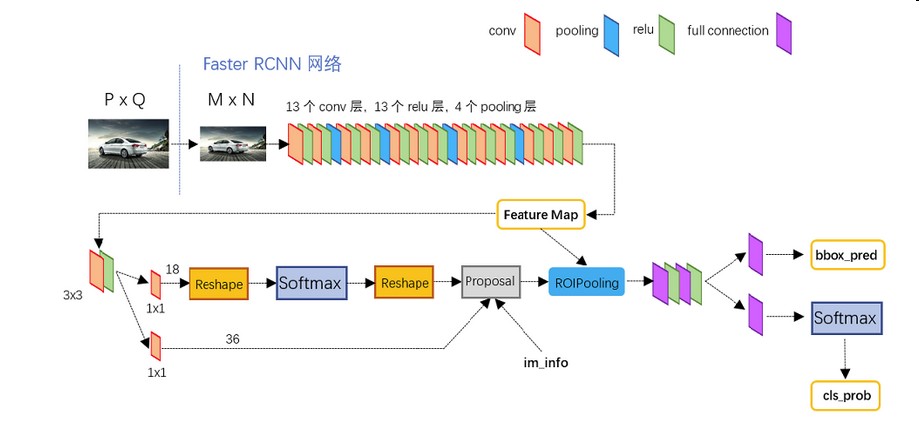
**（1）输入图像预处理；**

1、图像（三通道），每个像素点减去一个均值像素，默认为（102.9801, 115.9465, 122.7717）；

2、缩放；根据插值法进行图像缩放，规则为：首先将图像最短边固定为600个像素点，最长边根据这个缩放比例进行相应调整，如果此时最长边大于1000个像素点，则将最长边固定为1000个像素，最短边根据长边的缩放比例进行相应的调整；（保证都不大于1000，也即长宽比<5:3）

**（2）网络前向计算；**

Faster RCNN的检测过程，主要分为三部分（如下图）：第一部分利用VGG网络结构进行基础的特征提取；第二部分是RPN网络，负责计算可能存在目标的区域（proposals）的坐标以及判断是前景/背景以及利用RPN网络得到的目标区域再经过ROIPooling层得到相同长度的特征向量；第三部分，最后经过两个全连接层接入softmax实现具体分类和更精确的回归坐标。整体结构如下图所示：



1、基础的特征提取网络结构（vgg16）：

(我们需要处理的图像长宽比基本符合小于5:3 ，因此缩放后输入网络的大小通常为：（800,600,3）)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Layer** | **kernel** | **Stride,pad** | **output shape** | **name** |
| Input |  |  | (1,3,600,800) | data |
| Convolution | 3×3 | 1,1 | (1,64,600,800) | conv1\_1 |
| Relu |  |  | (1,64,600,800) | relu1\_1 |
| Convolution | 3×3 | 1,1 | (1,64,600,800) | conv1\_2 |
| Relu |  |  | (1,64,600,800) | relu1\_2 |
| MaxPooling | 2×2 | 2,0 | (1,64,300,400) | pool1 |
| Convolution | 3×3 | 1,1 | (1,128,300,400) | conv2\_1 |
| Relu |  |  | (1,128,300,400) | relu2\_1 |
| Convolution | 3×3 | 1,1 | (1,128,300,400) | conv2\_2 |
| Relu |  |  | (1,128,300,400) | relu2\_2 |
| MaxPooling | 2×2 | 2,0 | (1,128,150,200) | pool2 |
| Convolution | 3×3 | 1,1 | (1,256,150,200) | conv3\_1 |
| Relu |  |  | (1,256,150,200) | relu3\_1 |
| Convolution | 3×3 | 1,1 | (1,256,150,200) | conv3\_2 |
| Relu |  |  | (1,256,150,200) | relu3\_2 |
| Convolution | 3×3 | 1,1 | (1,256,150,200) | conv3\_3 |
| Relu |  |  | (1,256,150,200) | relu3\_3 |
| MaxPooling | 2×2 | 2,0 | (1,256,75,100) | pool3 |
| Convolution | 3×3 | 1,1 | (1,512,75,100) | conv4\_1 |
| Relu |  |  | (1,512,75,100) | relu4\_1 |
| Convolution | 3×3 | 1,1 | (1,512,75,100) | conv4\_2 |
| Relu |  |  | (1,512,75,100) | relu4\_2 |
| Convolution | 3×3 | 1,1 | (1,512,75,100) | conv4\_3 |
| Relu |  |  | (1,512,75,100) | relu4\_3 |
| MaxPooling | 2×2 | 2,0 | (1,512,38,50) | pool4 |
| Convolution | 3×3 | 1,1 | (1,512,38,50) | conv5\_1 |
| Relu |  |  | (1,512,38,50) | relu5\_1 |
| Convolution | 3×3 | 1,1 | (1,512,38,50) | conv5\_2 |
| Relu |  |  | (1,512,38,50) | relu5\_2 |
| Convolution | 3×3 | 1,1 | (1,512,38,50) | conv5\_3 |
| Relu |  |  | (1,512,38,50) | relu5\_3 |

2、RPN网络：

* 1. 产生anchor box

conv5\_3对应的特征图大小为50×38，每个位置将产生9个不同尺度和长宽比的anchor box，用于目标区域位置的计算。9个boxes的坐标为：array([ [ -83., -39., 100., 56.],

