YOLO v3网络结构分析

先给出个链接，不想看文章的可以看下我在bilibili上的视频讲解：

1. YOLO系列理论视频合集
2. YOLOv3 SPP网络源码讲解（pytorch版）
3. YOLOv3 SPP网络代码仓库（pytorch版）

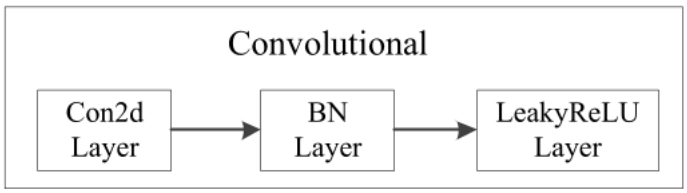
相信阅读了YOLO v3论文的小伙伴们会发现为什么这次的论文篇幅这么少？除去参考文献就四面？Excuse me？我是下了篇假文献吧。读完后感觉内容确实不多，而且总感觉写的不够细致，很多地方都比较模糊，可能是作者想让大家去观摩他的代码吧。

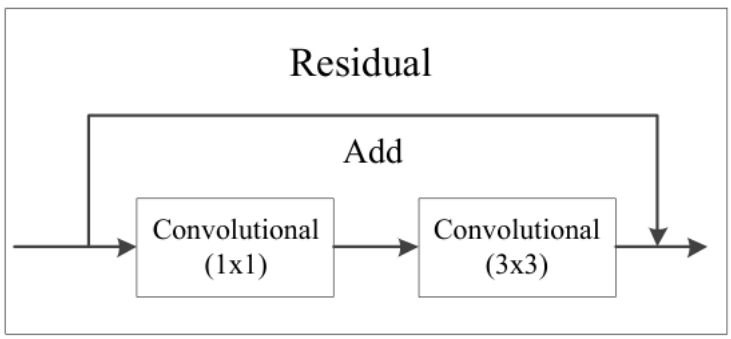
本人是小白，看后表示有点蒙。于是在Github上搜了大牛们基于Tensorflow搭建的YOLOv3模型进行分析（本人只接触过TF，所以就不去看caffe的源码了）。接下来我会根据我阅读的代码来进一步分析网络的结构。Github YOLOv3大牛代码**链接**。

1.Darknet-53 模型结构

在论文中虽然有给网络的图，但我还是简单说一下。这个网络主要是由一系列的1x1和3x3的卷积层组成（每个卷积层后都会跟一个BN层和一个LeakyReLU）层，作者说因为网络中有53个convolutional layers，所以叫做Darknet-53（2 + 1*2 + 1 + 2*2 + 1 + 8*2 + 1 + 8*2 + 1 + 4*2 + 1 = 53 按照顺序数，最后的Connected是全连接层也算卷积层，一共53个）。下图就是Darknet-53的结构图，在右侧标注了一些信息方便理解。（卷积的strides默认为（1，1），padding默认为same，当strides为（2，2）时padding为valid）

