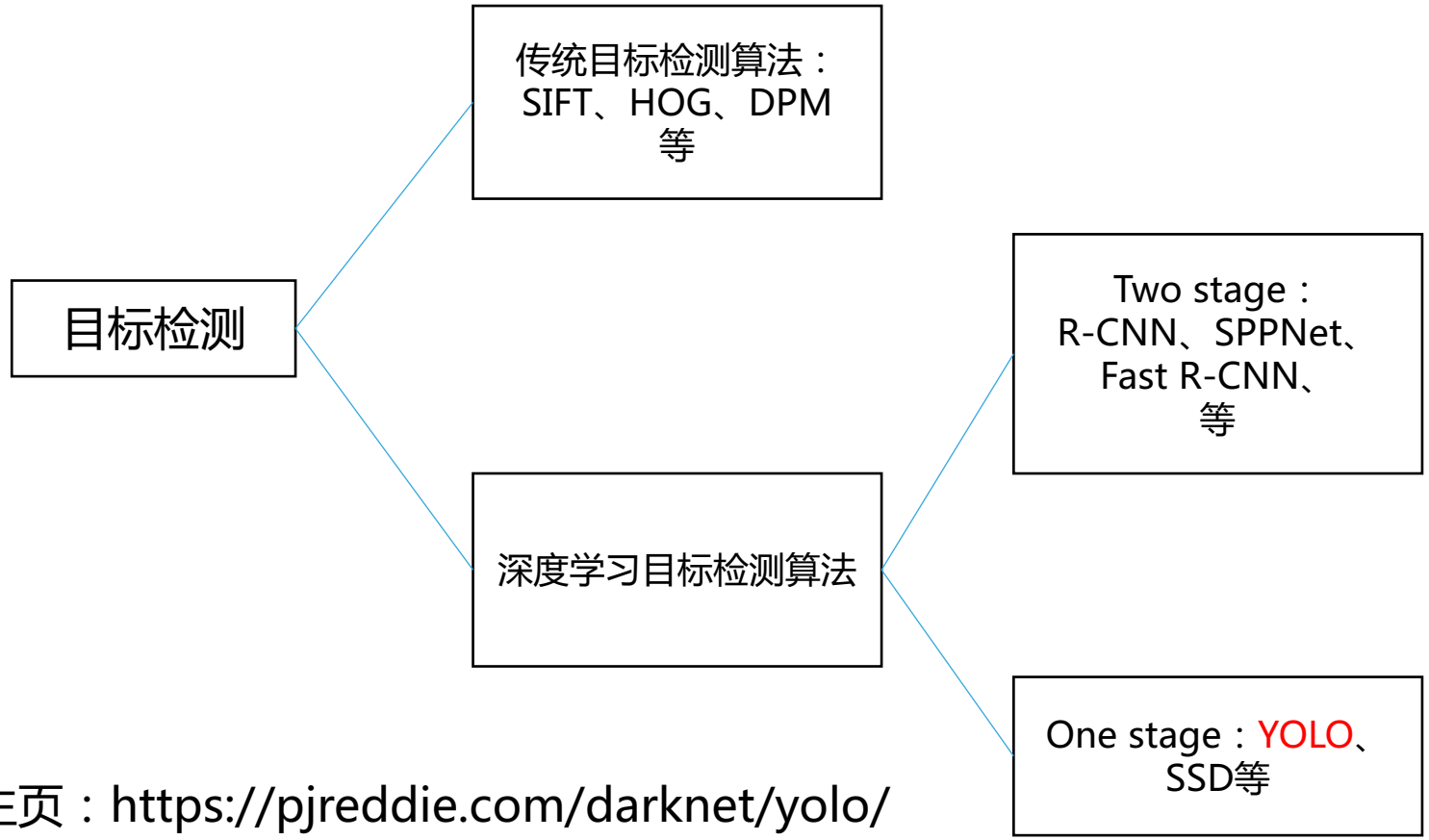
一种实时目标检测算法——YOLO (You Only Look Once)

- 报告人： 周航

- 报告时间：2018年5月3日

目标检测算法



项目主页：https://pjreddie.com/darknet/yolo/

文章地址：https://arxiv.org/pdf/1506.02640v5.pdf

发展趋势

OverFeat 2013

由Yann Lecun 提出，其利用滑动窗口和规则块生成候选框，再利用多尺度滑动窗口增加检测结果，解决图像目标形状复杂、尺寸不一问题，最后利用卷积神经网络和回归模型分类、定位目标。

2014 **YOLO**

Joseph Redmon 等提出的 YOLO 继承了 OverFeat 算法这种基于回归的 one stage 方法，速度能达到每秒 45 帧，由于其速度优势迅速成为端到端方法的领先者.

