简介

RCNN 论文地址为《Rich feature hierarchies for accurate object detection and semantic segmentation》此论文发表于 2013 年。本文提出了一个可伸缩的物体检测算法，将 mAP 值提高了 30% 相对于之前的算法。作者的方法的两个关键点是：

为了定位和物体分割，在从底到上的候选区域中适应高容量的 CNNs 当被训练的数据标签非常少，监督预训练模型为辅助的任务，然后接着进行特定领域的微调 之所以叫 R-CNN 是因为将候选区域（region proposal）与 CNN 进行了结合。文章也与 OverFeat 进行了对比，R-CNN 要远胜于 OverFeat。

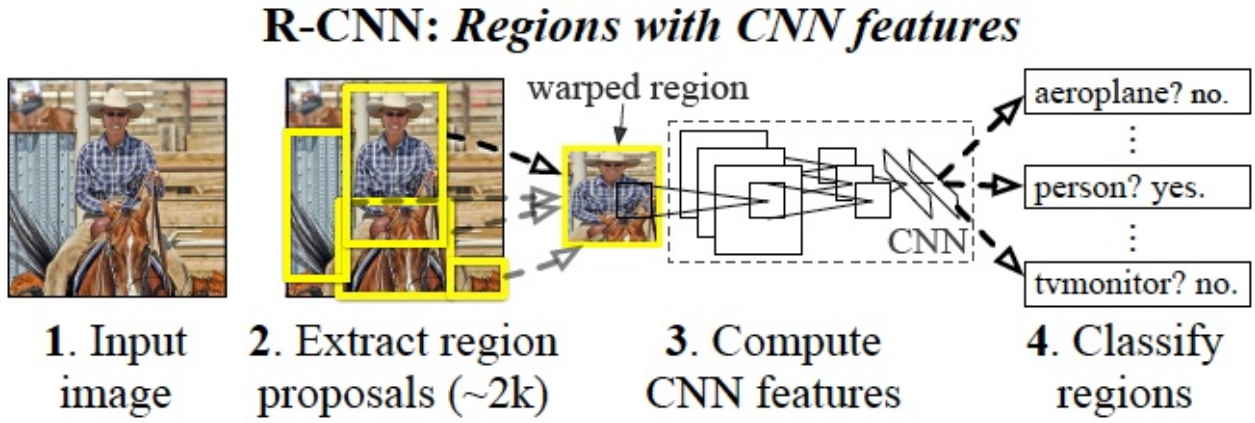
论文中的内容就不过多的说明了，这里主要给出自己看论文时遇到的一些疑惑，结合自己查找的资料记录下自己的理解。对 R-CNN 内容更感兴趣的可以看论文原文，或者看下面参考中给出的其他人精彩的解读，相信看了这些解读文章你一定有收获的。

[参考]

- 个站 - 论文笔记：Rich feature hierarchies for accurate object detection and semantic segmentation
- 个站 - 物体检测论文-RCNN
- CSDN - RCNN学习笔记(2):Rich feature hierarchies for accurate object detection and semantic segmentation
- CSDN - R-CNN论文详解——推荐阅读

R-CNN 介绍

论文中作者使用的结构图



R-CNN流程，主要由三部分组成，

- 独立类别的候选区域（ category-independent region proposals），生成一组对检测器可用的检测坐标。常见的候选区有：
  - objectness、selective search、
  - category-independent object proposal、
  - constrainted parametric min-cuts（CPMC）、
  - multi-scale combinatorial grouping。
- 此处使用选择搜索（selective search ）算法产生 2000 个候选区域（region proposal）

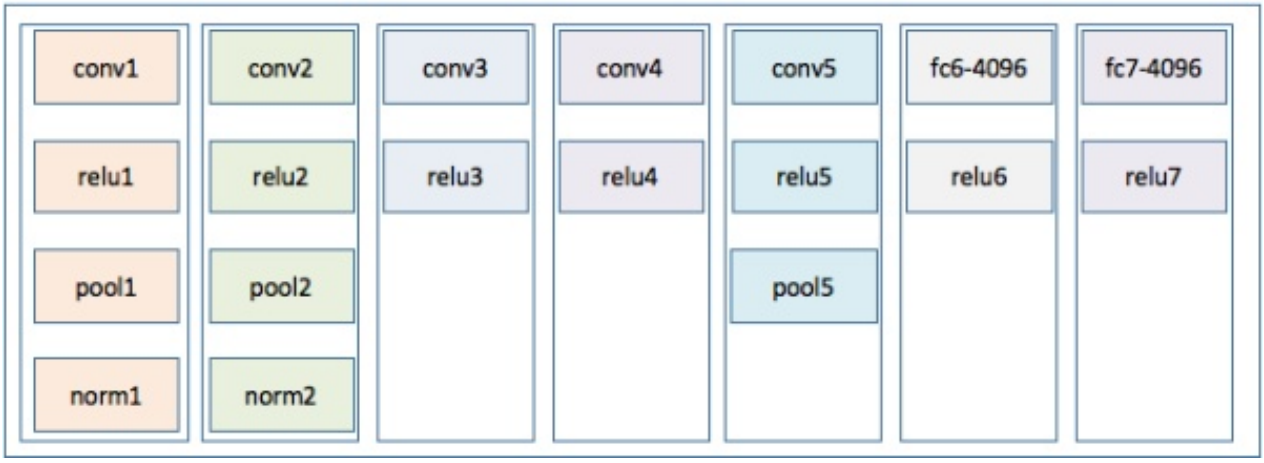
- 使用卷积神经网络从每个区域（即 bounding box）中提取固定尺寸的特征向量
- SVMs 线性分类器，对特征进行分类

因为当前标记的数据比较少，所以论文中使用了非监督预训练（unsupervised pre-training）的神经网络跟在监督的微调（supervised fine-tuning）神经网络后面。

使用边界盒子回归（bounding-box regression） 可以显著的减少定位错误，这是一个显性错误模式（dominant error mode）。

模型设计

论文直接使用了 AlexNet 的模型



因此模型需要的输入图片大小为 227×227 论文中通过验证也证实了，CNN 的结构对 R-CNN 影响很大使用 VGG 要好于 AlexNet。测试结果如下图：

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN T-Net	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
R-CNN T-Net BB	68.1	72.8	56.8	43.0	36.8	66.3	74.2	67.6	34.4	63.5	54.5	61.2	69.1	68.6	58.7	33.4	62.9	51.1	62.5	64.8	58.5
R-CNN O-Net	71.6	73.5	58.1	42.2	39.4	70.7	76.0	74.5	38.7	71.0	56.9	74.5	67.9	69.6	59.3	**35.7**	62.1	64.0	66.5	**71.2**	62.2
R-CNN O-Net BB	**73.4**	**77.0**	**63.4**	**45.4**	**44.6**	**75.1**	**78.1**	**79.8**	**40.5**	**73.7**	**62.2**	**79.4**	**78.1**	**73.1**	**64.2**	35.6	**66.8**	**67.2**	**70.4**	71.1	**66.0**

