

深度学习-目标检测篇

作者：神秘的wz

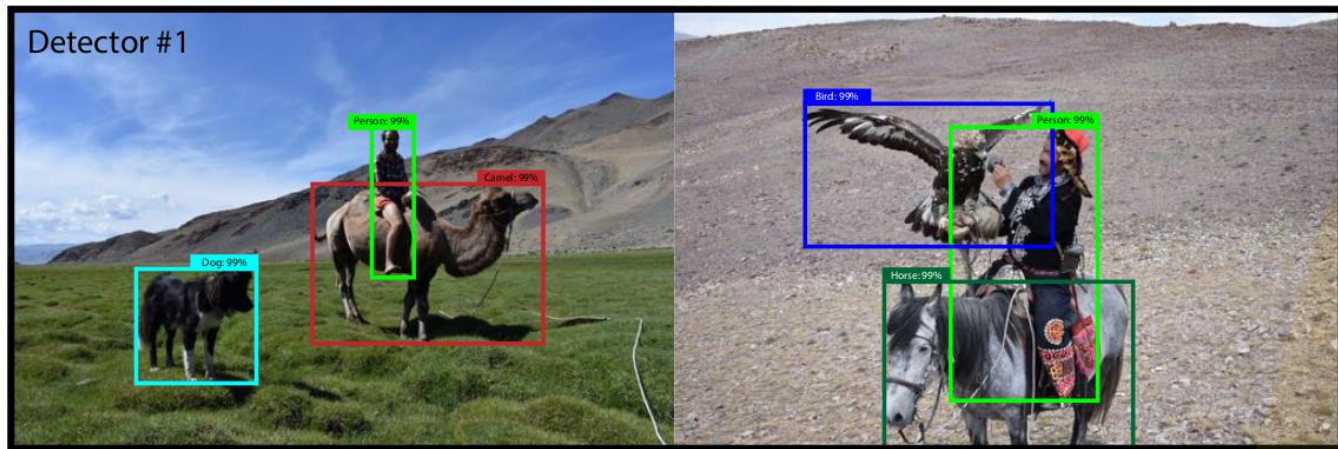
YOLO v3

YOLOv3: An Incremental Improvement

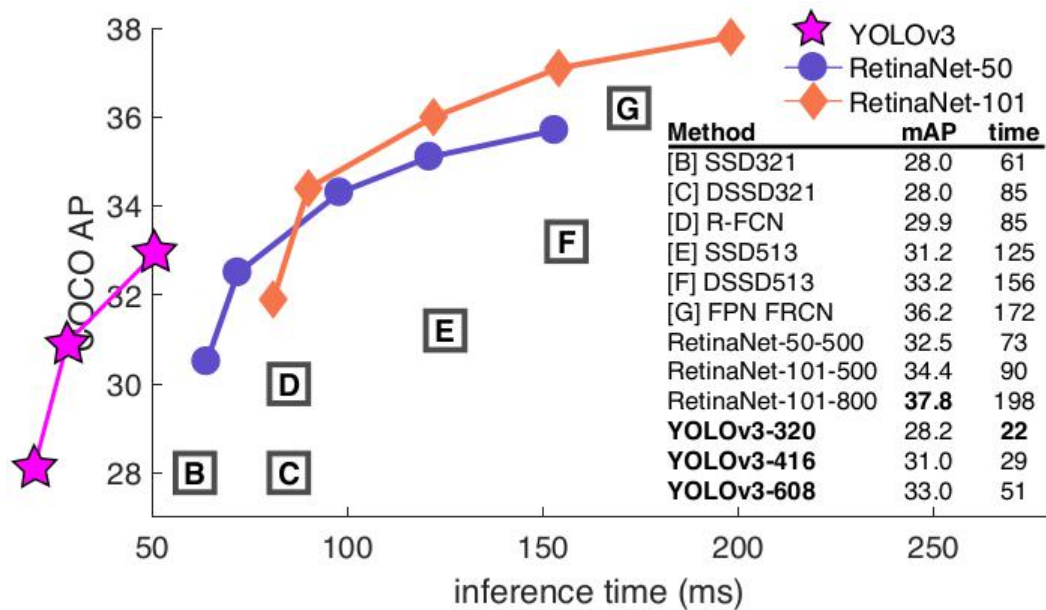
做大做强，再创辉煌

Joseph Redmon Ali Farhadi
University of Washington

2018 CVPR

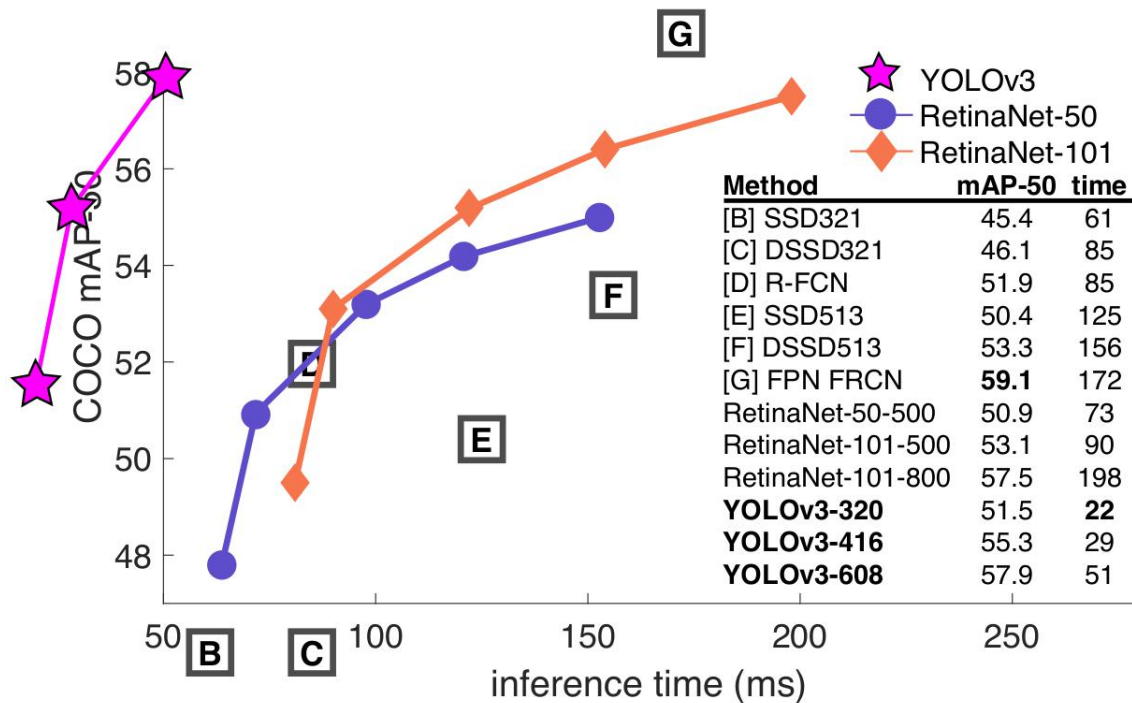


YOLO v3



COCO AP

YOLO v3



COCO AP IOU=0.5

YOLO v3

Darknet-53

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

Table 1. **Darknet-53.**

Backbone	Top-1	Top-5	Bn Ops	BFLOP/s	FPS
Darknet-19 [15]	74.1	91.8	7.29	1246	171
ResNet-101[5]	77.1	93.7	19.7	1039	53
ResNet-152 [5]	77.6	93.8	29.4	1090	37
Darknet-53	77.2	93.8	18.7	1457	78

Table 2. **Comparison of backbones.** Accuracy, billions of operations, billion floating point operations per second, and FPS for various networks.

$$\begin{aligned}
 &2+ \\
 &(1 \times 2) + 1 + \\
 &(2 \times 2) + 1 + \\
 &(8 \times 2) + 1 + \\
 &(8 \times 2) + 1 + \\
 &(4 \times 2) + 1 = 53
 \end{aligned}$$

