YOLO9000:
Better, Faster, Stronger

----------A04 张公子

论文的三大部分

- Better
- Faster
- Stronger

前两个部分主要是对yolov2框架改进进行讲解，最后一部分才是讲的yolo9000

Better

- 1. Batch Normalization

  对网络的每一层的输入都做了归一化，这样能提高2%的map

- 2. High Resolution Classifie
- 原来的YOLO网络在预训练的时候采用的是224*224的输入，然后在detection的时候采用448*448的输入，这会导致从分类模型切换到检测模型的时候，模型还要适应图像分辨率的改变。
- 而YOLOv2则将预训练分成两步：先用224*224的输入从头开始训练网络，大概160个epoch，然后再将输入调整到448*448，再训练10个epoch。这两步都是在ImageNet数据集上操作。最后再在检测的数据集上fine-tuning，也就是detection的时候用448*448的图像作为输入就可以顺利过渡了。作者的实验表明这样可以提高几乎4%的MAP。

Better

- 3. Convolutional With Anchor Boxes
- 原来的YOLO是利用全连接层直接预测bounding box的坐标，而YOLOv2借鉴了Faster R-CNN的思想，引入anchor。
- 原YOLO一个grid_cell只能检测一个物体，这样一个grid_cell里有5个anchor，就能检测5个物体。
- 作者的实验证明：虽然加入anchor使得MAP值下降了一点（69.5降到69.2），但是提高了recall（81%提高到88%）。

- 4. Dimension Clusters
- 采用k-means的方式对训练集的bounding boxes做聚类
- k-means的距离函数这里采用：

$$d(\text{box}, \text{centroid}) = 1 - \text{IOU}(\text{box}, \text{centroid})$$

Box Generation	#	Avg IOU
Cluster SSE	5	58.7
Cluster IOU	5	61.0
Anchor Boxes [15]	9	60.9
Cluster IOU	9	67.2

Table 1: Average IOU of boxes to closest priors on VOC 2007. The average IOU of objects on VOC 2007 to their closest, unmodified prior using different generation methods. Clustering gives much better results than using hand-picked priors.

Better

- 5. Direct Location prediction
- 每个cell预测5个bounding box，然后每个bounding box预测5个值：tx，ty，tw，th和to（这里的to类似YOLOv1中的confidence）

$$
\begin{aligned}
b_x &= \sigma(t_x) + c_x \\
b_y &= \sigma(t_y) + c_y \\
b_w &= p_w e^{t_w} \\
b_h &= p_h e^{t_h} \\
Pr(\text{object}) * IOU(b, \text{object}) &= \sigma(t_o)
\end{aligned}
$$

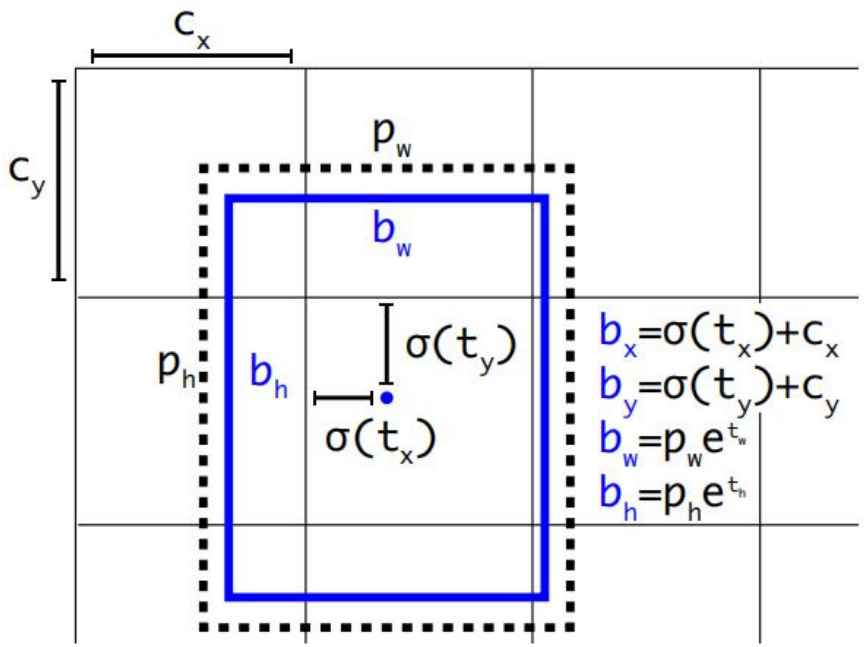

Figure 3: Bounding boxes with dimension priors and location prediction. We predict the width and height of the box as offsets from cluster centroids. We predict the center coordinates of the box relative to the location of filter application using a sigmoid function.

Better

- 6. Fine-Grained Features
- 这里主要是添加了一个层：passthrough layer。这个层的作用就是将前面一层的26*26的feature map和本层的13*13的feature map进行连接。

- 7. Multi-Scale Training
- 多尺度训练。注意这一步是在检测数据集上fine tune时候采用的，不要跟前面在Imagenet数据集上的两步预训练分类模型混淆。
- 具体来讲，在训练网络时，每训练10个batch，网络就会随机选择另一种size的输入。那么输入图像的size的变化范围要怎么定呢？前面我们知道本文网络本来的输入是416*416，最后会输出13*13的feature map，也就是说downsample的factor是32，因此作者采用32的倍数作为输入的size，具体来讲文中作者采用从{320,352,…,608}的输入尺寸。

Faster

- 1. Darknet-19
- Darknet-19只需要5.58 billion operation。这个网络包含19个卷积层和5个max pooling层，而在YOLO v1中采用的GooleNet，包含24个卷积层和2个全连接层，因此Darknet-19整体上卷积卷积操作比YOLO v1中用的GoogleNet要少，这是计算量减少的关键。

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Table 6: Darknet-19.

Faster

- 2. Training for Classification
- 主要分两步：1、从头开始训练Darknet-19，数据集是ImageNet，训练160个epoch，输入图像的大小是224*224，初始学习率为0.1。2、再fine-tuning 网络，这时候采用448*448的输入，参数的除了epoch和learning rate改变外，其他都没变，这里learning rate改为0.001，并训练10个epoch。
- 结果表明fine-tuning后的top-1准确率为76.5%，top-5准确率为93.3%，而如果按照原来的训练方式，Darknet-19的top-1准确率是72.9%，top-5准确率为91.2%。因此可以看出第1,2两步分别从网络结构和训练方式两方面入手提高了主网络的分类准确率。

Faster

- 3. Training for Detection
- 首先把最后一个卷积层去掉，然后添加3个3*3的卷积层，每个卷积层有1024个filter，而且每个后面都连接一个1*1的卷积层，1*1卷积的filter个数根据需要检测的类来定。

Stronger

- During training we mix images from both detection and classification datasets. When our network sees an image labelled for detection we can backpropagate based on the full YOLOv2 loss function. When it sees a classification image we only backpropagate loss from the classification specific parts of the architecture.