上图中的 T-Net（TorontoNet） 指的是 AlexNet，而 O-Net（OxfordNet） 指的是 VGG，可以看到使用 VGG 的结果会更好。

相关操作

mAP(Mean Average Precision)

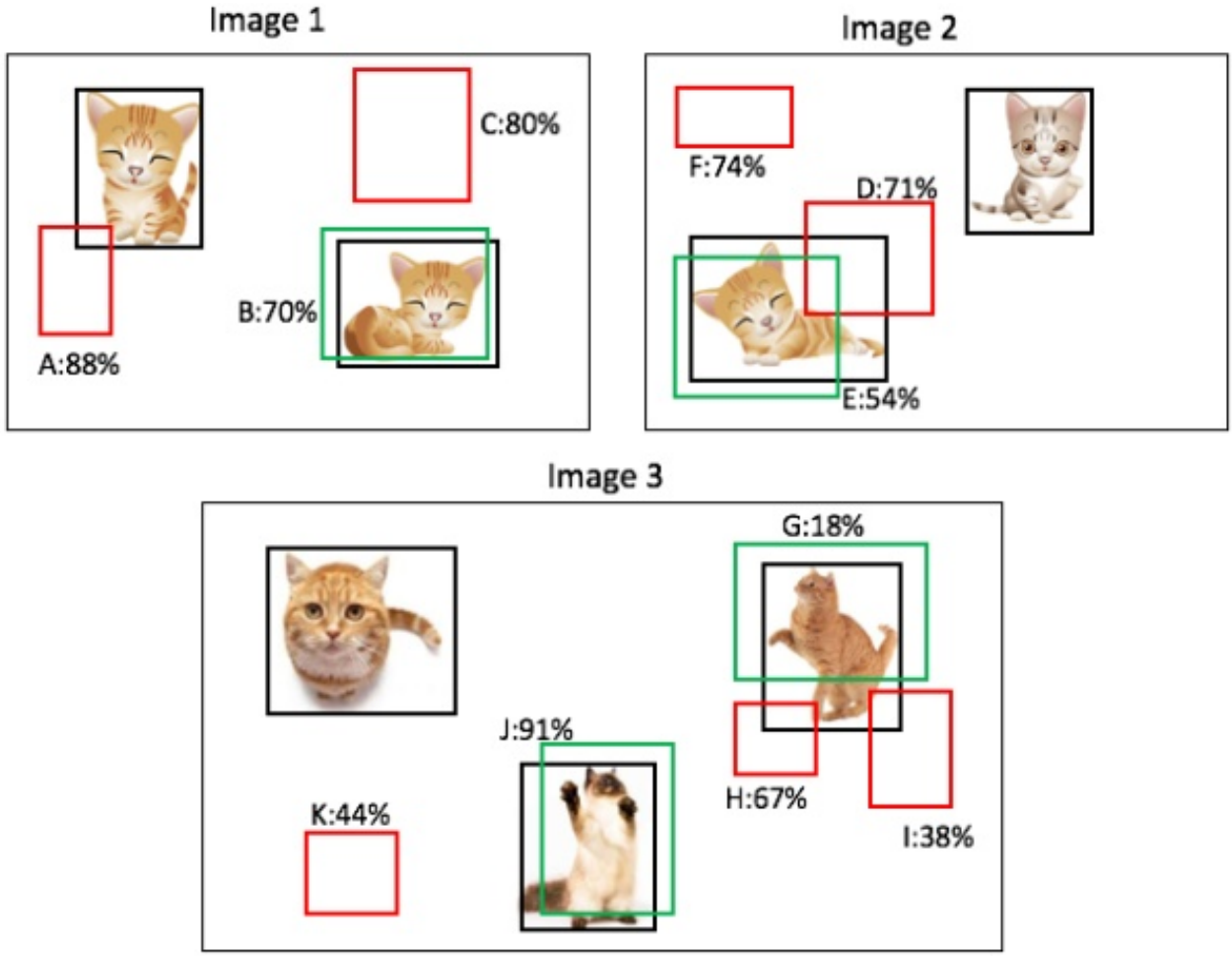
在目标检测的论文中经常出现的一个衡量结果好坏的标准，那么什么是 mAP 呢？通过查阅资料发现，mAP 的计算在不同的时候有不同的方法，这里只给出自己的理解，不一定准确。想要了解这些理解的来源，可以查看从参考部分。

要明白 mAP 的概念，首选需要回顾一下 precision 、recall 以及 IoU的计算方式。

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections}$$
$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths}$$

IoU 的计算可以参考《VGG 论文阅读记录》的 IoU 部分， 在这里我们我们可以设置 IoU 的阈值为 30%。

然而 precision 与 recall 去衡量分类的好坏并不准确，这时候就需要使用 PR（precision-recall） 曲线，其中 Precision 作为 y 轴，Recall 作为纵轴。PR 曲线通常是一个之字形的曲线，在目标检测领域该如何绘制 PR 曲线呢?

