load\_data读了一次图片

datum

{‘id’:文件名,

‘h’: int,

‘w’: int,

‘gt\_boxes’:列表的列表，每一个内层列表是用int存储的x1,y1,x2,y2,

‘region\_proposals’：列表的列表，每一个内层列表是用int存储的x1,y1,x2,y2,

}

json是对data的序列化

data是datum的列表

db/images (num\_images, 1, max\_h, max\_w)

db/image\_mean 所有图像均值

db/image\_heights(num\_images)各个图像的高

db/image\_widths(num\_images)各个图像的宽

\_getitem\_\_

Img (1, 1, h, w)

Gt\_boxes ndarray (1, A, 4)-> tensor

Proposals (1, A, 4)

Ctrlf\_net中init\_weights会把cnn权重初始化

如果opt.init\_weights，那么定位和box\_scoring也会初始化

Basicblock是两个卷积加上residual，可能还有一个downsample

63 112 113 1 1 0

Make\_anchors x\_centers,y\_centers首先分别是特征图中的各个像素的横纵坐标，之后放大到原图

Localization层中所有的框是以xc, yc, w, h的形式存在

RPN层先经过公共的卷积conv\_layer,然后分别通过box\_conv\_layer输出4个坐标，rpn\_conv\_layer输出2个分数

默认是ResNet34

BasicBlock

由两个叠加的3x3卷积组成

Bottleneck

这里是三个卷积，分别是1x1,3x3,1x1,分别用来压缩维度，卷积处理，恢复维度，inplane是输入的通道数，plane是输出的通道数，expansion是对输出通道数的倍乘，在basic中expansion是1，输出的通道数就是plane，bottleneck中 plane不再代表输出的通道数，而是block内部压缩后的通道数，输出通道数变为plane\*expansion。

box\_to\_affine将bounding box转化为仿射变换矩阵

(xc, yc, w, h) ->

给出了边界框的大小和中心坐标，定义了由正则化后输出坐标系统到正则化输入系统的映射

特征图的坐标相对于输入图片的坐标做了平移和放缩

affine\_grid由放射变换矩阵得到采样的坐标

grid\_sample由采样的坐标得到采样结果

测试时网络结构

image->conv1->bn1->relu->maxpool->layer1->layer2->localization\_layer->roi\_boxes

roi\_boxes->\_eval\_helper->localization\_layer.eval\_boxes->layer3->layer4->bn2->relu->avgpool->fc->box\_scoring\_branch

eval\_boxes就是roi\_pooling

proposals->\_eval\_helper->…

返回roi\_boxes, proposals\_scores和roi\_scores

可以在训练时候不使用proposals因为score本质上是由feature\_map经过roi\_pooling得到的区域计算的