本人原创，谢绝转载。

1. R-CNN：Rich feature hierarchies for accurate object detection and semantic segmentation

技术路线：selective search + CNN + SVMs



Step1:候选框提取***(selective search)***

训练：给定一张图片，利用seletive search方法从中提取出2000个候选框。由于候选框大小不一，考虑到后续CNN要求输入的图片大小统一，将2000个候选框全部resize到227*227分辨率（为了避免图像扭曲严重，中间可以采取一些技巧减少图像扭曲）。

测试：给定一张图片，利用seletive search方法从中提取出2000个候选框。由于候选框大小不一，考虑到后续CNN要求输入的图片大小统一，将2000个候选框全部resize到227*227分辨率（为了避免图像扭曲严重，中间可以采取一些技巧减少图像扭曲）。

Step2:特征提取***(CNN)***

训练：提取特征的CNN模型需要预先训练得到。训练CNN模型时，对训练数据标定要求比较宽松，即SS方法提取的proposal只包含部分目标区域时，我们也将该proposal标定为特定物体类别。这样做的主要原因在于，CNN训练需要大规模的数据，如果标定要求极其严格（即只有完全包含目标区域且不属于目标的区域不能超过一个小的阈值），那么用于CNN训练的样本数量会很少。因此，宽松标定条件下训练得到的CNN模型只能用于特征提取。

测试：得到统一分辨率227*227的proposal后，带入训练得到的CNN模型，最后一个全连接层的输出结果---4096*1维度向量即用于最终测试的特征。

Step3:分类器***(SVMs)***

训练：对于所有proposal进行严格的标定（可以这样理解，当且仅当一个候选框完全包含ground truth区域且不属于ground truth部分不超过e.g,候选框区域的5%时认为该候选框标定结果为目标，否则位背景），然后将所有proposal经过CNN处理得到的特征和SVM新标定结果输入到SVMs分类器进行训练得到分类器预测模型。

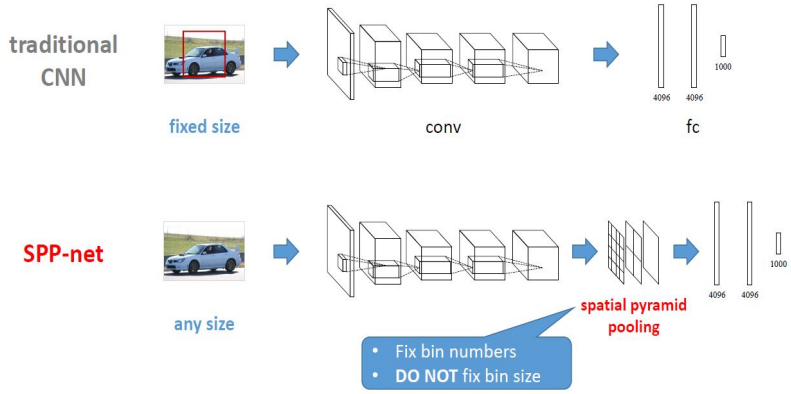
测试：对于一副测试图像，提取得到的2000个proposal经过CNN特征提取后输入到SVM分类器预测模型中，可以给出特定类别评分结果。

结果生成：得到SVMs对于所有Proposal的评分结果，将一些分数较低的proposal去掉后，剩下的proposal中会出现候选框相交的情况。采用非极大值抑制技术，对于相交的两个框或若干个框，找到最能代表最终检测结果的候选框（非极大值抑制方法可以参考：http://blog.csdn.net/pb09013037/article/details/45477591）

R-CNN需要对SS提取得到的每个proposal进行一次前向CNN实现特征提取，因此计算量很大，无法实时。此外，由于全连接层的存在，需要严格保证输入的proposal最终resize到相同尺度大小，这在一定程度造成图像畸变，影响最终结果。

2. SPP-Net : Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition）

传统CNN和SPP-Net流程对比如下图所示（引自http://www.image-net.org/challenges/LSVRC/2014/slides/sppnet_ilsvrc2014.pdf）



SPP-net具有以下特点：

1.传统CNN网络中，卷积层对输入图像大小不作特别要求，但全连接层要求输入图像具有统一尺寸大小。因此，在R-CNN中，对于selective search方法提出的不同大小的proposal需要先通过Crop操作或Wrap操作将proposal区域裁剪为统一大小，然后用CNN提取proposal特征。相比之下，SPP-net在最后一个卷积层与其后的全连接层之间添加了一个SPP (spatial pyramid pooling) layer，从而避免对propsal进行Crop或Warp操作。总而言之，SPP-layer适用于不同尺寸的输入图像，通过SPP-layer对最后一个卷积层特征进行pool操作并产生固定大小feature map,进而匹配后续的全连接层。

2.由于SPP-net支持不同尺寸输入图像，因此SPP-net提取得到的图像特征具有更好的尺度不变性，降低了训练过程中的过拟合可能性。

3.R-CNN在训练和测试是需要对每一个图像中每一个proposal进行一遍CNN前向特征提取，如果是2000个propsal,需要2000次前向CNN特征提取。但SPP-net只需要进行一次前向CNN特征提取，即对整图进行CNN特征提取，得到最后一个卷积层的feature map，然后采用SPP-layer根据空间对应关系得到相应proposal的特征。SPP-net速度可以比R-CNN速度快24~102倍，且准确率比R-CNN更高（下图引自SPP-net原作论文，可以看到SPP-net中spp-layer前有5个卷积层，第5个卷积层的输出特征在位置上可以对应到原来的图像，例如第一个图中左下角车轮在其conv5的图中显示为"^"的激活区域，因此基于此特性，SPP-net只需要对整图进行一遍前向卷积，在得到的conv5特征后，然后用SPP-net分别提取相应proposal的特征）。