YOLO v3

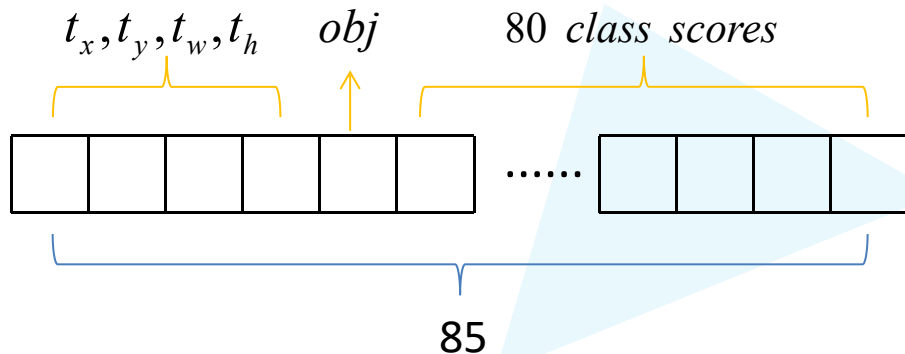
YOLO V3 model structure

Next we take the feature map from 2 layers previous and upsample it by $2\times$. We also take a feature map from earlier in the network and merge it with our upsampled features using concatenation. This method allows us to get more meaningful semantic information from the upsampled features and finer-grained information from the earlier feature map. We then add a few more convolutional layers to process this combined feature map, and eventually predict a similar tensor, although now twice the size.

We perform the same design one more time to predict boxes for the final scale. Thus our predictions for the 3rd scale benefit from all the prior computation as well as fine-grained features from early on in the network.

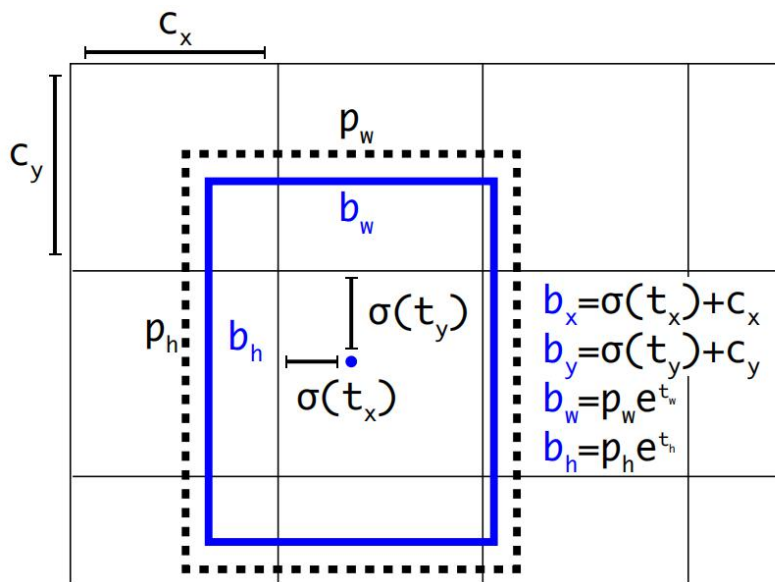
We still use k-means clustering to determine our bounding box priors. We just sort of chose 9 clusters and 3 scales arbitrarily and then divide up the clusters evenly across scales. On the COCO dataset the 9 clusters were: (10×13) , (16×30) , (33×23) , (30×61) , (62×45) , (59×119) , (116×90) , (156×198) , (373×326) .

YOLOv3 predicts boxes at 3 different scales. Our system extracts features from those scales using a similar concept to feature pyramid networks [8]. From our base feature extractor we add several convolutional layers. The last of these predicts a 3-d tensor encoding bounding box, objectness, and class predictions. In our experiments with COCO [10] we predict 3 boxes at each scale so the tensor is $N \times N \times [3 * (4 + 1 + 80)]$ for the 4 bounding box offsets, 1 objectness prediction, and 80 class predictions.



YOLO v3

目标边界框的预测



Following YOLO9000 our system predicts bounding boxes using dimension clusters as anchor boxes [15]. The network predicts 4 coordinates for each bounding box, t_x, t_y, t_w, t_h . 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:

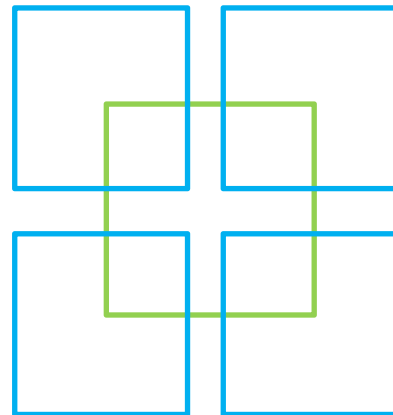
$$\sigma(x) = \text{Sigmoid}(x)$$

将预测的边界框中心
限制在当前cell中

YOLO v3

正负样本的匹配

YOLOv3 predicts an objectness score for each bounding box using logistic regression. This should be 1 if the bounding box prior overlaps a ground truth object by more than any other bounding box prior. If the bounding box prior is not the best but does overlap a ground truth object by more than some threshold we ignore the prediction, following [17]. We use the threshold of .5. Unlike [17] our system only assigns one bounding box prior for each ground truth object. If a bounding box prior is not assigned to a ground truth object it incurs no loss for coordinate or class predictions, only objectness.