[-175., -87., 192., 104.],

[-359., -183., 376., 200.],

[ -55., -55., 72., 72.],

[-119., -119., 136., 136.],

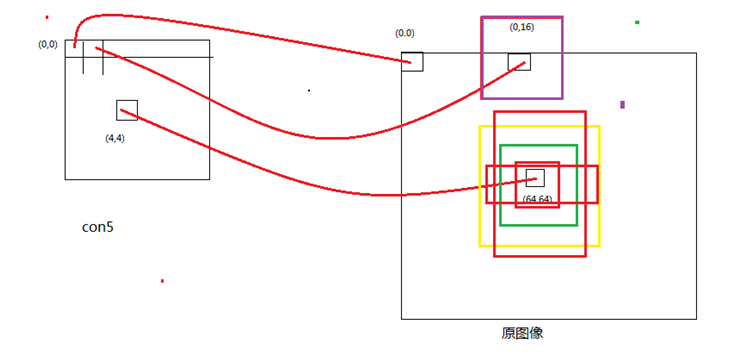
[-247., -247., 264., 264.],

[ -35., -79., 52., 96.],

[ -79., -167., 96., 184.],

[-167., -343., 184., 360.]])

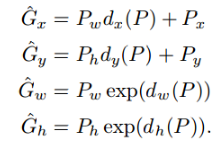
计算这些boxes在原图像上的实际坐标（从原图像到conv5\_3的映射，其实就是缩小了16倍），保证conv5\_3每个特征点位置对应原图像上9个boxes，如下图所示，在实际图像中包含的boxes个数为50×38×9。



* 1. rpn网络结构：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Layer** | **input** | **kernel** | **Stride,pad** | **output shape** | **name** |
| Convolution | relu5\_3(conv5\_3) | 3×3 | 1,1 | (1,512,38,50) | rpn\_conv/3x3 |
| Relu | rpn\_conv/3x3 |  |  | (1,512,38,50) | rpn\_relu/3x3 |
| Convolution | rpn\_relu/3x3 | 1×1 | 1,0 | (1,18,38,50) | rpn\_cls\_score |
| Reshape | rpn\_cls\_score |  |  | (1,2,9\*38,50) | rpn\_cls\_score\_reshape |
| Softmax | rpn\_cls\_score\_reshape |  |  | (1,2,9\*38,50) | rpn\_cls\_prob |
| Reshape | rpn\_cls\_prob |  |  | (1,18,38,50) | rpn\_cls\_prob\_reshape |
| Convolution | relu5\_3(conv5\_3) | 1×1 | 1,0 | (1,36,38,50) | rpn\_bbox\_pred |
| Python | rpn\_bbox\_pred, rpn\_cls\_prob\_reshape |  |  | 产生最多300个roi，每个roi的数据格式为(flag,x1,y1,x2,y2) | proposal |
| ROIPooling | relu5\_3(conv5\_3)，proposal | 2×2 |  | (num\_rois,512,7,7) | roi\_pool5 |

proposal(Python)层操作： rpn\_bbox\_pred得到四个数值，并不是实际的坐标值，而是4个修改量（boundingbox regression），再根据上面计算的anchor boxes对应在原图像中的坐标，就可以计算回归得到的实际坐标(bounding boxes)：

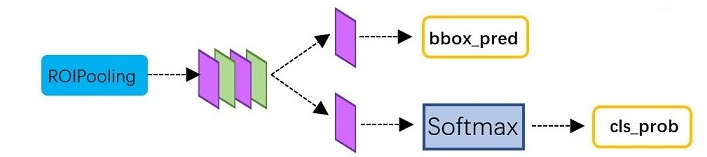


得到实际坐标之后，会存在一些坐标超出图像边界，此时删除这类boxes.

接着根据余下的bounding boxes对应的rpn\_bbox\_pred（是前景的概率）进行降序排序，根据设定的参数，提取概率较大的6000个box；然后根据nms（非极大值抑制,阈值设为0.7）再一次减小boxes的个数；最后再次根据前面的排序结果只留下300个box，用于后续的具体分类。

roi\_pool5层：将proposal层产生的实际坐标值映射到relu5\_3(conv5\_3)层，在feature map上产生不同区域大小的boxes，再经过ROIPooling操作得到固定长度的特征向量，我们这里固定为7\*7，所以每个roi经过该层，都会得到数据大小为（channels(512),7,7），整体的数据格式为(num\_rois,512,7,7)。

3、全连接网络结构：



|  |  |  |  |
| --- | --- | --- | --- |
| **Layer** | **input** | **output shape** | **name** |
| InnerProduct | roi\_pool5 | (num\_rois,4096) | fc6 |
| Relu | fc6 | (num\_rois,4096) | relu6 |
| InnerProduct | relu6 | (num\_rois,4096) | fc7 |
| Relu | fc7 | (num\_rois,4096) | relu7 |
| InnerProduct | relu7 | (num\_rois，类别数+1) | cls\_score |
| Softmax | cls\_score | (num\_rois，类别数+1) | cls\_prob |
| InnerProduct | relu7 | (num\_rois,类别数\*4) | bbox\_pred |

从PoI Pooling获取到7x7=49大小的proposal feature maps后，送入后续网络，可以看到做了2个操作：

1. 通过全连接和softmax对proposals进行分类，这实际上是深度学习的识别范畴了。
2. 再次对proposals进行bounding box regression，获取更高精度的rect box。