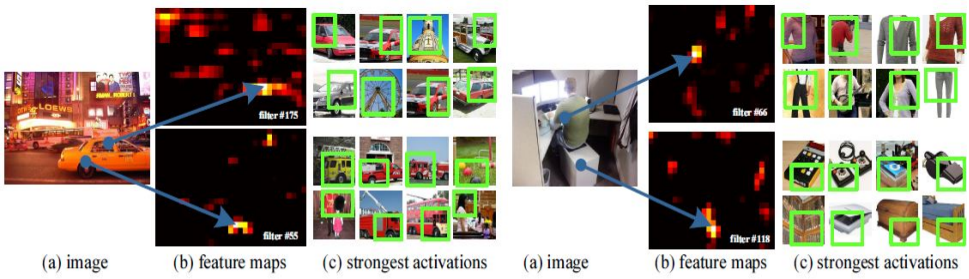
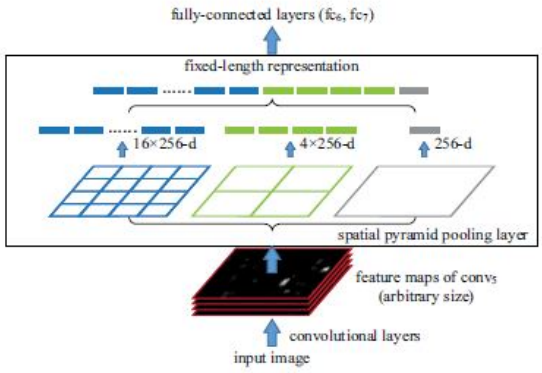


Figure 2: Visualization of the feature maps. (a) Two images in Pascal VOC 2007. (b) The feature maps of some $conv_5$ filters. The arrows indicate the strongest responses and their corresponding positions in the images. (c) The ImageNet images that have the strongest responses of the corresponding filters. The green rectangles mark the receptive fields of the strongest responses.

SPP-Layer原理：

在RNN中，conv5后是pool5;在SPP-net中，用SPP-layer替代原来的pool5，其目标是为了使不同大小输入图像在经过SPP-Layer后得到的特征向量长度相同。其原理如图如下所示



SPP与金字塔pooling类似，即我们先确定最终pooling得到的featuremap大小，例如4*4 bins，3*3 bins，2*2 bins，1*1 bins。那么我们已知conv5输出的featuremap大小（例如，256个13*13的feature map）.那么，对于一个13*13的feature map,我们可以通过spatial pyramid pooling （SPP）的方式得到输出结果：当window=ceil(13/4)=4, stride=floor(13/4)=3,可以得到的4*4 bins；当window=ceil(13/3)=5, stride=floor(13/3)=4,可以得到的3*3 bins；

最新评论

1. Re:深度学习检测方法梳理

@fanninnypeom多多交流...

--venus024

2. Re:深度学习检测方法梳理

太感谢了，将这个和paper对照解除了我很多疑惑

--fanninnypeom

3. Re:CVPR2016目标检测之定位准确性篇：Loc-Net

爱上一个人，就像突然有了盔甲，也突然有了软肋。

--一晌贪欢

4. Re:R-CNN,SPP-NET, Fast-R-CNN,Faster-R-CNN, YOLO, SSD系列深度学习检测方法梳理

看着等带劲的时候发现到SSD后面就木有了期待博主更新，包括rfcn相关的内容

--codename.net

阅读排行榜

1. 深度学习检测方法梳理(6447)
2. CVPR2016目标检测之识别效率篇：YOLO, G-CNN, Loc-Net(5011)
3. R-CNN,SPP-NET, Fast-R-CNN,Faster-R-CNN, YOLO, SSD系列深度学习检测方法梳理(4499)
4. CVPR2016目标检测之识别精度篇：ResNet, ION, HyperNet，R-FCN(1092)
5. CVPR2016目标检测之定位准确性篇： Loc-Net(887)

评论排行榜

1. 深度学习检测方法梳理(2)
2. R-CNN,SPP-NET, Fast-R-CNN,Faster-R-CNN, YOLO, SSD系列深度学习检测方法梳理(1)
3. CVPR2016目标检测之定位准确性篇： Loc-Net(1)

推荐排行榜

1. R-CNN,SPP-NET, Fast-R-CNN,Faster-R-CNN, YOLO, SSD系列深度学习检测方法梳理(4)
2. 深度学习检测方法梳理(3)
3. CVPR2016目标检测之识别效率篇：YOLO, G-CNN, Loc-Net(1)

当window=ceil(13/2)=7, stride=floor(13/2)=6,可以得到的2*2 bins；当window=ceil(13/1)=13, stride=floor(13/1)=13,可以得到的1*1 bins.因此SPP-layer后的输出是256*（4*4+3*3+2*2+1*1）=256*30长度的向量。不难看出，SPP的关键实现在于通过conv5输出的feature map宽高和SPP目标输出bin的宽高计算spatial pyramid pooling中不同分辨率Bins对应的pooling window和pool stride尺寸。

原作者在训练时采用两种不同的方式，即1.采用相同尺寸的图像训练SPP-net 2.采用不同尺寸的图像训练SPP-net。实验结果表明：使用不同尺寸输入图像训练得到的SPP-Net效果更好。

