YOLO2——YOLO9000论文总结

整理：李英杰

论文地址：http://pjreddie.com/yolo9000/

> 论文参考点：
>
> - 1：如何对模型进行一系列分析和改进。
> - 2：多尺度的训练方法，模型可以适应不同的输入尺寸。
> - 3：综合不同类型的数据集。
> - 4：联合训练方法。
> - 5：YOLO端到端的实时检测和分类的思路。

YOLO2

介绍

> 1，YOLOv2在YOLOv1的基础上，使用新的网络结构（darknet19）和技巧（Batch Normalization、High Resolution Classifier、Convolutional With Anchor Boxes等），提高了检测速度和检测精度。
>
> 2、作者提出了一种联合训练方法，可以同时使用检测数据集和分类数据集来训练检测模型，用分层的观点对物体分类，用检测数据集学习准确预测物体的位置，用分类数据集来增加可识别的类别量，提升鲁棒性。
>
> 3、作者基于YOLOv2，采用wordTree的方法，综合ImageNet和COCO数据集训练YOLO9000，能够实时预测9000多类。，可以实时检测九千多种物体。

Better

> YOLO相较于其他的state-of-the-art的检测系统有一些**缺陷**，主要表现在两点：
>
> > 1、和Fast R-CNN相比，YOLO会产生较多的bounding boxes的定位错误。
> >
> > 2、和基于region proposal的检测系统相比，YOLO的Recall较低。

- **YOLO**的目标是高精度实时检测，所以期望在不增大网络、精度不下降的前提下来对定位错误和低Recall进行改善，为此作者尝试了一系列方法:

	YOLO							YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓	✓
anchor boxes?				✓	✓				
new network?					✓	✓	✓	✓	✓
dimension priors?						✓	✓	✓	✓
location prediction?						✓	✓	✓	✓
passthrough?							✓	✓	✓
multi-scale?								✓	✓
hi-res detector?									✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

Table 2: The path from YOLO to YOLOv2. Most of the listed design decisions lead to significant increases in mAP. Two exceptions are switching to a fully convolutional network with anchor boxes and using the new network. Switching to the anchor box style approach increased recall without changing mAP while using the new network cut computation by 33%.

1.1: **Batch Normalization** 可以提高模型收敛速度，减少过拟合。作者在所有卷积层应用了Batch Normalization，使结果提升了2% mAP。同时，Batch Normalization的应用，去除了dropout，也不会过拟合。

1.2: **High Resolution Classifier** 目前，所有state-of-the-art的检测方法都在ImageNet上对分类器进行了预训练。从AlexNet开始，多数分类器都把输入图像resize到256 * 256以下，这会容易丢失一些小物体的信息。

YOLOv1先使用224 * 224的分辨率来训练分类网络，在训练检测网络的时候再切换到448 * 448的分辨率，这意味着YOLOv1的卷积层要重新适应新的分辨率同时YOLOv1的网络还要学习检测网络。

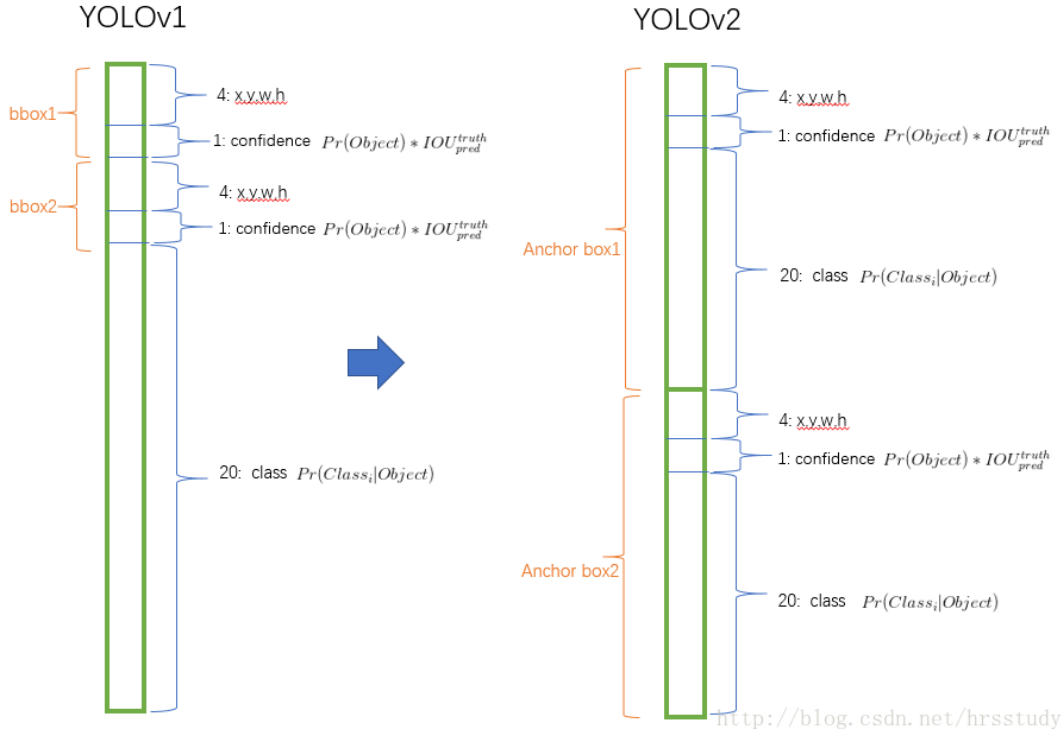
现在，YOLOv2直接使用**448 * 448**的分辨率来fine tune分类网络，好让网络可以调整filters来适应高分辨率。然后再用这个结果来fine tune检测网络。