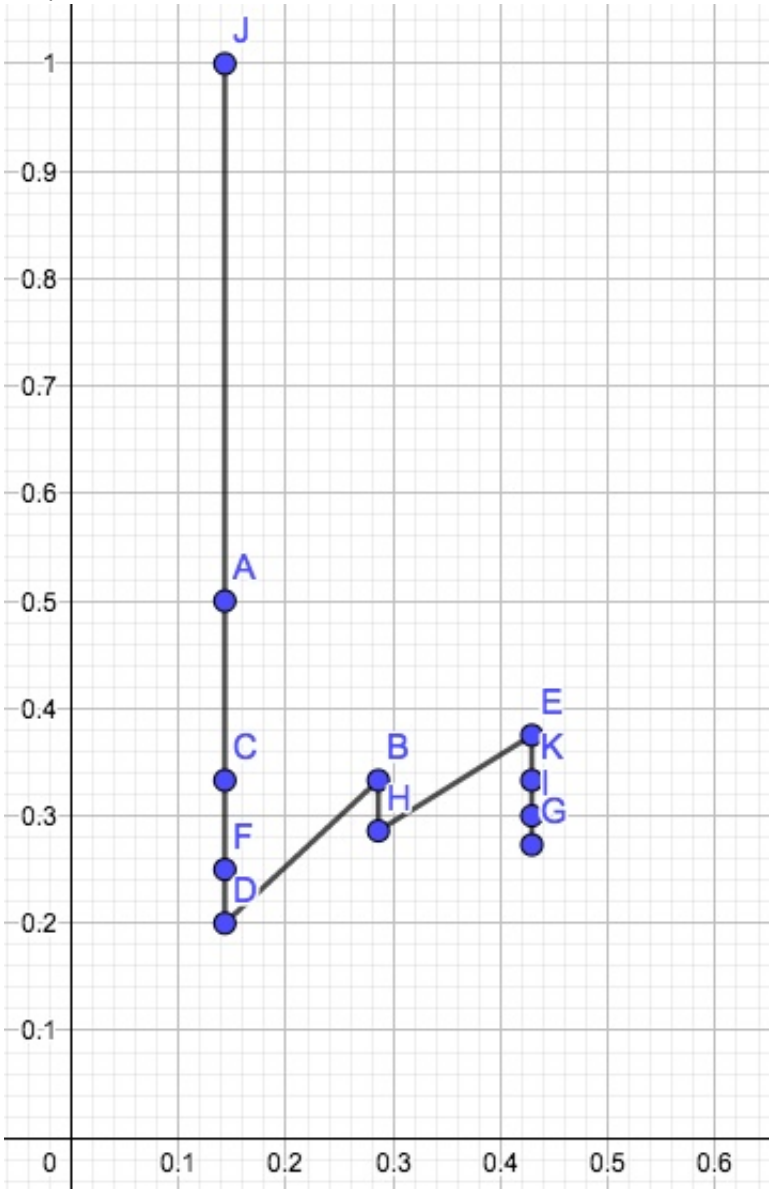


如上图，一共三张图片一共有 7 个 ground-truths ，用黑色矩形表示。一共产生了 11 个检测目标，用绿色与红色表示，编号从 A ~ K旁边的数字表示置信度，因为 IoU 设置的是 0.3 ，因此可以看到标注绿色的为 true positive，红色的则为 false positive。

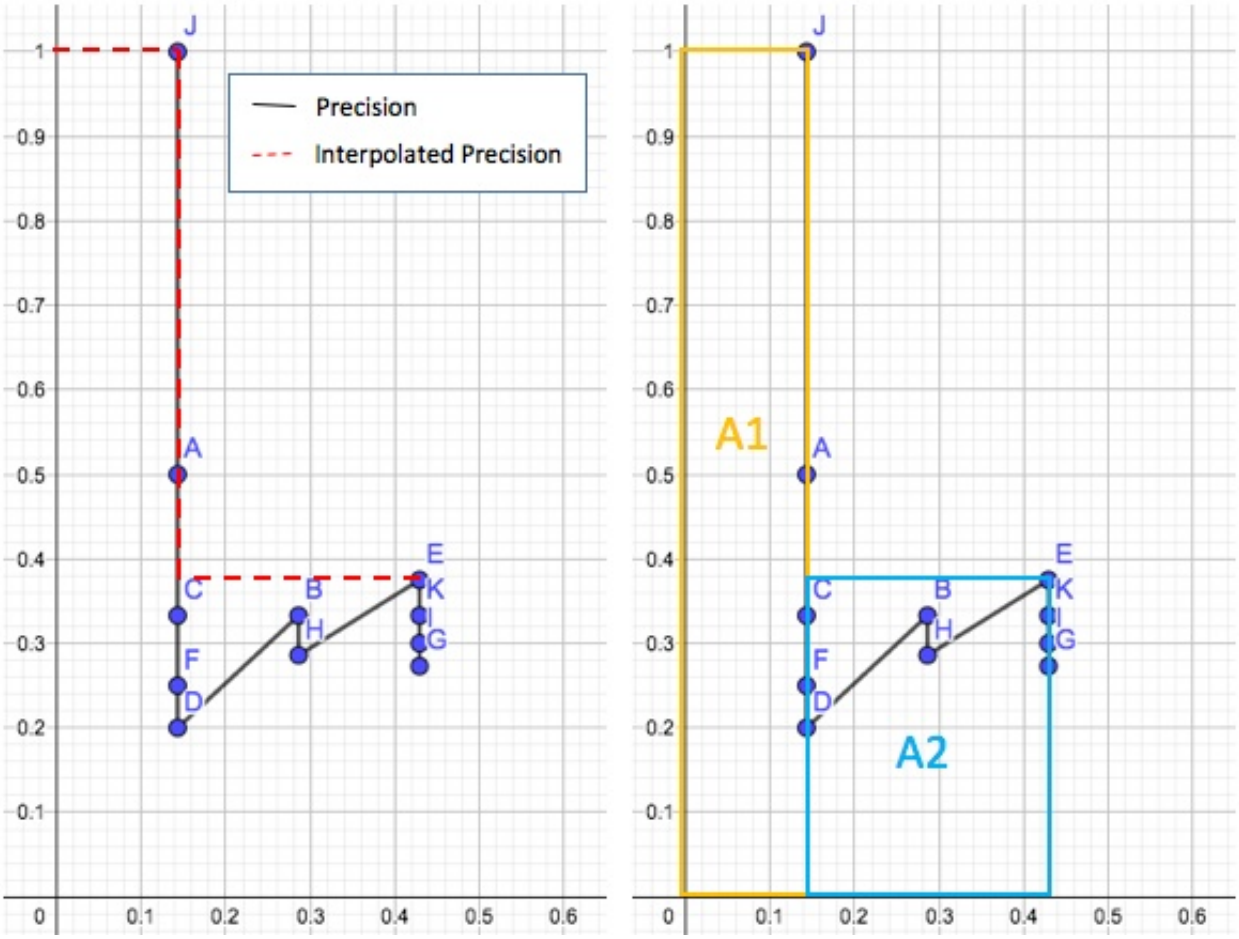
如果想要画出 Precision-Recall 曲线，就需要按照检测出的矩形框的置信度从高到低进行排序，然后计算累积的 TP 和 FP 数量并计算出 Precision 与 Recall（注意他的计算是 TP/all ground-truths）,如下表：（可参考基础概念（2）：各类曲线）

image	Detections	confidences	TP	FP	累积 TP	累积 FP	Precision	Recall
Image3	J	91%	1	0	1	0	1.000	0.143
Image1	A	88%	0	1	1	1	0.500	0.143
Image1	C	80%	0	1	1	2	0.333	0.143
Image2	F	74%	0	1	1	3	0.250	0.143
Image2	D	71%	0	1	1	4	0.200	0.143
Image1	B	70%	1	0	2	4	0.333	0.286
Image3	H	67%	0	1	2	5	0.286	0.286
Image2	E	54%	1	0	3	5	0.375	0.429
Image3	K	44%	0	1	3	6	0.333	0.429
Image3	I	38%	0	1	3	7	0.300	0.429
Image3	G	18%	0	1	3	8	0.273	0.429

然后把点按照（recall，precision）绘制到坐标图中：



绘制完成之后，接着绘制插值Precision 与 AUC（area under curve）：



计算上面右图的面就可以得到 AP：

- AP = A1+A2
- A1 = (0.143 - 0) × 1 = 0.143
- A2 = (0.429 - 0.143) × 0.375 = 0.107
- AP = 0.143 + 0.107 = 0.250 = 25%

上面我们求得的是猫别被的 AP 为0.25，若还还有其他类别，比如狗的为 0.36 、飞机的为 0.54、车子的为0.52，那么mAP 就是这些类别的平均值，即：

$$mAP = \frac{0.25 + 0.36 + 0.54 + 0.52}{4} = 0.4175 = 41.75\%$$

[参考]