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

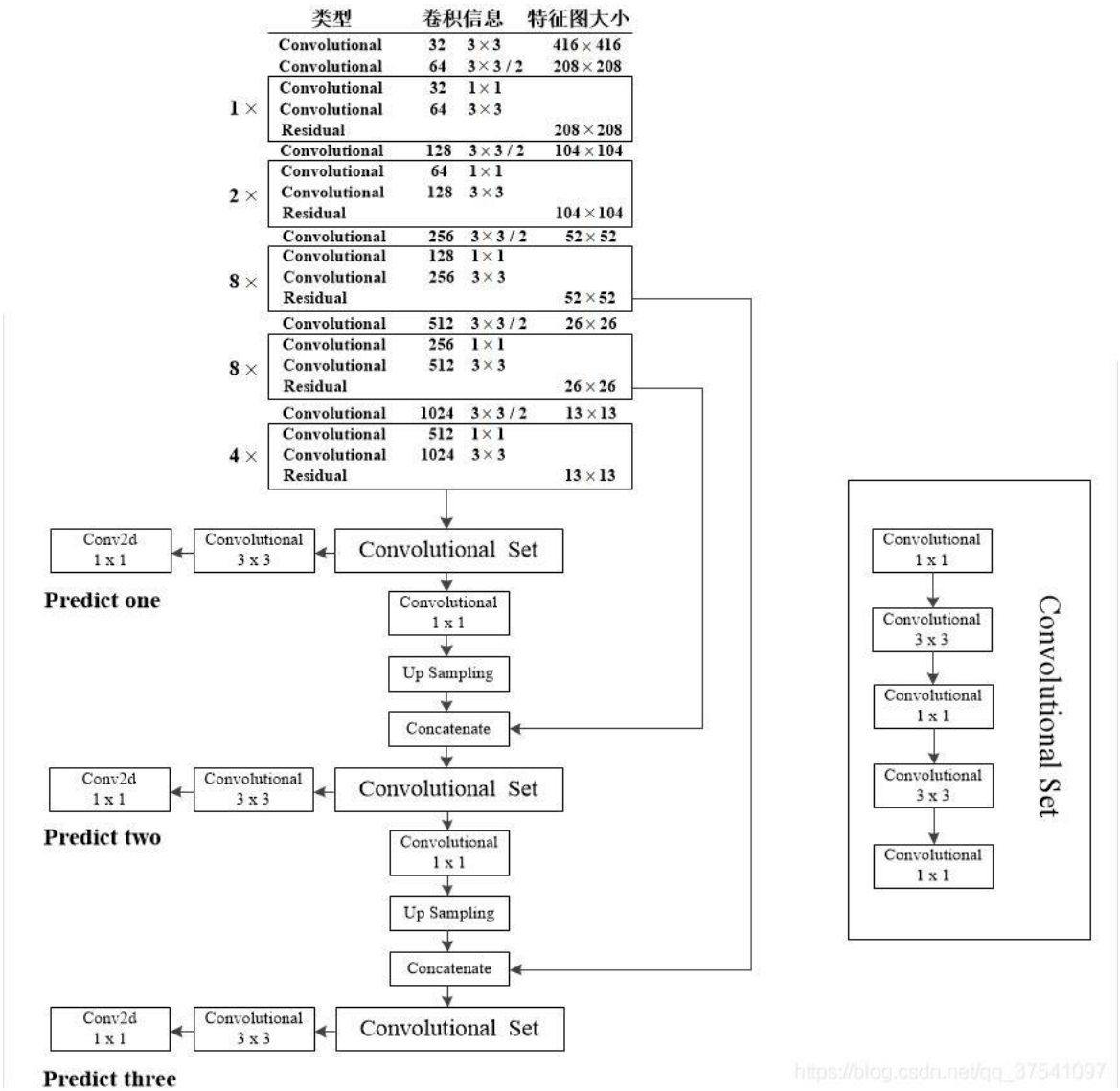




看完上图应该就能自己搭建出Darknet-53的网络结构了，上图是以输入图像256 x 256进行预训练来进行介绍的，常用的尺寸是416 x 416，都是32的倍数。下面我们再来分析下YOLOv3的特征提取器，看看究竟是在哪几层Features上做的预测。

2.YOLOv3 模型结构

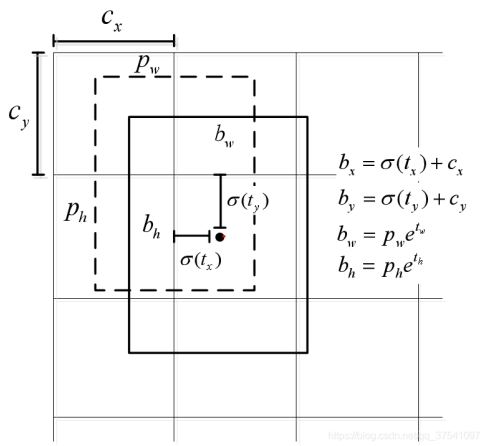
作者在论文中提到利用三个特征层进行边框的预测，具体在哪三层我感觉作者在论文中表述的并不清楚（例如文中有“添加几个卷积层”这样的表述），同样根据代码我将这部分更加详细的分析展示在下图中。**注意：原Darknet53中的尺寸是在图片分类训练集上训练的，所以输入的图像尺寸是256x256，下图是以YOLO v3 416模型进行绘制的，所以输入的尺寸是416x416，预测的三个特征层大小分别是52，26，13。**



在上图中我们能够很清晰的看到三个预测层分别来自的什么地方，以及Concatenate层与哪个层进行拼接。**注意Convolutional是指Conv2d+BN+LeakyReLU，和Darknet53图中的一样，而生成预测结果的最后三层都只是Conv2d。**通过上图小伙伴们就能更加容易地搭建出YOLOv3的网络框架了。

3.目标边界框的预测

YOLOv3网络在三个特征图中分别通过(4+1+c)× k个大小为1×1的卷积核进行卷积预测，k为预设边界框（bounding box prior）的个数（k默认取3），c为预测目标的类别数，其中4k个参数负责预测目标边界框的偏移量，k个参数负责预测目标边界框内包含目标的概率，ck个参数负责预测这k个预设边界框对应c个目标类别的概率。下图展示了目标边界框的预测过程（该图是本人重新绘制的，与论文中的示意图有些不同，个人感觉自己绘制的更便于理解）。图中虚线矩形框为预设边界框，实线矩形框为通过网络预测的偏移量计算得到的预测边界框。其中(c_x, c_y)为预设边界框在特征图上的中心坐标，(p_w, p_y)为预设边界框在特征图上的宽和高，(t_x, t_y, t_w, t_h)分别为网络预测的边界框中心偏移量(t_x, t_y)以及宽高缩放比(t_w, t_y)，(b_x, b_y, b_w, b_h)为最终预测的目标边界框，从预设边界框到最终预测边界框的转换过程如图右侧公式所示，其中$\sigma(x)$函数是sigmoid函数其目的是将预测偏移量缩放到0到1之间（这样能够将预设边界框的中心坐标固定在一个cell当中，作者说这样能够加快网络收敛）。