使用高分辨率的分类网络提升了将近4%的mAP。

1.3: **Convolutional With Anchor Boxes**

> Convolutional With Anchor Boxes的由来：
>
> 1.YOLOv1使用卷积层之后的全连接层来直接预测bounding boxes的坐标。Faster R-CNN的做法和YOLO不同，使用精心挑选的anchor boxes来预测bounding boxes的坐标。
>
> 2.Faster R-CNN的region proposal network（RPN）使用全卷积网络来预测相对anchor boxes的offsets和confidences。因为预测层是个卷积层，RPN在一个特征图上预测所有bounding boxes的offsets。和直接预测坐标相比，预测offsets简化了问题，而且网络更容易学习。
>
> **YOLOv2去掉了全连接层，使用anchor boxes预测bounding boxes。**
>
> **YOLOv2去掉了一个池化层，使得卷积层的输出有更高的分辨率。**
>
> **YOLOv2将输入图像的尺寸从448 * 448缩减到416 * 416，这样特征图的输出就是一个奇数，有一个中心栅格。YOLOv2对图像进行了32倍的降采样，最终输出的特征图尺寸是13 * 13。因为作者观察到，大物体通常占据了图像的中间位置，可以只用一个中心的cell来预测这些物体的位置，否则就要用中间的4个cell来进行预测，这个技巧可稍稍提升效率。**

YOLOv2不再由栅格去预测条件类别概率，而由Bounding boxes去预测。在YOLOv1中每个栅格只有1组条件类别概率，而在YOLOv2中，因为每个栅格有B个bounding boxes，所以有B组条件类别概率。 在YOLOv1中输出的维度为S * S * （B * 5 + C），而YOLOv2为 S * S * （B * （5 + C））。



使用anchor boxes, 模型的的精度有一点点下降，但是Recall有大幅上升。没有anchor box，我们的中间模型的mAP为69.5，Recall为81%。使用anchor boxes 模型的mAP为69.2，Recall为88%。尽管mAP有轻微的下降，但是Recall的增加意味着模型有更多的改进空间。

1.4：**Dimension Clusters** YOLO上采用anchor boxes面临两个问题。1：box的维度是人工设定的，模型可以学会去适应boxes，但是如果我们更好的boxes。模型容易学习。采用K-means聚类方法来自动选择最佳的初始boxes。值得注意的是，因为使用欧氏距离会让大的bounding boxes比小的bounding boxes产生更多的error，而我们希望能通过anchor boxes获得好的IOU scores，并且IOU scores是与box的尺寸无关的。为此，作者定义了新的距离公式：