SPP-Net +SVM训练：

采用selective search可以提取到一系列proposals，由于已经训练完成SPP-Net,那么我们先将整图代入到SPP-Net中，得到的conv5的输出。接下来，区别于R-CNN，新方法不需要对不同尺寸的proposals进行Crop或Wrap，直接根据proposal在图中的相对位置关系计算得到proposal在整图conv5输出中的映射输出结果。这样，对于2000个proposal，我们事实上从conv1-->conv5只做了一遍前向，然后进行2000次conv5 featuremap的集合映射，再通过SPP-Layer，就可以得到的2000组长度相同的SPP-Layer输出向量，进而通过全连接层生成最终2000个proposal的卷积神经网络特征。接下来就和R-CNN类似，训练SVMs时对于所有proposal进行严格的标定（可以这样理解，当且仅当一个候选框完全包含ground truth区域且不属于ground truth部分不超过e.g,候选框区域的5%时认为该候选框标定结果为目标，否则位背景），然后将所有proposal经过CNN处理得到的特征和SVM新标定结果输入到SVMs分类器进行训练得到分类器预测模型。

当然，如果觉得SVM训练很麻烦，可以直接在SPP-Net后再加一个softmax层，用好的标定结果去训练最后的softmax层参数。

3. Fast-R-CNN

基于R-CNN和SPP-Net思想，RBG提出了Fast-R-CNN算法。如果选用VGG16网络进行特征提取，在训练阶段，Fast-R-CNN的速度相比RCNN和SPP-Net可以分别提升9倍和3倍；在测试阶段，Fast-R-CNN的速度相比RCNN和SPP-Net可以分别提升213倍和10倍。

R-CNN和***SPP-Net***缺点：

1.R-CNN和SPP-Net的训练过程类似，分多个阶段进行，实现过程较复杂。这两种方法首先选用Selective Search方法提取proposals,然后用CNN实现特征提取，最后基于SVMs算法训练分类器，在此基础上还可以进一步学习检测目标的boulding box。

2.R-CNN和SPP-Net的时间成本和空间代价较高。SPP-Net在特征提取阶段只需要对整图做一遍前向CNN计算，然后通过空间映射方式计算得到每一个proposal相应的CNN特征；区别于前者，RCNN在特征提取阶段对每一个proposal均需要做一遍前向CNN计算，考虑到proposal数量较多（~2000个），因此RCNN特征提取的时间成本很高。R-CNN和SPP-Net用于训练SVMs分类器的特征需要提前保存在磁盘，考虑到2000个proposal的CNN特征总量还是比较大，因此造成空间代价较高。

3.R-CNN检测速度很慢。RCNN在特征提取阶段对每一个proposal均需要做一遍前向CNN计算，如果用VGG进行特征提取，处理一幅图像的所有proposal需要47s。

4.特征提取CNN的训练和SVMs分类器的训练在时间上是先后顺序，两者的训练方式独立，因此SVMs的训练Loss无法更新SPP-Layer之前的卷积层参数，因此即使采用更深的CNN网络进行特征提取，也无法保证SVMs分类器的准确率一定能够提升。

Fast-R-CNN亮点：

1.Fast-R-CNN检测效果优于R-CNN和SPP-Net

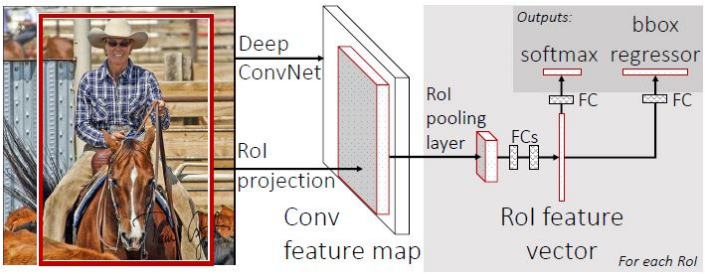
2.训练方式简单，基于多任务Loss,不需要SVM训练分类器。

3.Fast-R-CNN可以更新所有层的网络参数（采用ROI Layer将不再需要使用SVM分类器，从而可以实现整个网络端到端训练）。

4.不需要将特征缓存到磁盘。

Fast-R-CNN架构：

Fast-R-CNN的架构如下图所示（https://github.com/rbgirshick/fast-rcnn/blob/master/models/VGG16/train.prototxt，可以参考此链接理解网络模型）：输入一幅图像和Selective Search方法生成的一系列Proposals，通过一系列卷积层和Pooling层生成feature map,然后用RoI（region of ineterst）层处理最后一个卷积层得到的feature map为每一个proposal生成一个定长的特征向量roi_pool5。RoI层的输出roi_pool5接着输入到全连接层产生最终用于多任务学习的特征并用于计算多任务Loss。全连接输出包括两个分支：1.SoftMax Loss:计算K+1类的分类Loss函数，其中K表示K个目标类别，1表示背景；2.Regression Loss:即K+1的分类结果相应的Proposal的Bounding Box四个角点坐标值。最终将所有结果通过非极大抑制处理产生最终的目标检测和识别结果。



3.1 RoI Pooling Layer

事实上，RoI Pooling Layer是SPP-Layer的简化形式。SPP-Layer是空间金字塔Pooling层，包括不同的尺度；RoI Layer只包含一种尺度，如论文中所述7*7。这样对于RoI Layer的输入（r,c,h,w），RoI Layer首先产生7*7个r*c*(h/7)*(w/7)的Block(块)，然后用Max-Pool方式求出每一个Block的最大值，这样RoI Layer的输出是r*c*7*7。

3.2 预训练网络初始化

RBG采用前辈们训练ImageNet时得到的网络模型（例如VGG16模型）初始化Fast-R-CNN模型中RoI层之前的所有层，我们可以把网络结构总结如下：13个卷积层+4个Pooling层+RoI层+2个FC层+两个平级层（即SoftmaxLoss层和SmoothL1Loss层）。其中，VGG16的第5个Pool层倍RoI层替换掉。