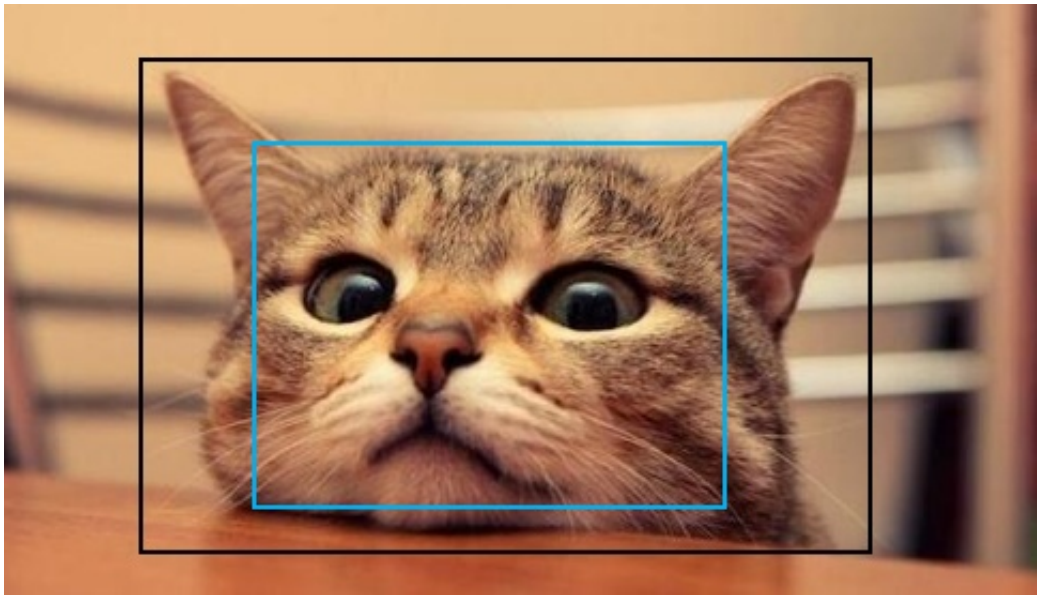
- **GITHUB - Most popular metrics used to evaluate object detection algorithms** mAP 的具体计算，建议阅读此文，此部分主要也是参考此文
- MEDIUM - mAP (mean Average Precision) for Object Detection
- 个站 - Measuring Object Detection models - mAP - What is Mean Average Precision?
- CSDN - 目标检测模型中的性能评估——MAP(Mean Average Precision) 和上文有很多相似的地方
- CSDN - AP: average precision
- stackoverflow - What is a threshold in a Precision-Recall curve?
- 知乎 - 目标检测中的mean average precision是什么含义?
- 个站 - Introduction to the precision-recall plot

Bounding-box Regression

在论文中使用到了 Bounding-box Regression（BBR）方法来提高模型定位的精度，论文附录 C 部分有介绍，但还是不太清楚，于是在网上找了些资料，这里做一下综述。搜索资料的时候看到一段这样的话：

> 最近一直看检测有关的Paper，从rcnn， fast rcnn，faster rcnn，yolo，r-fcn，ssd，到今年cvpr最新的yolo9000。这些paper中损失函数都包含了边框回归，除了rcnn详细介绍了，其他的paper都是一笔带过，或者直接引用rcnn就把损失函数写出来了。
> —— 《CSDN - 边框回归(Bounding Box Regression)详解》

感同深受，很多论文会介绍使用了哪些方法，但是都没有对使用的方法进行详细的介绍。这就怪不得为什么有人提出建议，作者在发表论文之后，再写一些博客来对论文做详细的介绍。很奇怪的是，在网上看资料，大部分资料都来同一个人的回答。。。
BBR 在物体检测的作用主要是用来微调候选区域的边框的位置，使得最终的输出和 ground-truths 更加接近：

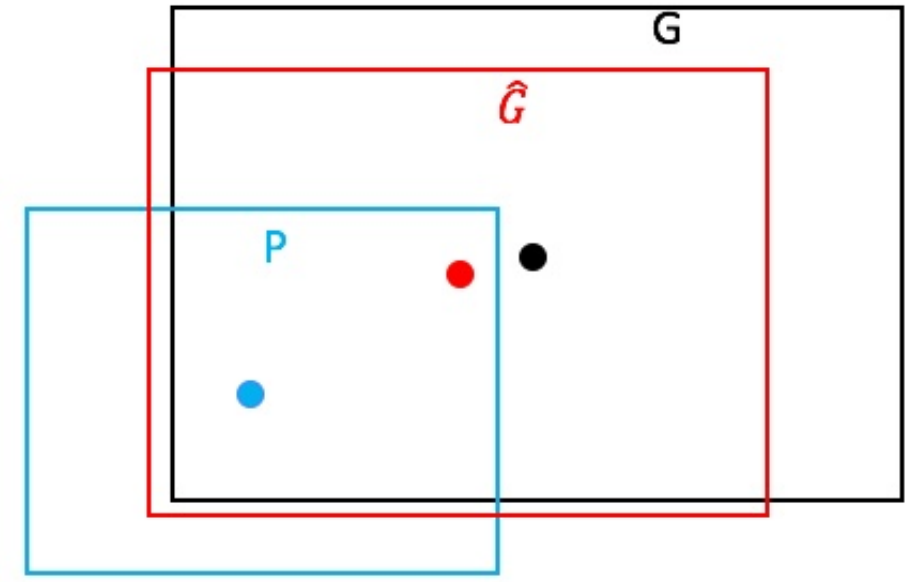


上图黑色框为 ground truth，蓝色框为通过 selective search 选择出的候选区域。可以看到蓝色的框定位并不是特别的准确，这里就会使用 BBR 对蓝色的框进行微调，使得候选区域与 ground-truth 更加的接近。在参考资料都提到了 IoU 小于某个值定位不准确，然后进行微调，那么如果是大于 0.5 的就不进行微调了吗?

我自己的理解是，不管 IoU 的值是多少，都会使用 BBR 对产生的所有候选区（又觉着是对positive 样本进行的）进行微调，如论文中里提到的：

> Inspired by the bounding-box regression employed in DPM , we train a linear regression model to predict a new detection window given the pool5 features for a selective search region proposal.