下图给出了三个预测层的特征图大小以及每个特征图上预设边界框的尺寸（这些预设边界框尺寸都是作者根据COCO数据集聚类得到的）：

特征图层	特征图大小	预设边界框尺寸	预设边界框数量
特征图层 1	13×13	（116×90）；（156×198）；（373×326）	13×13×3
特征图层 2	26×26	（30×61）；（62×45）；（59×119）	26×26×3
特征图层 3	52×52	（10×13）；（16×30）；（33×23）	52×52×3

4.损失函数的计算

关于YOLOv3的损失函数文章中写的很粗略，比如坐标损失采用的是误差的平方和，类别损失采用的是二值交叉熵，本人在github上也找了很多YOLO v3的公开代码，有的采用的是YOLOv1或者YOLOv2的损失函数，下面给出本人认为正确的损失函数（这里偷个懒，公式都是从本人之前写的论文中截图的）。

YOLOv3的损失函数主要分为三个部分：目标定位偏移量损失$L_{loc}(l,g)$，目标置信度损失$L_{conf}(o,c)$以及目标分类损失$L_{cla}(O,C)$,其中$\lambda_1,\lambda_2,\lambda_3$是平衡系数。

$$L(O,o,C,c,l,g)=\lambda_1 L_{conf}(o,c)+\lambda_2 L_{cla}(O,C)+\lambda_3 L_{loc}(l,g)$$

4.1目标置信度损失

目标置信度可以理解为预测目标矩形框内存在目标的概率，目标置信度损失$L_{conf}(o,c)$采用的是二值交叉熵损失(Binary Cross Entropy)，其中$o_i\in\{0,1\}$,表示预测目标边界框i中是否真实存在目标，0表示不存在，1表示存在。$\hat{c}_i$表示预测目标矩形框i内是否存在目标的Sigmoid概率（将预测值c_i通过sigmoid函数得到）。

$$L_{conf}(o,c)=-\sum(o_i\ln(\hat{c}_i)+(1-o_i)\ln(1-\hat{c}_i))$$

$$\hat{c}_i=Sigmoid(c_i)$$

4.2目标类别损失

目标类别损失$L_{cla}(O,C)$同样采用的是二值交叉熵损失（采用二值交叉熵损失的原因是，作者认为同一目标可同时归为多类，比如猫可归为猫类以及动物类，这样能够应对更加复杂的场景。但在本人实践过程中发现使用原始的多类别交叉熵损失函数效果会更好一点，原因是本人针对识别的目标都是固定归于哪一类的，并没有可同时归于多类的情况），其中$O_{ij}\in\{0,1\}$,表示预测目标边界框i中是否真实存在第j类目标，0表示不存在，1表示存在。$\hat{C}_{ij}$表示网络预测目标边界框i内存在第j类目标的Sigmoid概率（将预测值C_{ij}通过sigmoid函数得到）。

$$L_{cla}(O,C)=-\sum_{i\in Pos}\sum_{j\in cla}(O_{ij}\ln(\hat{C}_{ij})+(1-O_{ij})\ln(1-\hat{C}_{ij}))$$

$$\hat{C}_{ij}=Sigmoid(C_{ij})$$

4.3目标定位损失

目标定位损失$L_{loc}(l,g)$采用的是真实偏差值与预测偏差值差的平方和，其中$\hat{l}$表示预测矩形框坐标偏移量（注意网络预测的是偏移量，不是直接预测坐标），$\hat{g}$表示与之匹配的GTbox与默认框之间的坐标偏移量，(b^x,b^y,b^w,b^h)为预测的目标矩形框参数,(c^x,c^y,p^w,p^h)为默认矩形框参数,(g^x,g^y,g^w,g^h)为与之匹配的真实目标矩形框参数，这些参数都是映射在预测特征图上的。

$$L_{loc}(l,g)=\sum_{i\in pos}\sum_{m\in\{x,y,w,h\}}(\hat{l}_i^m-\hat{g}_i^m)^2$$

$$\hat{l}_i^x=b_i^x-c_i^x,\qquad \hat{l}_i^y=b_i^y-c_i^y$$

$$\hat{l}_i^w=\log(b_i^w/p_i^w),\quad \hat{l}_i^h=\log(b_i^h/p_i^h)$$

$$\hat{g}_i^x=g_i^x-c_i^x,\qquad \hat{g}_i^y=g_i^y-c_i^y$$

$$\hat{g}_i^w=\log(g_i^w/p_i^w),\ \hat{g}_i^h=\log(g_i^h/p_i^h)$$

先写这么多吧，以后如果有什么需要补充的在补充，有问题还请指出，谢谢！

相关推荐