3.3 Finetuning for detection

3.3.1 Fast-R-CNN在网络训练阶段采用了一些trick，每个minibatch由N个图片（N=2）中的R个Proposal（R=128）组成。这种方式比从128张不同图片中提取1个Proposal的方式块64倍。当然，这种方式在一定程度会造成收敛速度变慢。另外，Fast-R-CNN无需SVM分类器，而是通过Softmax Classifer和Bounding-Box Regressors联合训练的方式更新所有参数。注意：从2张图中选取128个proposals时，需要保证至少25%的proposals与groundtruth的IoU超过0.5，剩下的全部作为背景类。不需要其它任何数据扩增操作。

3.3.2 多任务Loss:Fast R-CNN网络有两个同级别子Layer，分别用于分类和回归。分类选用SoftmaxLoss，回归使用SmoothL1Loss.两者的权重比例为1：1

3.3.3 SGD hyer-parameters：用于softmax分类任务和bounding-box回归的fc层参数用标准差介于0.01~0.001之间的高斯分布初始化。

3.4 Truncated SVD快速检测

在检测段，*RBG*使用*truncated SVD*优化较大的*FC*层，这样*RoI*数目较大时检测端速度会得到的加速。

Fast-R-CNN实验结论：

1.多任务Loss学习方式可以提高算法准确率

2.多尺度图像训练Fast-R-CNN与单尺度图像训练相比只能提升微小的mAP,但是时间成本却增加了很多。因此，综合考虑训练时间和mAP，作者建议直接用一种尺度的图像训练Fast-R-CNN.

3.基本上没人不会赞成：训练图像越多，模型准确率也会越高。

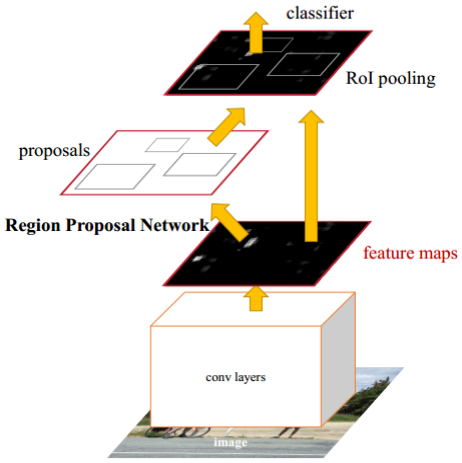
4.RBG的结果表明SoftmaxLoss的方式比SVMs分类器的结果略好一点点，虽然这不能绝对性说明自己的SoftmaxLoss好到哪儿去，但是至少大家不用再那么麻烦的去分步训练一个检测和识别网络了。

5.不是说Proposal提取的越多效果会越好，提的太多反而会导致mAP下降。

4. Faster-R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

在之前介绍的Fast-R-CNN中，第一步需要先使用Selective Search方法提取图像中的proposals。基于CPU实现的Selective Search提取一幅图像的所有Proposals需要约2s的时间。在不计入proposal提取情况下，Fast-R-CNN基本可以实时进行目标检测。但是，如果从端到端的角度考虑，显然proposal提取成为影响端到端算法性能的瓶颈。目前最新的EdgeBoxes算法虽然在一定程度提高了候选框提取的准确率和效率，但是处理一幅图像仍然需要0.2s。因此，Ren Shaoqing提出新的Faster-R-CNN算法，该算法引入了RPN网络（Region Proposal Network）提取proposals。RPN网络是一个全卷积神经网络，通过共享卷积层特征可以实现proposal的提取，RPN提取一幅像的proposal只需要10ms.

Faster-R-CNN算法由两大模块组成：1.PRN候选框提取模块 2.Fast R-CNN检测模块。其中，RPN是全卷积神经网络，用于提取候选框；Fast R-CNN基于RPN提取的proposal检测并识别proposal中的目标。

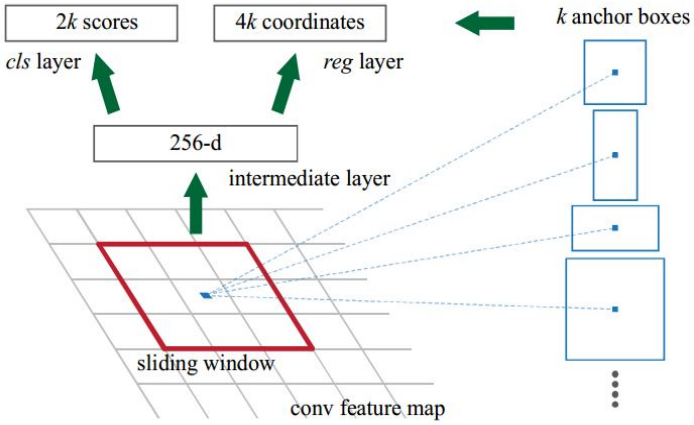


4.1 Region Proposal Network (RPN)

RPN网络的输入可以是任意大小（但还是有最小分辨率要求的，例如VGG是228*228）的图片。如果用VGG16进行特征提取，那么RPN网络的组成形式可以表示为VGG16+RPN。

VGG16：参考https://github.com/rbgirshick/py-faster-rcnn/blob/master/models/pascal_voc/VGG16/faster_rcnn_end2end/train.prototxt，可以看出VGG16中用于特征提取的部分是13个卷积层（conv1_1---->conv5.3），不包括pool5及pool5后的网络层次结构。