YOLO v3

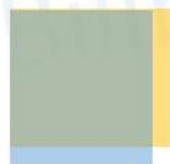
GT

Anchor Template



AT 1

$IoU = 0.25$



AT 2

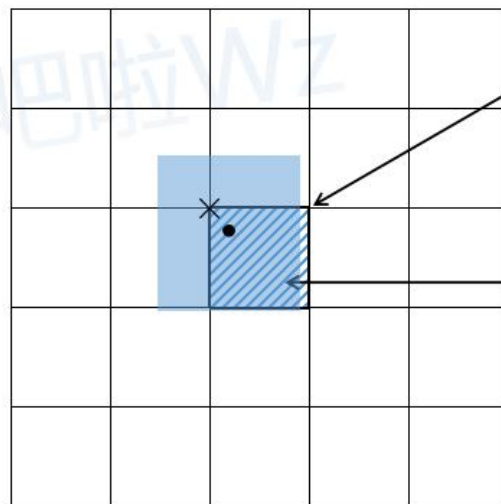
$IoU = 0.7$



AT 3

$IoU = 0.2$

$IoU > 0.3$



GT中心点落入该Cell
则该Cell中的AT 2为正样本

GT映射在Grid网络上

Grid网络

YOLO v3

损失的计算

$$L(o, c, O, C, l, g) = \overset{\text{置信度损失}}{\lambda_1 L_{conf}(o, c)} + \overset{\text{分类损失}}{\lambda_2 L_{cla}(O, C)} + \overset{\text{定位损失}}{\lambda_3 L_{loc}(l, g)}$$

$\lambda_1, \lambda_2, \lambda_3$ 为平衡系数

置信度损失

Binary Cross Entropy

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

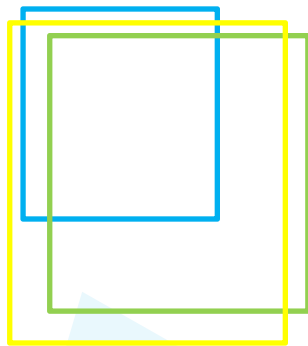
$$\hat{c}_i = \text{Sigmoid}(c_i)$$

可能和原文有出入

其中 $o_i \in [0, 1]$, 表示预测目标边界框与真实目标边界框的 IOU ,

c 为预测值, \hat{c}_i 为 c 通过 $Sigmoid$ 函数得到的预测置信度,

N 为正负样本个数



YOLOv3 predicts an objectness score for each bounding box using logistic regression. This should be 1 if the bounding box prior overlaps a ground truth object by more than any other bounding box prior. If the bounding box prior