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

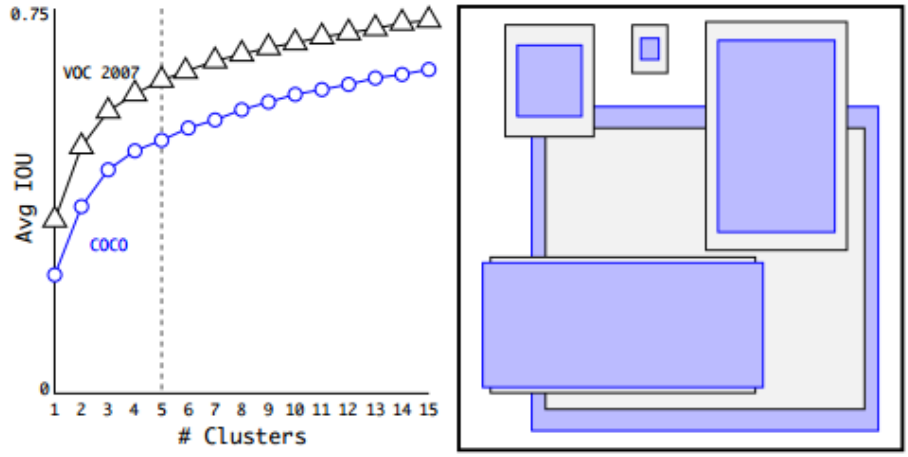


Figure 2: Clustering box dimensions on VOC and COCO. We run k-means clustering on the dimensions of bounding boxes to get good priors for our model. The left image shows the average IOU we get with various choices for k. We find that $k = 5$ gives a good tradeoff for recall vs. complexity of the model. The right image shows the relative centroids for VOC and COCO. Both sets of priors favor thinner, taller boxes while COCO has greater variation in size than VOC.

和使用手工挑选的anchor boxes相比，使用K-means得到的anchor boxes表现更好。使用5个k-means得到的anchor boxes的性能（IOU 61.0）和使用9个手工挑选的anchor boxes的性能（IOU 60.9）相当。这意味着使用k-means获取anchor boxes来预测bounding boxes让模型更容易学习如何预测bounding boxes。

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.

1.5 **Direct location prediction** 遇到的第二个问题就是**模型变得不稳定**。特别是前几轮的训练。模型随机初始化后需要很长一段时间才能稳定预测不确定因素来源于预测框的（x,y）。将预测偏移量改为YOLO的预测grid cell的位置匹配行（location coordinate）。将预测值限制在（0，1）范围内，以增强稳定性。网络对feature map中的每个cell预测5个Bounding boxes。对每一个Bounding box，模型预测5个匹配性值。

网络在特征图（13 *13 ）的每个cell上预测5个bounding boxes，每一个bounding box预测5个坐标值：tx，ty，tw，th，to。如果这个cell距离图像左上角的边距为（cx，cy）以及该cell对应的box维度（bounding box prior）的长和宽分别为（pw，ph），那么对应的box为：

the output feature map. The network predicts 5 coordinates for each bounding box, t_x, t_y, t_w, t_h, and t_o. If the cell is offset from the top left corner of the image by (c_x, c_y) and the bounding box prior has width and height p_w, p_h, then the predictions correspond to:

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

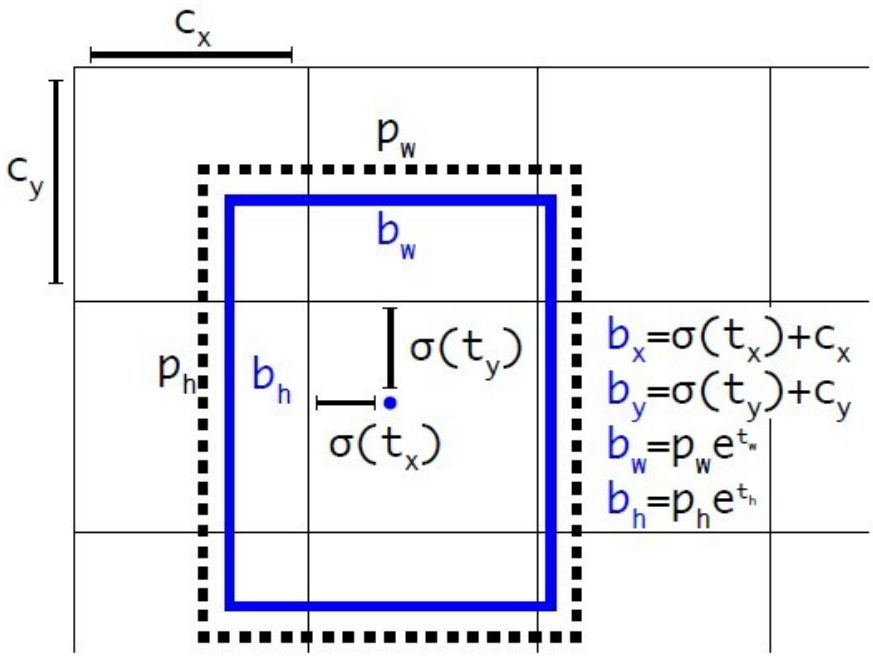
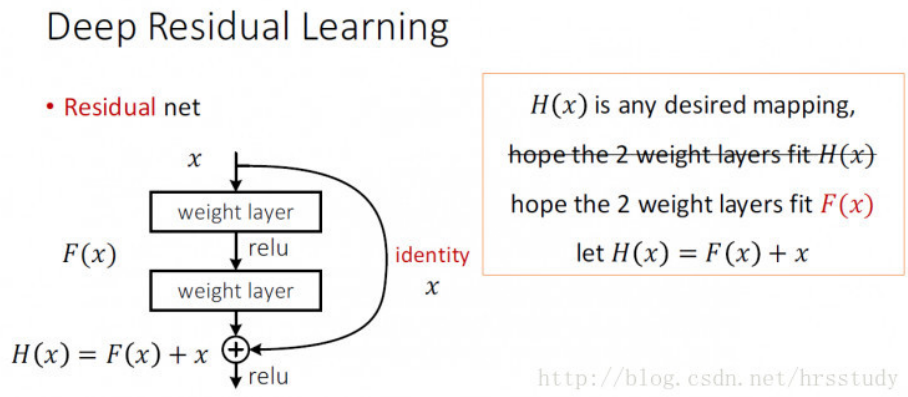


Figure 3: Bounding boxes with dimension priors and location prediction. We predict the width and height of the box as offsets from cluster centroids. We predict the center coordinates of the box relative to the location of filter application using a sigmoid function.

使用k-means得到的anchor boxes直接预测相对栅格的坐标，mAP提升了将近5%。

1.6 **Fine-Grained Features** 修改后的网络最终在13 * 13的特征图上进行预测，虽然这足以胜任大尺度物体的检测，如果用上细粒度特征的话可能对小尺度的物体检测有帮助。**Faser R-CNN和SSD都在不同层次的特征图上产生区域建议以获得多尺度的适应性。**YOLOv2使用了一种不同的方法，简单添加一个 passthrough layer，把浅层特征图（分辨率为26 * 26）连接到深层特征图。

passthroughlaye把高低分辨率的特征图做连结，叠加相邻特征到不同通道（而非空间位置），类似于Resnet中的identity mappings。这个方法把26 * 26 * 512的特征图叠加成13 * 13 * 2048的特征图，与原生的深层特征图相连接。



YOLOv2的检测器使用的就是经过扩展后的的特征图，它可以使用细粒度特征，使得模型的性能获得了1%的提升。

1.7**Multi-Scale Training** 模型只含有卷积层和池化层，可以改变输入尺寸。在训练时，每隔几轮便改变输入尺寸，增加模型可以对不同的尺寸的鲁棒性，每隔10batches 就随机改变输入图像的尺寸 following multiples of 32: $\{320, 352, ..., 608\}$.
这样网络可以更好地预测不同尺寸的图像，在小尺寸图像上运行更快，在速度和精度上达到了平衡。

在低分辨率的图像上，YOLO2是检测速度快（计算消耗低），精度较高的检测器。输入为228 * 228的时候，帧率达到90FPS，mAP几乎和Faster R-CNN的水准相同。使得其更加适用于低性能GPU、高帧率视频和多路视频场景。

在高分辨率图片检测中，YOLOv2达到了先进水平（state-of-the-art），VOC2007 上mAP为78.6%，而且超过实时速度要求。下图是YOLOv2和其他网络在VOC2007上的对比：

Detection Frameworks	Train	mAP	FPS
Fast R-CNN [5]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[15]	2007+2012	73.2	7
Faster R-CNN ResNet[6]	2007+2012	76.4	5
YOLO [14]	2007+2012	63.4	45
SSD300 [11]	2007+2012	74.3	46
SSD500 [11]	2007+2012	76.8	19
YOLOv2 288 × 288	2007+2012	69.0	91
YOLOv2 352 × 352	2007+2012	73.7	81
YOLOv2 416 × 416	2007+2012	76.8	67
YOLOv2 480 × 480	2007+2012	77.8	59
YOLOv2 544 × 544	2007+2012	**78.6**	40

Table 3: Detection frameworks on PASCAL VOC 2007. YOLOv2 is faster and more accurate than prior detection methods. It can also run at different resolutions for an easy tradeoff between speed and accuracy. Each YOLOv2 entry is actually the same trained model with the same weights, just evaluated at a different size. All timing information is on a Geforce GTX Titan X (original, not Pascal model).