RPN：RPN是作者重点介绍的一种网络，如下图所示。RPN的实现方式：在conv5-3的卷积feature map上用一个n*n的滑窗（论文中作者选用了n=3，即3*3的滑窗）生成一个长度为256（对应于ZF网络）或512（对应于VGG网络）维长度的全连接特征。然后在这个256维或512维的特征后产生两个分支的全连接层：1.reg-layer,用于预测proposal的中心锚点对应的proposal的坐标x，y和宽高w，h；2.cls-layer，用于判定该proposal是前景还是背景。sliding window的处理方式保证reg-layer和cls-layer关联了conv5-3的全部特征空间。事实上，作者用全连接层实现方式介绍RPN层实现容易帮助我们理解这一过程，但在实现时作者选用了卷积层实现全连接层的功能。个人理解：全连接层本来就是特殊的卷积层，如果产生256或512维的fc特征，事实上可以用Num_out=256或512, kernel_size=3*3, stride=1的卷积层实现conv5-3到第一个全连接特征的映射。然后再用两个Num_out分别为2*9=18和4*9=36，kernel_size=1*1，stride=1的卷积层实现上一层特征到两个分支cls层和reg层的特征映射。注意：这里2*9中的2指cls层的分类结果包括前后背景两类，4*9的4表示一个Proposal的中心点坐标x，y和宽高w，h四个参数。采用卷积的方式实现全连接处理并不会减少参数的数量，但是使得输入图像的尺寸可以更加灵活。在RPN网络中，我们需要重点理解其中的anchors概念，Loss fucntions计算方式和RPN层训练数据生成的具体细节。



Anchors:字面上可以理解为锚点，位于之前提到的n*n的sliding window的中心处。对于一个sliding window,我们可以同时预测多个proposal，假定有k个。k个proposal即k个reference boxes，每一个reference box又可以用一个scale，一个aspect_ratio和sliding window中的锚点唯一确定。所以，我们在后面说一个anchor,你就理解成一个anchor box 或一个reference box.作者在论文中定义k=9，即3种scales和3种aspect_ratio确定出当前sliding window位置处对应的9个reference boxes， 4*k个reg-layer的输出和2*k个cls-layer的score输出。对于一幅W*H的feature map,对应W*H*k个锚点。所有的锚点都具有尺度不变性。

Loss functions:在计算Loss值之前，作者设置了anchors的标定方法。正样本标定规则：1.如果Anchor对应的reference box与ground truth的IoU值最大，标记为正样本；2.如果Anchor对应的reference box与ground truth的IoU>0.7，标记为正样本。事实上，采用第2个规则基本上可以找到足够的正样本，但是对于一些极端情况，例如所有的Anchor对应的reference box与groud truth的IoU不大于0.7,可以采用第一种规则生成。负样本标定规则：如果Anchor对应的reference box与ground truth的IoU<0.3，标记为负样本。剩下的既不是正样本也不是负样本，不用于最终训练。训练RPN的Loss是有classification loss （即softmax loss）和regression loss （即L1 loss）按一定比重组成的。计算softmax loss需要的是anchors对应的groundtruth标定结果和预测结果，计算regression loss需要三组信息：1.预测框，即RPN网络预测出的proposal的中心位置坐标x,y和宽高w,h；2.锚点reference box:之前的9个锚点对应9个不同scale和aspect_ratio的reference boxes，每一个reference boxes都有一个中心点位置坐标x_a,y_a和宽高w_a,h_a。3.ground truth:标定的框也对应一个中心点位置坐标x*,y*和宽高w*,h*。因此计算regression loss和总Loss方式如下：

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a,$$
$$t_w = \log(w/w_a), \quad t_h = \log(h/h_a),$$
$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a,$$
$$t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a),$$

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

RPN训练设置：在训练RPN时，一个Mini-batch是由一幅图像中任意选取的256个proposal组成的，其中正负样本的比例为1：1。如果正样本不足128，则多用一些负样本以满足有256个Proposal可以用于训练，反之亦然。训练RPN时，与VGG共有的层参数可以直接拷贝经ImageNet训练得到的模型中的参数；剩下没有的层参数用标准差=0.01的高斯分布初始化。

***4.2 RPN*与*Faster-R-CNN*特征共享**

RPN在提取得到proposals后，作者选择使用Fast-R-CNN实现最终目标的检测和识别。RPN和Fast-R-CNN共用了13个VGG的卷积层，显然将这两个网络完全孤立训练不是明智的选择，作者采用交替训练阶段卷积层特征共享：

交替训练（Alternating training）：Step1:训练RPN;Step2:用RPN提取得到的proposal训练Fast R-CNN;Step3:用Faster R-CNN初始化RPN网络中共用的卷积层。迭代执行Step1,2,3，直到训练结束为止。论文中采用的就是这种训练方式，注意：第一次迭代时，用ImageNet得到的模型初始化RPN和Fast-R-CNN中卷积层的参数；从第二次迭代开始，训练RPN时，用Fast-R-CNN的共享卷积层参数初始化RPN中的共享卷积层参数，然后只Fine-tune不共享的卷积层和其他层的相应参数。训练Fast-RCNN时，保持其与RPN共享的卷积层参数不变，只Fine-tune不共享的层对应的参数。这样就可以实现两个网络卷积层特征共享训练。相应的网络模型请参考https://github.com/rbgirshick/py-faster-rcnn/tree/master/models/pascal_voc/VGG16/faster_rcnn_alt_opt