产生的边框通常使用一个四维向量表示(x, y , w, h)，x、y 的值是边框的中心点的坐标，w 表示边框的框，h 表示高。



如上图蓝色代表原始的候选区域，红色代表微调后的候选区，黑色代表目标 ground truth。如果有 N 个训练数据，我们就会得到 N 个训练对 $\{(P^i, G^i)\}_{i=1,...,N}$ 。这里就是要找到 P 经过映射 f 的处理之后得到更接近 G 的回归边框 $\hat{G}$ 。即：

$$(\widehat{G_x}, \widehat{G_y}, \widehat{G_w}, \widehat{G_h}) = f(P_x, P_y, P_w, P_h)$$

使得

$$(\widehat{G_x}, \widehat{G_y}, \widehat{G_w}, \widehat{G_h}) \approx (G_x, G_y, G_w, G_h)$$

那么接下来要做的就是找出这个 f 。

【以下内容大部分来自此文《 边框回归(Bounding Box Regression)详解》】BBR 是怎么做的呢，其实就替你干通过 平移+缩放。首先定义四个转换函数 $d_x(P), d_y(P), d_w(P), d_h(P)$ ，前两个表示相对于 P 中心尺寸不变的变换(只移动中心点，不改变框的大小)，后两个表示 P 的宽、高在对数空间的转换。这四个函数都是可学习的，通过训练来得到。

首先做平移，平移的大小为$(\Delta x, \Delta y)$，且 $\Delta x = P_w d_x(P), \Delta y = P_h d_y(P)$,因此有：

$$\widehat{G_x} = P_w d_x(P) + P_x \tag{1}$$

$$\widehat{G_y} = P_h d_y(P) + P_y \tag{2}$$

然后做缩放， (S_w, S_h)， $S_w = exp(d_w(P))$， $S_h = exp(d_h(P))$ ，因此有：

$$\widehat{G_w} = P_w exp(d_w(P)) \tag{3}$$

$$\widehat{G_h} = P_h exp(d_h(P)) \tag{4}$$

然后线性回归依据输入特征向量学习一组参数，从而使得回归后的线框更接近 ground truth。

而回归的输入并不是候选区的 (P_x, P_y, P_w, P_h)，而是从模型的第五层提取的特征向量，即 pool5 feature， 在训练的阶段还包括 ground truth 的输入，用来计算候选区与 ground truth 的差值，我们的目标就是为了让此误差变小。此误差定义为： $t_* = (t_x, t_y, t_w, t_h)$，我们需要的就是平移量 (t_x, t_y) 和尺度缩放 (t_w, t_h)：

$$t_x = (G_x - P_x)/P_w \tag{6}$$

$$t_y = (G_y - P_y)/P_h \tag{7}$$

$$t_w = log(G_x/P_w) \tag{8}$$

$$t_h = log(G_h/P_h) \tag{9}$$

目标函数可以表示为 $d_*(P) = w_*^T \Phi_5(P)$，其中 $\Phi_5(P)$ 表示第五层 Pool 提取的特征向量， w_* 是要学习的参数（* 表示 x,y,w,h， 也就是每一个变换对应一个目标函数），$d_*(P)$ 是得到的预测值。我们的目标就是让预测值跟真实值 $t_* = (t_x, t_y, t_w, t_h)$ 差距最小，得到如下的损失函数：

$$Loss = \sum_i^N (t_*^i - \hat{w}_*^T \phi_5(P^i))^2$$

优化的函数目标为：

$$W_* = argmin_{w_*} \sum_i^N (t_*^i - \hat{w}_*^T \phi_5(P^i))^2 + \lambda \|\hat{w}_*\|^2$$

然后使用梯度下降算法或者最小二乘法就可以求得 w_*。

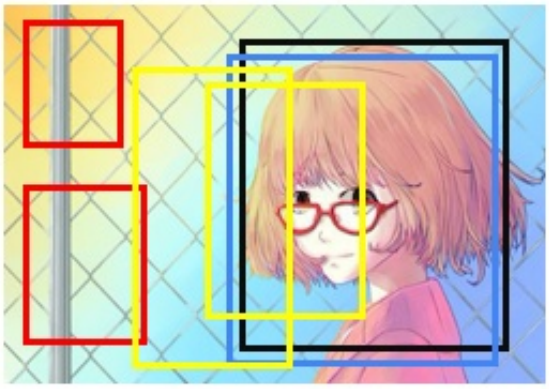
[参考]

- CSDN - 边框回归(Bounding Box Regression)详解
- caffe - bounding box regression

难分样本挖掘(Hard Negative Mining)

在训练过程中，作者使用到了 hard negative mining（难分样本挖掘），然而什么是 难分样本挖掘呢? 在网上搜了下资料，大部分引用的都是 reddit 上同一个人的回答，这里就结合资料整理一下。

在进行目标检测的过程中会产生的候选区域，将与ground-truth box 的 IoU 大于 0.5（论文中设置的即使此值） 的当做正样本（positive sample），小于此值的当做负样本（negative sample）。然后把这些产生的样本送入分类器进行训练，这时就会产生一个问题，负样本的数量远远大于正样本的数量（毕竟图片中的物体数量是有限的），这样训练的过程中就会产生很多假正例（false positive），这样就变成了训练了一个判断假正例的分类器，这显然不是我们想要的，那该怎么办呢? 我们可以把 false positive 中得分较高的样本，重新放入网络中训练，从而加强网络对于 false positive 的判断能力。



如下图：其中黑色的为 ground-truth box，蓝色的为 positive sample，红色和黄色为 false negative sample。可以看到红色是很容易被判断成背景的，即 true negative。而黄色部分就很容易被判断成 false positive，其中的小的黄色矩形被误判的概率更高。

从上面的描述过程，我们就可以知道为什么此方法叫 hard negative mining 了。首先此方法针对的是 negative 的样本，其次这样的样本很难判断 hard。上图中的红色就属于 easy negative，应该他们特别容易判断。在分类器训练之后，得到了得分最高的 false positive，即 mining 出了这些 hard negative，即上图小的黄色矩形，所以叫 hard negative mining。最后在把这些 hard negative 重新送入网络中训练。

引用知乎上 想养一只狗 对此过程的回答《rcnn中的Hard negative mining方法是如何实现的？》

> hard negative mining的实现贯穿于网络的训练过程，简单来说有以下三个步骤
> 1.目标检测中如何根据有标签的数据划分正负训练集?
> 用带标签的图像随机生成图像块，IOU大于某一个阈值的图像块做为正样本，否则为负样本。但一般负样本远远多于正样本，为避免训练出来的模型会偏向预测为负例，需要保持样本均衡，所以初始负样本训练集需要选择负样本集的子集，一般正:负=1:3。
> 2.有了正负训练集集就可以训练神经网络了。经过一轮训练，就可以用这个训练出的模型预测其余的负样本了(就是没有加入训练集的那些负样本)。模型在预测一张图像块后会给出其属于正负的概率，在这里设置一个阈值，预测为正的概率大于这个阈值，就可以把这个图像块加入负样本训练集了。
> 3.正样本训练集不变，负样本训练集除了初始的那些，还有新加入的。拿着这个新的训练集，就可以开始新的一轮训练了。
> 跳到第二步（这个过程是重复的）

[参考]

- REDDIT - What is hard negative mining? And how is it helpful in doing that while training classifiers?
- 知乎 - rcnn中的Hard negative mining方法是如何实现的?
- Refining Bounding-Box Regression for Object Localization

切除研究法(Ablation study)