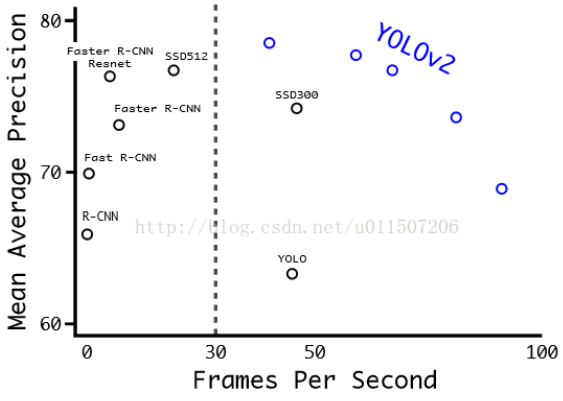


Figure 4: Accuracy and speed on VOC 2007.

Faster

- 大部分的检测框架基于VGG-16作为特征提取网络，但其比较复杂，耗费计算量大.
  YOLO使用的是基于Googlenet的自定制网络，比VGG-16更快，计算量比VGG-16小，准确率比VGG-16略低。224 * 224图片取 single-crop, top-5 accuracy，YOLO的定制网络得到88%（VGG-16得到90%）。

Darknet-19

- YOLOv2使用了一个新的分类网络作为特征提取部分，参考了前人的工作经验。类似于VGG，网络使用了较多的3 * 3卷积核，在每一次池化操作后把通道数翻倍。借鉴了network in network的思想，网络使用了全局平均池化（global average pooling）做预测，把1 * 1的卷积核置于3 * 3的卷积核之间，用来压缩特征。使用batch normalization稳定模型训练，加速收敛，正则化模型。

- 最终得出的基础模型就是Darknet-19，包含19个卷积层、5个最大值池化层（max pooling layers ）。Darknet-19处理一张照片需要55.8亿次运算，imagenet的top-1准确率为72.9%，top-5准确率为91.2%。

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.

训练分类

- 作者采用ImageNet1000类数据集来训练分类模型。训练了160次，用随机梯度下降法，starting learning rate 为0.1，polynomial rate decay 为4，weight decay为0.0005 ，momentum 为0.9。训练的时候仍然使用了很多常见的数据扩充方法（data augmentation），包括random crops, rotations, and hue, saturation, and exposure shifts。

- 初始的224 * 224训练后把分辨率上调到了448 * 448，使用同样的参数又训练了10次，学习率调整到了10^{-3}。高分辨率下训练的分类网络top-1准确率76.5%，top-5准确率93.3%。

训练检测

- 为了把分类网络改成检测网络，去掉原网络最后一个卷积层，增加了三个 3 * 3 （1024 filters）的卷积层，并且在每一个卷积层后面跟一个1 * 1的卷积层，输出维度是检测所需数量。对于VOC数据集，我们需要预测5个boxes，每个boxes包含5个坐标值和20类别。因此，输出为125（5 * 20+5 * 5）。最后还加入了passthough 层，从最后3 * 3 * 512的卷积层连到倒数第二层，使模型有了细粒度特征。

```
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   416 x 416 x   3   ->   416 x 416 x  32
    1 max          2 x 2 / 2   416 x 416 x  32   ->   208 x 208 x  32
    2 conv     64  3 x 3 / 1   208 x 208 x  32   ->   208 x 208 x  64
    3 max          2 x 2 / 2   208 x 208 x  64   ->   104 x 104 x  64
    4 conv    128  3 x 3 / 1   104 x 104 x  64   ->   104 x 104 x 128
    5 conv     64  1 x 1 / 1   104 x 104 x 128   ->   104 x 104 x  64
    6 conv    128  3 x 3 / 1   104 x 104 x  64   ->   104 x 104 x 128
    7 max          2 x 2 / 2   104 x 104 x 128   ->    52 x  52 x 128
    8 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256
    9 conv    128  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x 128
   10 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256
   11 max          2 x 2 / 2    52 x  52 x 256   ->    26 x  26 x 256
   12 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x  26 x 512
   13 conv    256  1 x 1 / 1    26 x  26 x 512   ->    26 x  26 x 256
   14 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x  26 x 512
   15 conv    256  1 x 1 / 1    26 x  26 x 512   ->    26 x  26 x 256
   16 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x  26 x 512
   17 max          2 x 2 / 2    26 x  26 x 512   ->    13 x  13 x 512
   18 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024
   19 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x  13 x 512
   20 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024
   21 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x  13 x 512
   22 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024
   23 conv   1024  3 x 3 / 1    13 x  13 x1024   ->    13 x  13 x1024
   24 conv   1024  3 x 3 / 1    13 x  13 x1024   ->    13 x  13 x1024
   25 route  16
   26 reorg              / 2    26 x  26 x 512   ->    13 x  13 x2048
   27 route  26 24
   28 conv   1024  3 x 3 / 1    13 x  13 x3072   ->    13 x  13 x1024
   29 conv    510  1 x 1 / 1    13 x  13 x1024   ->    13 x  13 x 510
   30 detection
```