SSD 2016

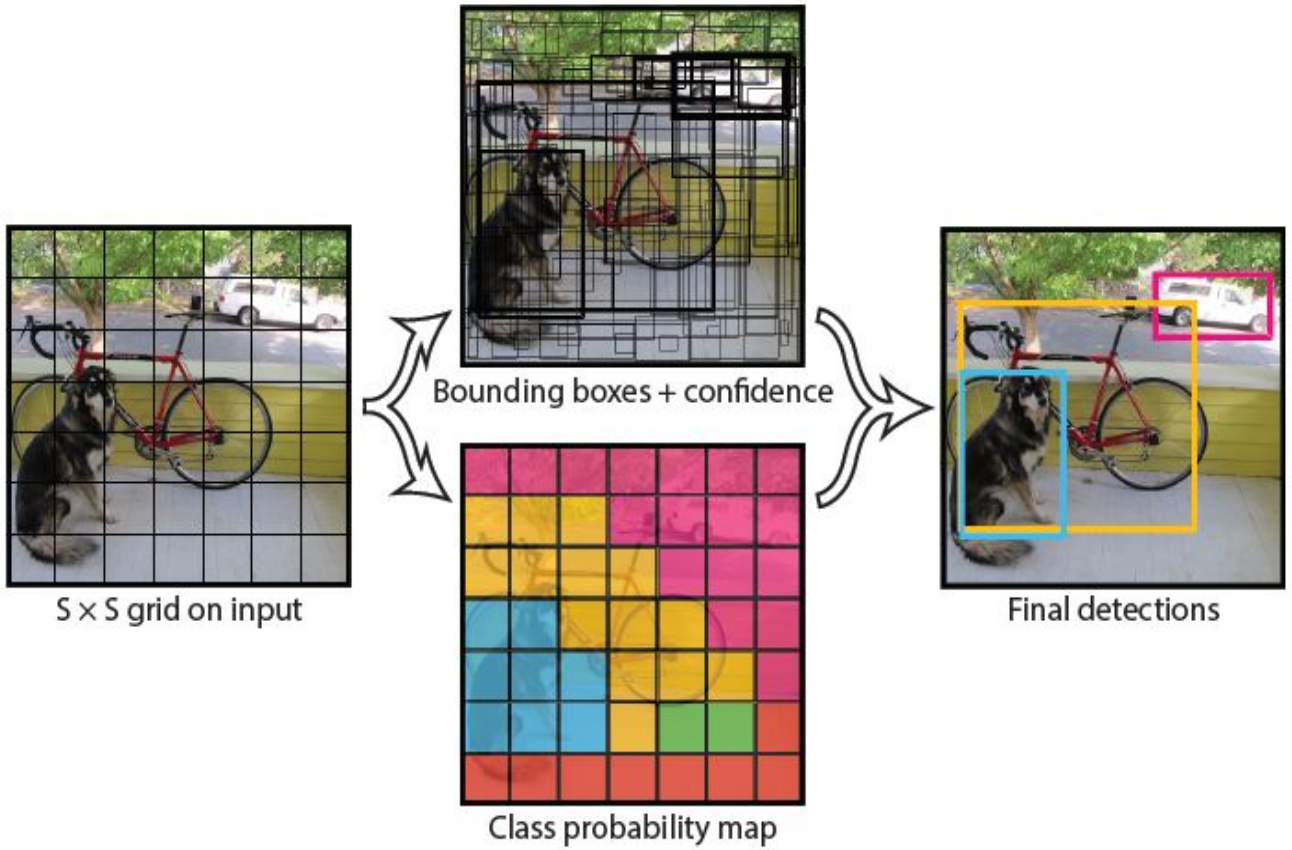
Wei Liu 等提出 SSD 算法，将 YOLO 的回归思想和 Faster R-CNN 的 anchor box 机制结合。