在论文 3.2节中作者使用了 ablation study 方法研究哪一层和使用什么方法对最后的结果起的作用最大。研究的结果如下图：

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN $pool_5$	51.8	60.2	36.4	27.8	23.2	52.8	60.6	49.2	18.3	47.8	44.3	40.8	56.6	58.7	42.4	23.4	46.1	36.7	51.3	55.7	44.2
R-CNN fc_6	59.3	61.8	43.1	34.0	25.1	53.1	60.6	52.8	21.7	47.8	42.7	47.8	52.5	58.5	44.6	25.6	48.3	34.0	53.1	58.0	46.2
R-CNN fc_7	57.6	57.9	38.5	31.8	23.7	51.2	58.9	51.4	20.0	50.5	40.9	46.0	51.6	55.9	43.3	23.3	48.1	35.3	51.0	57.4	44.7
R-CNN FT $pool_5$	58.2	63.3	37.9	27.6	26.1	54.1	66.9	51.4	26.7	55.5	43.4	43.1	57.7	59.0	45.8	28.1	50.8	40.6	53.1	56.4	47.3
R-CNN FT fc_6	63.5	66.0	47.9	37.7	29.9	62.5	70.2	60.2	32.0	57.9	47.0	53.5	60.1	64.2	52.2	31.3	55.0	50.0	57.7	63.0	53.1
R-CNN FT fc_7	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
R-CNN FT fc_7 BB	**68.1**	**72.8**	**56.8**	**43.0**	**36.8**	**66.3**	**74.2**	**67.6**	**34.4**	**63.5**	**54.5**	**61.2**	**69.1**	**68.6**	**58.7**	**33.4**	**62.9**	**51.1**	**62.5**	**64.8**	**58.5**
DPM v5 [20]	33.2	60.3	10.2	16.1	27.3	54.3	58.2	23.0	20.0	24.1	26.7	12.7	58.1	48.2	43.2	12.0	21.1	36.1	46.0	43.5	33.7
DPM ST [28]	23.8	58.2	10.5	8.5	27.1	50.4	52.0	7.3	19.2	22.8	18.1	8.0	55.9	44.8	32.4	13.3	15.9	22.8	46.2	44.9	29.1
DPM HSC [31]	32.2	58.3	11.5	16.3	30.6	49.9	54.8	23.5	21.5	27.7	34.0	13.7	58.1	51.6	39.9	12.4	23.5	34.4	47.4	45.2	34.3

那么就是 ablation study 究竟是什么？查了些资料，大部分都是互相引用，这里结合资料综述一下。在 CNN 中有不同的模块组成和不同的训练方法，我们想知道每一个模块和训练方法有没有起作用、起了多大的作用。这时候就可以用到 ablation study，他的过程就是先用一个最简单的模型，训练然后记录结果，之后在这个模型基础上在添加新的组件，训练、记录结果；不断地重复上面的过程。

论文中作者的 ablation study 过程是，划分出两种训练方法：含有 fine-tuning（FT）和不含有 FT，在此基础上再划分出使用 pool5、fc6、f7 几种情况。通过此研究，就可以看到 FT 到底有没有效果，以及 pool5、fc6、fc7 到底哪一层起的作用最大。测试的结果如上图。过程很像奥卡姆剃刀，在结果一样的情况下，选择最简单的模型。

更好的描述，可以查看参考中的 QUORA上 Jonathan Uesato的回答。

[参考]

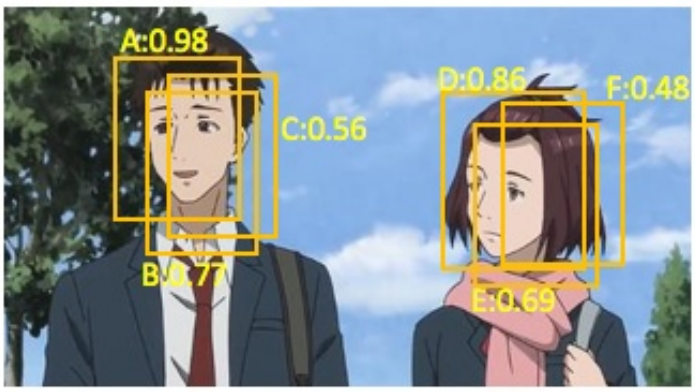
- 知乎 - 什么是 ablation study?
- QUORA - In the context of deep learning, what is an ablation study?
- 个站 - What is ablation study in machine learning

非极大值抑制(non-maximum suppression NMS)

在预测的阶段，论文中提到通过使用 NMS 对候选区域进行筛选，最后得到的矩形基本上就是比较靠近物体的位置了。从名字就可以看出来，该方法就是抑制不是极大值的值，该方法是一个区域搜索算法。该方法的流程如下：

- 首先对产生的矩形区域按得分进行降序排序，筛选出得分最好的矩形
- 然后将与得分最高的区域 IoU 高于某一阈值的矩形删除
- 重复上面步骤，知道没有矩形可选为止，最终得到想要的矩形区域