- 学习策略是：先以10^{-3}的初始学习率训练了160次，在第60次和第90次的时候学习率减为原来的十分之一。weight decay为0.0005，momentum为0.9，以及类似于Faster-RCNN和SSD的数据扩充（data augmentation）策略： random crops, color shifting, etc。使用相同的策略在 COCO 和VOC上训练。

stronger ——YOLO9000

联合数据集

- 提出了一种在分类数据集和检测数据集上联合训练的机制。使用检测数据集的图片去学习检测相关的信息，例如bounding box 坐标预测，是否包含物体以及属于各个物体的概率。使用仅有类别标签的分类数据集图片去扩展可以检测的种类。

- 训练过程中把检测数据和分类数据混合在一起。当网络遇到一张属于检测数据集的图片就基于YOLOv2的全部损失函数（包含分类部分和检测部分）做反向传播。当网络遇到一张属于分类数据集的图片就仅基于分类部分的损失函数做反向传播。

面临难点

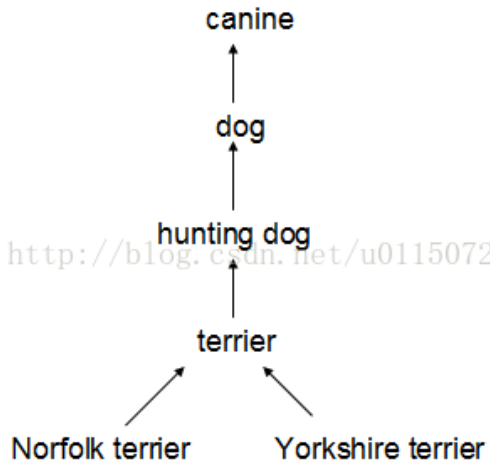
- 检测数据集只有常见物体和抽象标签（不具体），例如 “狗”，“船”。分类数据集拥有广而深的标签范围（例如ImageNet就有一百多类狗的品种，包括 “Norfolk terrier”, “Yorkshire terrier”, and “Bedlington terrier”等. ）。必须按照某种一致的方式来整合两类标签。

- 大多数分类的方法采用softmax层，考虑所有可能的种类计算最终的概率分布。但是softmax假设类别之间互不包含，但是整合之后的数据是类别是有包含关系的，例如 “Norfolk terrier” 和 “dog”。 所以整合数据集设法使用这种方式（softmax 模型）。

解决方法

Hierarchical classification（层次式分类）

- ImageNet的标签参考WordNet（一种结构化概念及概念之间关系的语言数据库）。例如:



- 很多分类数据集采用扁平化的标签。而整合数据集则需要结构化标签。
- WordNet是一个有向图结构（而非树结构），为了简化问题，作者从ImageNet的概念中构建了一个层次树结构（hierarchical tree）来代替图结构方案。

> 创建层次树的步骤：
>
> - 遍历ImageNet的所有视觉名词
> - 对每一个名词，在WordNet上找到从它所在位置到根节点（“physical object”）的路径。 许多同义词集只有一条路径。所以先把这些路径加入层次树结构。
> - 然后迭代检查剩下的名词，得到路径，逐个加入到层次树。路径选择办法是：如果一个名词有两条路径到根节点，其中一条需要添加3个边到层次树，另一条仅需添加一条边，那么就选择添加边数少的那条路径。