4.3 深度挖掘

1.由于Selective Search提取得到的Proposal尺度不一，因此Fast-RCNN或SPP-Net生成的RoI也是尺度不一，最后分别用RoI Pooling Layer或SPP-Layer处理得到固定尺寸金字塔特征，在这一过程中，回归最终proposal的坐标网络的权重事实上共享了整个FeatureMap，因此其训练的网络精度也会更高。但是，RPN方式提取的ROI由k个锚点生成，具有k种不同分辨率，因此在训练过程中学习到了k种独立的回归方式。这种方式并没有共享整个FeatureMap，但其训练得到的网络精度也很高。这，我竟然无言以对。有什么问题，请找Anchors同学。

2.采用不同分辨率图像在一定程度可以提高准确率，但是也会导致训练速度下降。采用VGG16训练RPN虽然使得第13个卷积层特征尺寸至少缩小到原图尺寸的1/16（事实上，考虑到kernel_size作用，会更小一些），然并卵，最终的检测和识别效果仍然好到令我无言以对。

3.三种scale(128*128，256*256，512*512),三种宽高比（1：2，1：1，2：1）,虽然scale区间很大，总感觉这样会很奇怪，但最终结果依然表现的很出色。

4.训练时（例如600*1000的输入图像），如果reference box （即anchor box）的边界超过了图像边界，这样的anchors对训练Loss不产生影响，即忽略掉这样的Loss.一幅600*1000的图经过VGG16大约为40*60，那么anchors的数量大约为40*60*9，约等于20000个anchor boxes.去除掉与图像边界相交的anchor boxes后，剩下约6000个anchor boxes,这么多数量的anchor boxes之间会有很多重叠区域，因此使用非极值抑制方法将IoU>0.7的区域全部合并，剩下2000个**anchor boxes**（同理，在最终检测端，可以设置规则将概率大于某阈值P且IoU大于某阈值T的预测框（注意，和前面不同，不是**anchor boxes**）采用非极大抑制方法合并）。在每一个epoch训练过程中，随机从一幅图最终剩余的这些anchors采样256个anchor box作为一个Mini-batch训练RPN网络。

4.3 实验

1.PASCAL VOC 2007：使用*ZF-Net*训练*RPN*和*Fast-R-CNN,*那么*SelectiveSearch+Fast-R-CNN, EdgeBox+Fast-R-CNN, RPN+Fast-R-CNN*的准确率分别为：*58.7%，58.6%，59.9%. SeletiveSeach*和*EdgeBox*方法提取*2000*个*proposal*，*RPN*最多提取*300*个*proposal,*因此卷积特征共享方式提取特征的*RPN*显然在效率是更具有优势。

*2.*采用*VGG*以特征不共享方式和特征共享方式训练*RPN+Fast-R-CNN,*可以分别得到*68.5%*和*69.9%*的准确率（*VOC2007*）。此外，采用*VGG*训练*RCNN*时，需要花*320ms*提取*2000*个*proposal*，加入*SVD*优化后需要*223ms*，而*Faster-RCNN*整个前向过程（包括*RPN+Fast-R-CNN*）总共只要*198ms.*

*3.Anchors*的*scales*和*aspect_ratio*的数量虽然不会对结果产生明显影响，但是为了算法稳定性，建议两个参数都设置为合适的数值。

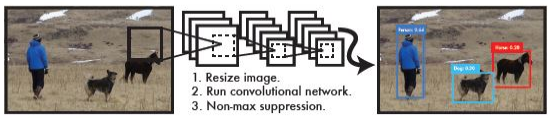
*4.*当*Selective Search*和*EdgeBox*提取的*proposal*数目由*2000*减少到*300*时，*Faste-R-CNN*的*Recall vs. IoU overlap ratio*图中*recall*值会明显下降；但*RPN*提取的*proposal*数目由*2000*减少到*300*时，*Recall vs. IoU overlap ratio*图中*recall*值会比较稳定。

4.4 总结

特征共享方式训练RPN+Fast-R-CNN能够实现极佳的检测效果，特征共享训练实现了买一送一，RPN在提取Proposal时不仅没有时间成本，还提高了proposal质量。因此Faster-R-CNN中交替训练RPN+Fast-R-CNN方式比原来的SlectiveSeach+Fast-R-CNN更上一层楼。

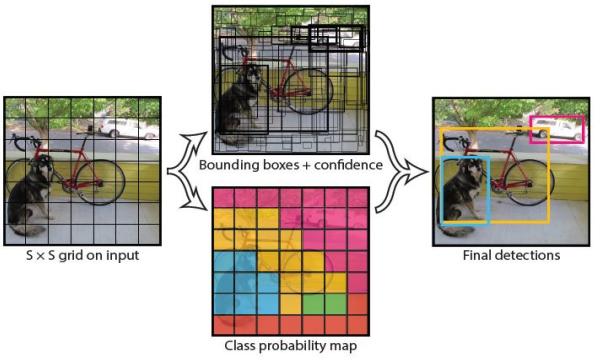
5.YOLO: You Only Look Once：Unified, Real-Time Object Detection

YOLO是一个可以一次性预测多个Box位置和类别的卷积神经网络，能够实现端到端的目标检测和识别，其最大的优势就是速度快。事实上，目标检测的本质就是回归，因此一个实现回归功能的CNN并不需要复杂的设计过程。YOLO没有选择滑窗或提取proposal的方式训练网络，而是直接选用整图训练模型。这样做的好处在于可以更好的区分目标和背景区域，相比之下，采用proposal训练方式的Fast-R-CNN常常把背景区域误检为特定目标。当然,YOLO在提升检测速度的同时牺牲了一些精度。下图所示是YOLO检测系统流程：1.将图像Resize到448*448；2.运行CNN；3.非极大抑制优化检测结果。有兴趣的童鞋可以按照http://pjreddie.com/darknet/install/的说明安装测试一下YOLO的scoring流程，非常容易上手。接下来将重点介绍YOLO的原理。