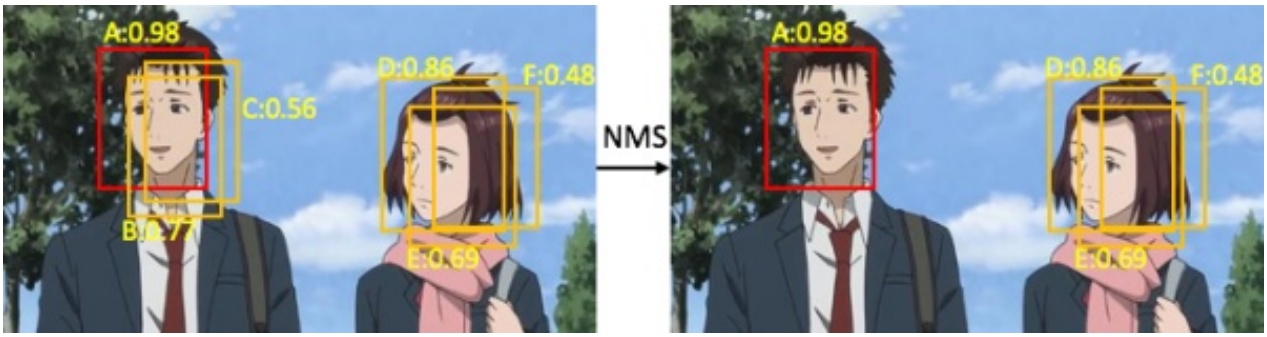
以下以人脸检测为例，如下图：



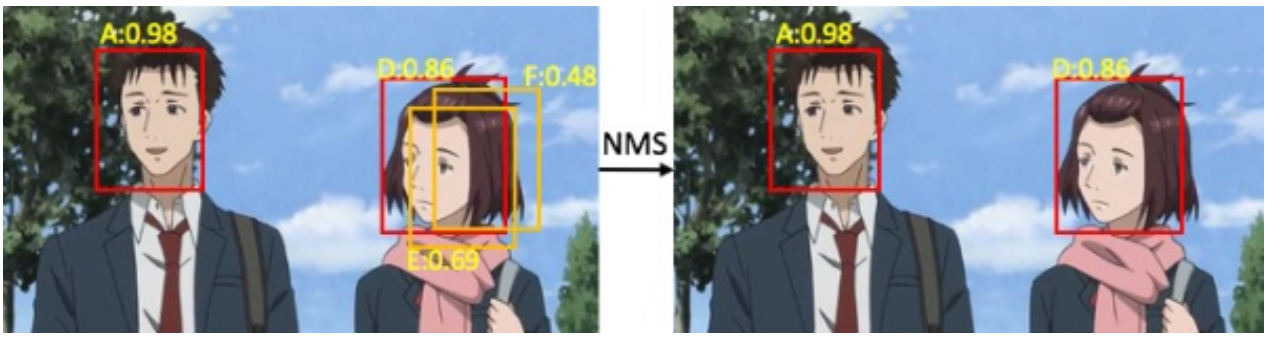
在测试阶段SVM 产生的结果，图中有两个人，产生了 A~E 六个矩形区域和分值。那么按照 NMS 的算法首先对矩形区域按照分值进行排序，排序结果如下：

```
A[0.98]  >  D[0.86]  >  B[0.77]  >  E[0.69]  >  C[0.56]  >  F[0.48]
```

从中出分值最大的，那么 A 就被选中，接下来就删除与 A 矩形的 IoU 大于某个阈值的矩形。从图中可以观察到 B、C 将被删除，A 被保留，如下图：



接下来继续 NMS 操作，从剩下来的 D、E、F 中选择分值最高的，那么 D 将被选中，与 D 的 IoU 大于阈值的矩形将被删除，E、F 被删除留下 D，如下图：



从上图中可以看出已经得到了我们想要的结果，如果还有矩形框，那么就按照上面的步骤不断重复，直到没有可供选择的矩形框位置。 从上述的过程中也可以看到，每一次删除的时候，都是将不是最大值的矩形删除，因此此方法才叫非极大值抑制。

上面的情况是较为理想的情况，对多类别检测任务，如果对每类分别进行NMS，那么当检测结果中包含两个被分到不同类别的目标且其IoU较大时，会得到不可接受的结果。这种情况下需要考虑 NMS 损失到总体的损失当中，其他问题可参考《CNBLOGS - 非极大值抑制（Non-Maximum Suppression，NMS）》。【其实这里不是特别理解说的这种情况，等看了《Rotated Region Based CNN for Ship Detection》论文之后再补充】

[参考]

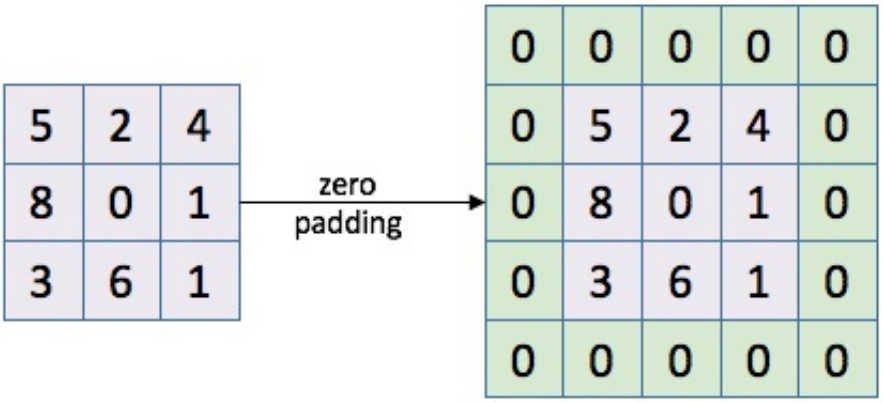
- CNBLOGS - 非极大值抑制（Non-Maximum Suppression，NMS）
- CSDN - NMS——非极大值抑制
- 知乎 - 在Rcnn中为什么使用IoU非极大值抑制?
- 国外 - Non-Maximum Suppression for Object Detection in Python

输入区域处理

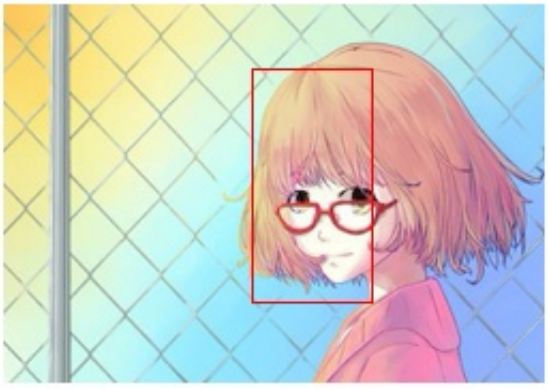
论文中使用 selective search 进行区域搜索，产生出 2000 个尺寸各异的候选区域，然后使用的模型 AlexNet 需要的是 227×227 大小的输入，因此需要对原始的候选区域进行处理，已符合模型的要求。论文中提到了几种不同的处理方法，涉及到了`各项同性（isotopically）`和`各向异性（anisotropically）缩放`，那什么是各项同性与异性缩放呢?

其实就是指的在缩放的时候是否保存长宽比，如果保持原有的宽高比例，就是各向同性，这样图片保持了原有的宽高比不会出现扭曲；如何不保持宽高比就是各向异性，图片在缩放时会出现扭曲。 比如，如果一个原始的图像尺寸为 1024×512，现在我想把 512 的边缩放为 256，如果是各项同性缩放，因为原始图像的比例是 2:1 ，那么缩放之后也应该保持 2:1 ，即长边应该变成 512。各向异性就是不保持原有的 2:1 ，缩放之后可能是 4:3 或者是 16:9 ，这样图片就会出现扭曲。

从上图可以看到使用各项异性处理，如果最初的候选区域不是正方形，那么使用各向异性直接缩放到模型要求的 227×227 ，因为物体的扭曲，会影响到最后的精度。 在进行变形前需要做一次像素填充处理，即 padding，即扩展原有的候选区域，在这里需要注意一下和 CNN 卷积中的 padding 的区别。CNN 中的卷积 padding 是为了输入在卷积之后，输出大小与输入大小一样，且常用的 padding 方式是补零（zero-padding）：