最终结果是一颗 WordTree （视觉名词组成的层次结构模型）。用WordTree执行分类时，预测每个节点的条件概率。例如： 在“terrier”节点会预测：

$$Pr(\text{Norfolk terrier}|\text{terrier})$$
$$Pr(\text{Yorkshire terrier}|\text{terrier})$$
$$Pr(\text{Bedlington terrier}|\text{terrier})$$
$$\dots$$

- 如果想求得特定节点的绝对概率，只需要沿着路径做连续乘积。例如 如果想知道一张图片是不是“Norfolk terrier ”需要计算：

$$\begin{aligned}Pr(\text{Norfolk terrier}) &= Pr(\text{Norfolk terrier}|\text{terrier}) \\ &\quad *Pr(\text{terrier}|\text{hunting dog}) \\ &\quad *\dots \\ &\quad *Pr(\text{mammal}|Pr(\text{animal}) \\ &\quad *Pr(\text{animal}|\text{physical object})\end{aligned}$$

> 注意
> 分类时假设 图片包含物体：Pr(physical object) = 1。

- 为了验证这种方法作者在WordTree（用1000类别的ImageNet创建）上训练了Darknet-19模型。为了创建WordTree1k作者天添加了很多中间节点，把标签由1000扩展到1369。训练过程中ground truth标签要顺着向根节点的路径传播：例如 如果一张图片被标记为“Norfolk terrier”它也被标

记为“dog” 和“mammal”等。为了计算条件概率，模型预测了一个包含1369个元素的向量，而且基于所有“同义词集”计算softmax，其中“同义词集”是同一概念的下位词。如下图所示，之前的ImageNet分类是使用一个大softmax进行分类。而现在，**WordTree只需要对同一概念下的同义词进行softmax分类。**

- 使用相同的训练参数，这种分层结构的Darknet19达到71.9%top-1精度和90.4%top-5精确度，精度只有微小的下降。

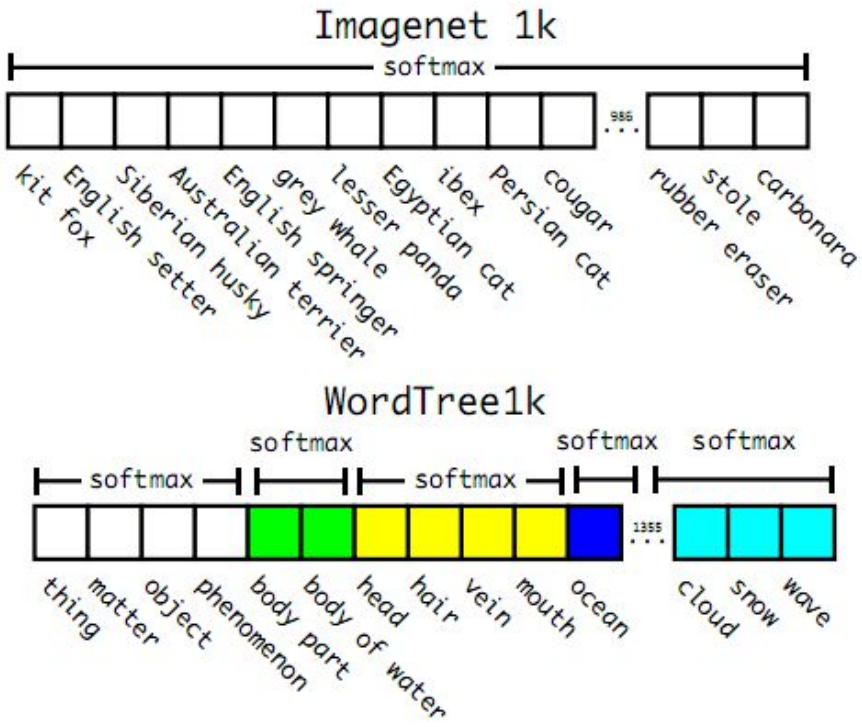


Figure 5: Prediction on ImageNet vs WordTree. Most ImageNet models use one large softmax to predict a probability distribution. Using WordTree we perform multiple softmax operations over co-hyponyms.