类别损失

Binary Cross Entropy

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

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

其中 $O_{ij} \in \{0, 1\}$, 表示预测目标边界框 i 中是否存在第 j 类目标,

C_{ij} 为预测值, \hat{C}_{ij} 为 C_{ij} 通过 *Sigmoid* 函数得到的目标概率

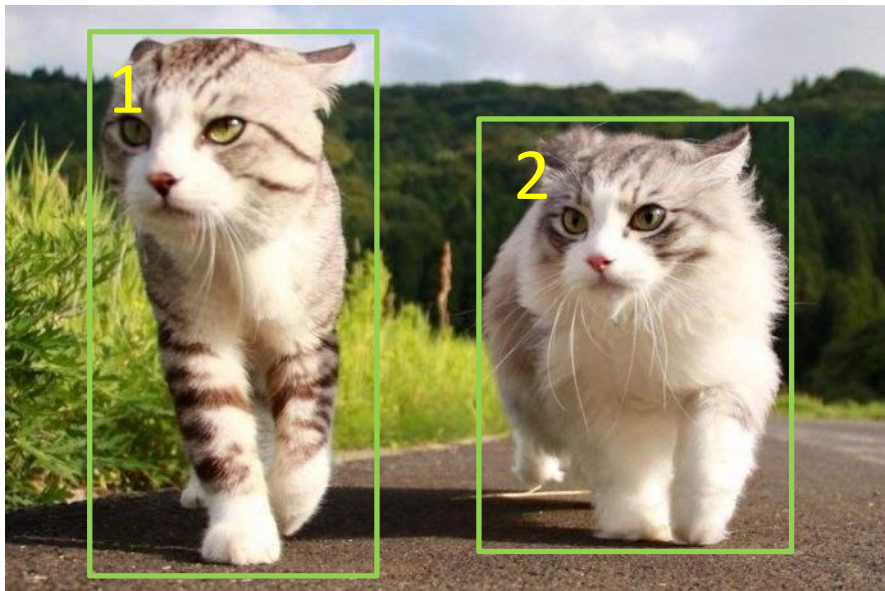
N_{pos} 为正样本个数

Each box predicts the classes the bounding box may contain using multilabel classification. We do not use a softmax as we have found it is unnecessary for good performance, instead we simply use independent logistic classifiers. During training we use binary cross-entropy loss for the class predictions.

YOLO v3

类别损失

[老虎, 豹子, 猫]



真实标签:

目标1: [0., 0., 1.]

目标2: [0., 0., 1.]

预测概率:

目标1: [0.1, 0.8, 0.9]

目标2: [0.2, 0.7, 0.8]

经过Sigmoid处理

YOLO v3

定位损失

$$L_{loc}(t, g) = \frac{\sum_{i \in pos} (\sigma(t_x^i) - \hat{g}_x^i)^2 + (\sigma(t_y^i) - \hat{g}_y^i)^2 + (t_w^i - \hat{g}_w^i)^2 + (t_h^i - \hat{g}_h^i)^2}{N_{pos}}$$

$$\hat{g}_x^i = g_x^i - c_x^i$$

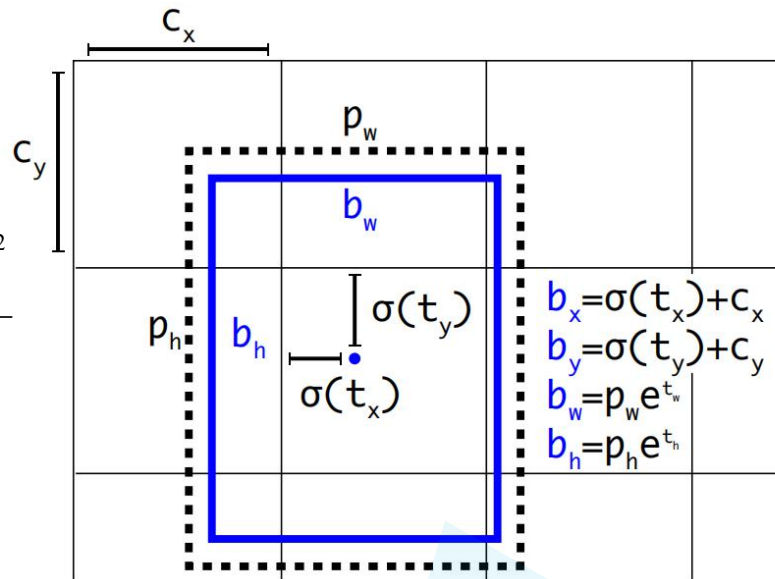
$$\hat{g}_y^i = g_y^i - c_y^i$$

$$\hat{g}_w^i = \ln(g_w^i / p_w^i)$$

$$\hat{g}_h^i = \ln(g_h^i / p_h^i)$$

t_x, t_y, t_w, t_h : 为网络预测的回归参数

g_x, g_y, g_w, g_h : 为GT 中心点的坐标
 x, y 以及宽度和高度
(映射在Grid网格中的)



During training we use sum of squared error loss. If the ground truth for some coordinate prediction is \hat{t}_* our gradient is the ground truth value (computed from the ground truth box) minus our prediction: $\hat{t}_* - t_*$. This ground truth value can be easily computed by inverting the equations above.

沟通方式

1.bilibili

<https://space.bilibili.com/18161609/channel/index>

2.CSDN

https://blog.csdn.net/qq_37541097/article/details/103482003

3.github

<https://github.com/WZMIAOMIAO/deep-learning-for-image-processing>

尽可能每周更新