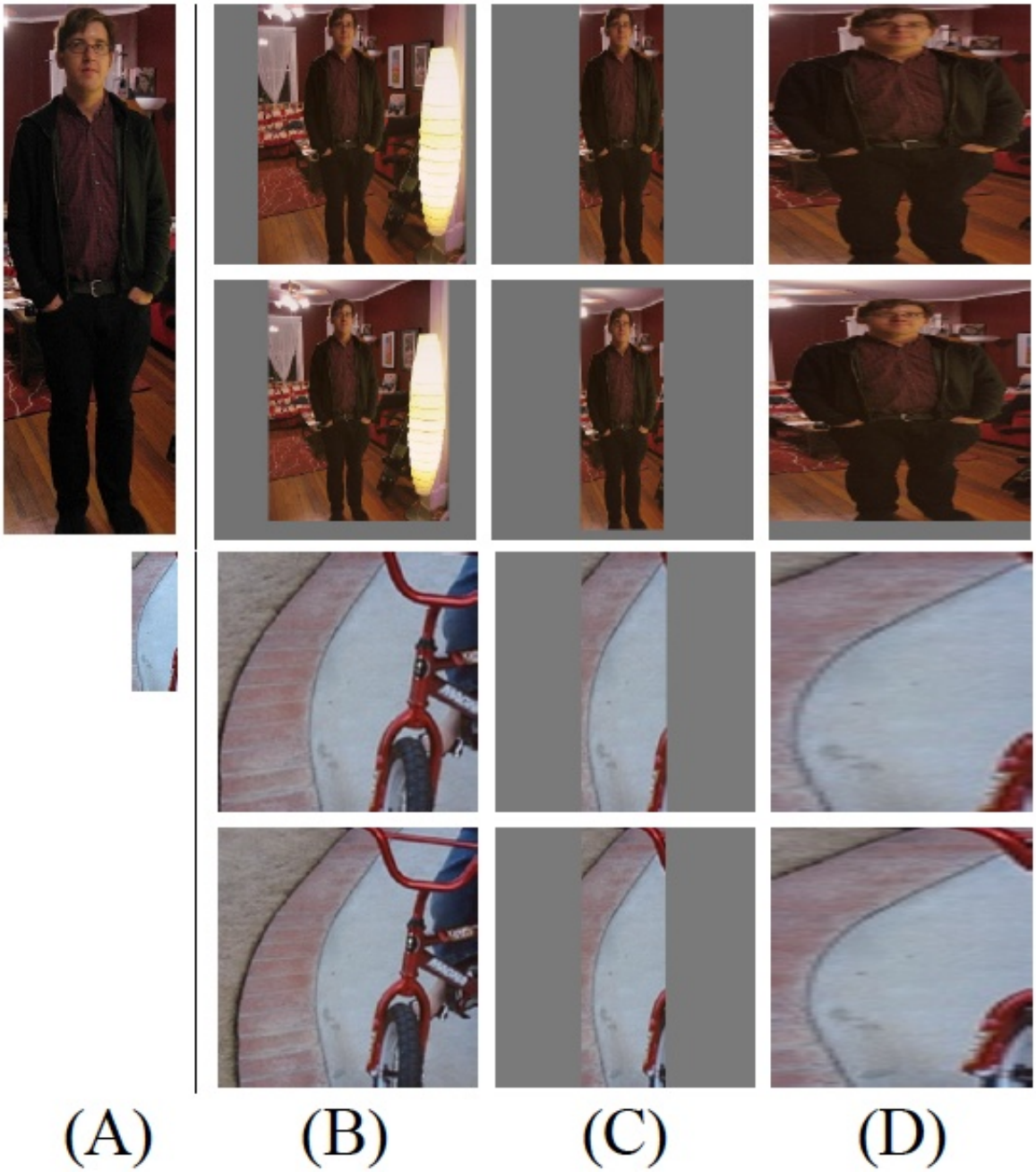
而论文中提到的 padding 是在原有候选区域之上，在把每个边界扩展16像素，如下图：



左图的红色框是原始候选区域的大小，右图的红色实线是在原有的虚线大小之上扩展16像素之后的结果。将候选区域填补处理好之之后就可以进行之后的处理。

如果在 padding 的过程中越过了原图的边界，那么就用候选区域中像素的均值进行补充，如下图：

之后论文中，作者给出的处理方法，处理结果如下图：



A 列是原始的候选区域，相对于CNN模型需要的输入，A 的第一行比 CNN 需要的输入高很多，第二行则比 CNN 需要的输入尺寸小很多。D 列就是各向异性直接缩放（warp transformation）到模型需要的输入尺寸，造成了图片扭曲。B 、C 采用的都是各项同性缩放方法，但具体处理方法不同：

- 对于 B 列，首先把候选区域扩展成正方形（已经 padding 过之后），然后在进行裁剪，如果已经扩展成正方形的过程中，遇到了图片的边界那么就用候选区域中的像素均值进行补充。对于上图中自行车图片，因为原始的候选区域可以扩充到正方形而不会超出边界，而人物因为原始的候选区下边已经达到原图的最底部，所以没有办法在扩充，需要使用均值进行补充。左右也是，扩充之后缩放之后需要在左右补充均值之后尺寸才能达到 227×227
- 对于 C 列，则是padding 之后，使用各向同性缩放到指定尺寸，没有达到输入要求的使用均值将其填充成 227×227。上图中，长度缩放到了227，但宽度达打不到227，因此空出来的部分使用均值填充。

论文中最后验证使用 padding=16 的各项异性是最好的。

[参考]

- QUORA - What is the meaning of isotropically-rescale in paper VGG?
- stackoverflow - what is anisotropic scaling in computer vision?
- CSDN - 各向同性，各向异性缩放
- GITHUB - cqchu/Paper-about-Shanggang

注：大部分图使用 PPT 绘制，点图使用 GeoGeBra 绘制。

【参考汇总】

- CSDN - 边框回归(Bounding Box Regression)详解
- caffe - bounding box regression
- Refining Bounding-Box Regression for Object Localization
- **GITHUB - Most popular metrics used to evaluate object detection algorithms** mAP 的具体计算，建议阅读此文，此部分主要也是参考此文
- MEDIUM - mAP (mean Average Precision) for Object Detection
- 个站 - Measuring Object Detection models - mAP - What is Mean Average Precision?
- CSDN - 目标检测模型中的性能评估——MAP(Mean Average Precision) 和上文有很多相似的地方
- CSDN - AP: average precision
- stackoverflow - What is a threshold in a Precision-Recall curve?
- 知乎 - 目标检测中的mean average precision是什么含义？
- 个站 - Introduction to the precision-recall plot
- 个站 - 论文笔记：Rich feature hierarchies for accurate object detection and semantic segmentation
- 个站 - 物体检测论文-RCNN
- CSDN - RCNN学习笔记(2):Rich feature hierarchies for accurate object detection and semantic segmentation
- CSDN - R-CNN论文详解
- CNBLOGS - 非极大值抑制（Non-Maximum Suppression，NMS）
- CSDN - NMS——非极大值抑制
- 知乎 - 在Rcnn中为什么使用IoU非极大值抑制?
- 国外 - Non-Maximum Suppression for Object Detection in Python
- QUORA - What is the meaning of isotropically-rescale in paper VGG?
- stackoverflow - what is anisotropic scaling in computer vision?
- CSDN - 各向同性，各向异性缩放
- GITHUB - cqchu/Paper-about-Shanggang
- REDDIT - What is hard negative mining? And how is it helpful in doing that while training classifiers?
- 知乎 - rcnn中的Hard negative mining方法是如何实现的？
- 知乎 - 什么是 ablation study？
- QUORA - In the context of deep learning, what is an ablation study?
- 个站 - What is ablation study in machine learning