- 这个策略也同样可用于检测。不在假设每一张图片都包含物体，取而代之使用YOLOv2的物体预测器（objectness predictor）得到Pr(physical|object)的值。检测器预测一个bounding box和概率树（WordTree）。沿着根节点向下每次都走置信度最高的分支直到达到某个阈值，最终预测物体的类别为最后的节点类别。

Dataset combination with WordTree

- 可以使用WordTree把多个数据集整合在一起。只需要把数据集中的类别映射到树结构中的同义词集合（synsets）。使用WordTree整合ImageNet和COCO的标签如下图：

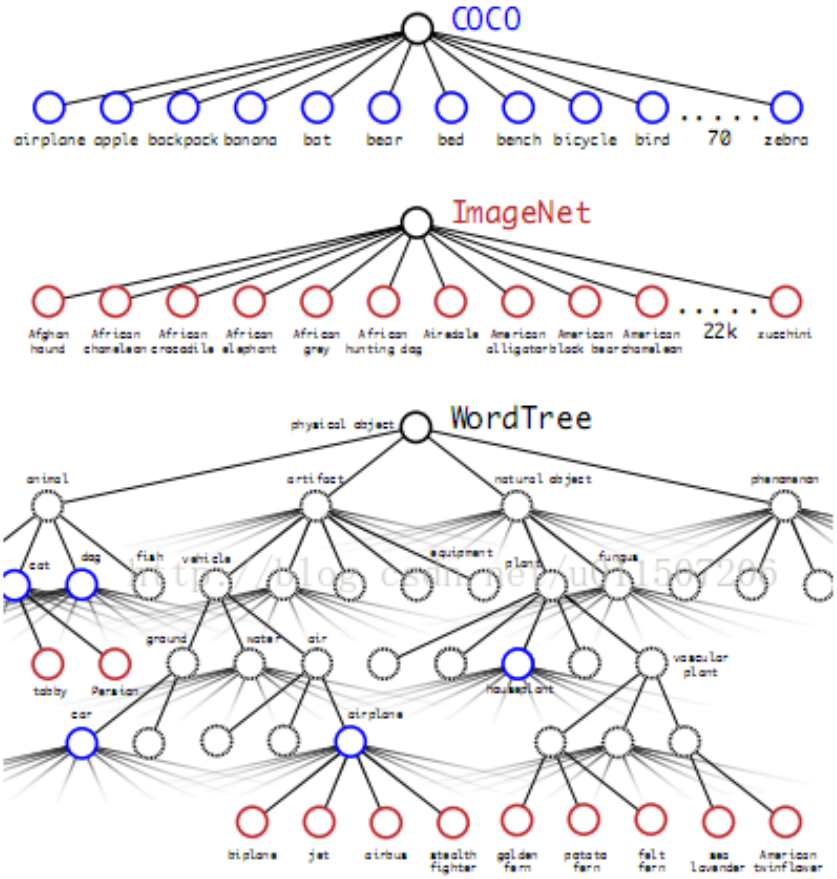


Figure 6: Combining datasets using WordTree hierarchy. Using the WordNet concept graph we build a hierarchical tree of visual concepts. Then we can merge datasets together by mapping the classes in the dataset to synsets in the tree. This is a simplified view of WordTree for illustration purposes.

joint classification and detection(联合训练分类和检测)

- 使用WordTree整合了数据集之后就可以在数据集（分类-检测数据）上训练联合模型。我们想要训练一个检测类别很大的检测器所以使用COCO检测数据集和全部ImageNet的前9000类创造一个联合数据集。为了评估我们使用的方法，也从ImageNet detection challenge 中向整合数据集添加一些还没有存在于整合数据集的类别。相应的WordTree有9418个类别。由于ImageNet是一个非常大的数据集，所以通过oversampling COCO数据集来保持平衡，使ImageNet：COCO = 4：1。

使用上面的数据集训练YOLO9000。

- 采用基本YOLOv2的结构，anchor box数量由5调整为3用以限制输出大小。
- 当网络遇到一张检测图片就正常反向传播。其中对于分类损失只在当前及其路径以上对应的节点类别上进行反向传播。
- 当网络遇到一张分类图片仅反向传播分类损失。在该类别对应的所有bounding box中找到一个置信度最高的（作为预测坐标），同样只反向传播该类及其路径以上对应节点的类别损失。反向传播objectness损失基于如下假设：预测box与ground truth box的重叠度至少0.31IOU。

采用这种联合训练，YOLO9000从COCO检测数据集中学习如何在图片中寻找物体，从ImageNet数据集中学习更广泛的物体分类。