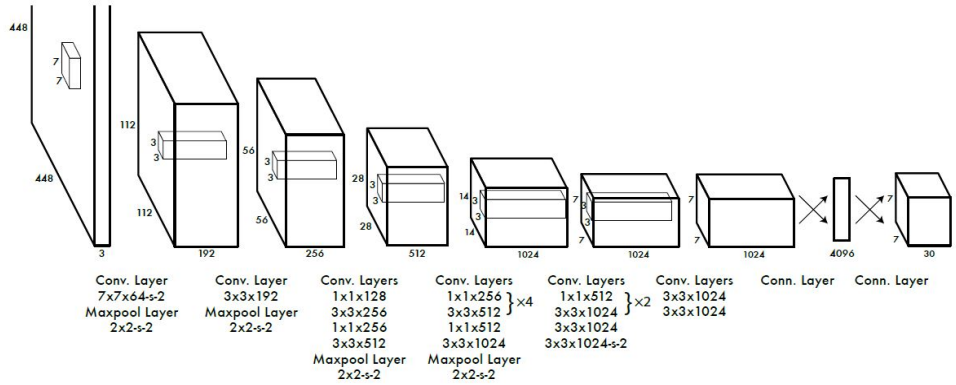
5.1 一体化检测方案

YOLO的设计理念遵循端到端训练和实时检测。YOLO将输入图像划分为S*S个网络，如果一个物体的中心落在某网格(cell)内，则相应网格负责检测该物体。在训练和测试时，每个网络预测B个bounding boxes，每个bounding box对应5个预测参数，即bounding box的中心点坐标(x,y)，宽高（w,h），和置信度评分。这里的置信度评分(Pr(Object)*IOU(pred|truth))综合反映基于当前模型bounding box内存在目标的可能性Pr(Object)和bounding box预测目标位置的准确性IOU(pred|truth)。如果bouding box内不存在物体，则Pr(Object)=0。如果存在物体，则根据预测的bounding box和真实的bounding box计算IOU，同时会预测存在物体的情况下该物体属于某一类的后验概率Pr(Class_i|Object)。假定一共有C类物体，那么每一个网格只预测一次C类物体的条件类概率Pr(Class_i|Object), i=1,2,...,C;每一个网格预测B个bounding box的位置。即这B个bounding box共享一套条件类概率Pr(Class_i|Object), i=1,2,...,C。基于计算得到的Pr(Class_i|Object)，在测试时可以计算某个bounding box类相关置信度：Pr(Class_i|Object)*Pr(Object)*IOU(pred|truth)=Pr(Class_i)*IOU(pred|truth)。如果将输入图像划分为7*7网格（S=7），每个网格预测2个bounding box (B=2)，有20类待检测的目标（C=20），则相当于最终预测一个长度为S*S*(B*5+C)=7*7*30的向量，从而完成检测+识别任务，整个流程可以通过下图理解。



5.1.1 网络设计

YOLO网络设计遵循了GoogleNet的思想，但与之有所区别。YOLO使用了24个级联的卷积（conv）层和2个全连接（fc）层，其中conv层包括3*3和1*1两种Kernel，最后一个fc层即YOLO网络的输出，长度为S*S*(B*5+C)=7*7*30.此外，作者还设计了一个简化版的YOLO-small网络，包括9个级联的conv层和2个fc层，由于conv层的数量少了很多，因此YOLO-small速度比YOLO快很多。如下图所示我们给出了YOLO网络的架构。



5.1.2 训练

作者训练YOLO网络是分步骤进行的：首先，作者从上图网络中取出前20个conv层，然后自己添加了一个average pooling层和一个fc层，用1000类的ImageNet数据与训练。在ImageNet2012上用224*224d的图像训练后得到的top5准确率是88%。然后，作者在20个预训练好的conv层后添加了4个新的conv层和2个fc层，并采用随即参数初始化这些新添加的层，在fine-tune新层时，作者选用448*448图像训练。最后一个fc层可以预测物体属于不同类的概率和bounding box中心点坐标

x,y和宽高w,h。Boundingbox的宽高是相对于图像宽高归一化后得到的，Bounding box的中心位置坐标是相对于某一个网格的位置坐标进行过归一化，因此x,y,w,h均介于0到1之间。

在设计Loss函数时，有两个主要的问题：1.对于最后一层长度为7*7*30长度预测结果，计算预测loss通常会选用平方和误差。然而这种Loss函数的位置误差和分类误差是1：1的关系。2.整个图有7*7个网格，大多数网格实际不包含物体（当物体的中心位于网格内才算包含物体），如果只计算Pr(Class_i),很多网格的分类概率为0，网格loss呈现出稀疏矩阵的特性，使得Loss收敛效果变差，模型不稳定。为了解决上述问题，作者采用了一系列方案：

1.增加bounding box坐标预测的loss权重，降低bounding box分类的loss权重。坐标预测和分类预测的权重分别是$\lambda_{coord}=5,\lambda_{noobj}=0.5$.

2.平方和误差对于大和小的bounding box的权重是相同的，作者为了降低不同大小bounding box宽高预测的方差，采用了平方根形式计算宽高预测loss，即sqrt(w)和sqrt(h)。

训练Loss组成形式较为复杂，这里不作列举，如有兴趣可以参考作者原文慢慢理解体会。

5.1.3 测试

作者选用PASAL VOC图像测试训练得到的YOLO网络，每幅图会预测得到98个（7*7*2）个bouding box及相应的类概率。通常一个cell可以直接预测出一个物体对应的bounding box,但是对于某些尺寸较大或靠近图像边界的物体，需要多个网格预测的结果通过非极大抑制处理生成。虽然YOLO对于非极大抑制的依赖不及R-CNN和DPM，但非极大抑制确实可以将mAP提高2到3个点。

5.2 方法对比

作者将YOLO目标检测与识别方法与其他几种经典方案进行比较可知：

DPM(Deformable parts models): DPM是一种基于滑窗方式的目标检测方法，基本流程包括几个独立的环节：特征提取，区域划分，基于高分值区域预测bounding box。YOLO采用端到端的训练方式，将特征提取、候选框预测，非极大抑制及目标识别连接在一起，实现了更快更准的检测模型。

R-CNN：R-CNN方案分需要先用SeletiveSearch方法提取proposal,然后用CNN进行特征提取，最后用SVM训练分类器。如此方案，诚繁琐也！YOLO精髓思想与其类似，但是通过共享卷积特征的方式提取proposal和目标识别。另外，YOLO用网格对proposal进行空间约束，避免在一些区域重复提取Proposal，相较于SeletiveSearch提取2000个proposal进行R-CNN训练，YOLO只需要提取98个proposal，这样训练和测试速度怎能不快？

Fast-R-CNN、Faster-R-CNN、Fast-DPM: Fast-R-CNN和Faster-R-CNN分别替换了SVMs训练和SelectiveSeach提取proposal的方式，在一定程度上加速了训练和测试速度，但其速度依然无法和YOLO相比。同理，将DPM优化在GPU上实现也无出YOLO之右。

5.3 实验

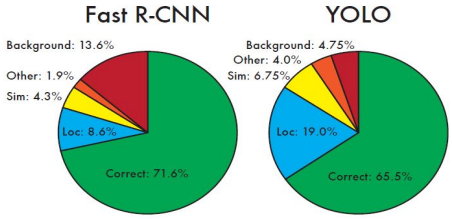
5.3.1 实时检测识别系统对比

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

*5.3.2 VOC2007*准确率比较

	mAP	Combined	Gain
Fast R-CNN	71.8	-	-
Fast R-CNN (2007 data)	**66.9**	72.4	.6
Fast R-CNN (VGG-M)	59.2	72.4	.6
Fast R-CNN (CaffeNet)	57.1	72.1	.3
YOLO	63.4	**75.0**	**3.2**

*5.3.3 Fast-R-CNN*和*YOLO*错误分析



如图所示，不同区域分别表示不同的指标：

Correct：正确检测和识别的比例，即分类正确且IOU>0.5

Localization:分类正确，但0.1<IOU<0.5

Similar:类别相似，IOU>0.1

Other:分类错误，IOU>0.1

Background: 对于任何目标IOU<0.1

可以看出，YOLO在定位目标位置时准确度不及Fast-R-CNN。YOLO的error中，目标定位错误占据的比例最大，比Fast-R-CNN高出了10个点。但是，YOLO在定位识别背景时准确率更高，可以看出Fast-R-CNN假阳性很高（Background=13.6%，即认为某个框是目标，但是实际里面不含任何物体）。

5.3.4 VOC2012准确率比较

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	**79.8**	87.7	49.6	74.9	52.1	86.0	81.7	83.3	**81.8**	**48.6**	**73.5**	59.4	79.9	65.7
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [28]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEP_ENS_COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [29]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [33]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

由于YOLO在目标检测和识别是处理背景部分优势更明显，因此作者设计了Fast-R-CNN+YOLO检测识别模式，即先用R-CNN提取得到一组bounding box，然后用YOLO处理图像也得到一组bounding box。对比这两组bounding box是否基本一致，如果一致就用YOLO计算得到的概率对目标分类，最终的bouding box的区域选取二者的相交区域。Fast-R-CNN的最高准确率可以达到71.8%,采用Fast-R-CNN+YOLO可以将准确率提升至75.0%。这种准确率的提升是基于YOLO在测试端出错的情况不同于Fast-R-CNN。虽然Fast-R-CNN_YOLO提升了准确率，但是相应的检测识别速度大大降低，因此导致其无法实时检测。

使用VOC2012测试不同算法的mean Average Precision，YOLO的mAP=57.9%，该数值与基于VGG16的RCNN检测算法准确率相当。对于不同大小图像的测试效果进行研究，作者发现：YOLO在检测小目标时准确率比R-CNN低大约8~10%，在检测大目标是准确率高于R-CNN。采用Fast-R-CNN+YOLO的方式准确率最高，比Fast-R-CNN的准确率高了2.3%。

5.4 总结

YOLO是一种支持端到端训练和测试的卷积神经网络，在保证一定准确率的前提下能图像中多目标的检测与识别。

6.SSD:Single Shot MultiBox Detector

爱上一个人，就像突然有了盔甲，也突然有了软肋。

分类: DeepLearning

venus024
关注 - 16
粉丝 - 22
+加关注

4 1

« 上一篇：CVPR2016目标检测之定位准确性篇： Loc-Net
» 下一篇：LSTM神经网络资源总结

posted @ 2016-07-29 11:40 venus024 阅读(4499) 评论(1) 编辑 收藏

评论列表

#1楼 2016-07-30 17:12 codename.net 回复 引用

看着等带劲的时候发现到SSD后面就木有了
期待博主更新，包括rfcn相关的内容

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

发表评论

昵称： 学而优，思则通

评论内容：

提交评论 退出登录 订阅评论

[Ctrl+Enter快捷键提交]