2017 **YOLOV2**

YOLOV3 2018

YOLOv2:https://arxiv.org/pdf/1612.08242v1.pdf
YOLOv3:https://pjreddie.com/media/files/papers/YOLOv3.pdf

YOLO检测流程



(1) 给个一个输入图像，首先将图像划分成7 * 7的网格。
(2) 对于每个网格，每个网格预测2个bouding box（每个box包含5个预测量）以及20个类别概率，总共输出7×7×（2*5+20）=1470个tensor
(3) 根据上一步可以预测出7 * 7 * 2 = 98个目标窗口，然后根据阈值去除可能性比较低的目标窗口，去除冗余窗口。

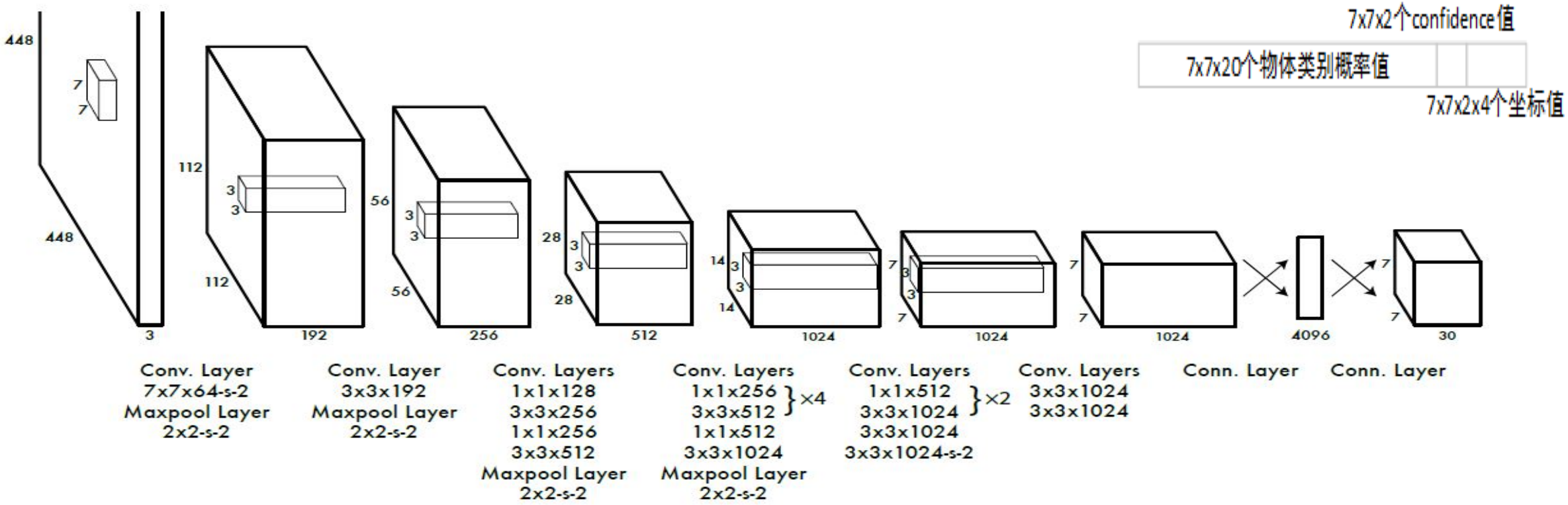
置信度的计算

每个边界框置信数：$Pr(Object) * IOU_{pred}^{truth}$

如果某网格单元中没有目标，则它的置信分数为0， 每个网格单元也预测C，类别的条件概率，即Pr(Classi|Object)。该概率值表示该网格单元包含某目标的可能性。同时这只预测每个网格单元某个类别集的概率，不考虑边界的数目.

每个网格的置信度：$Pr(class_i|Obejct) \cdot Pr(Object) \cdot IOU_{pred}^{truth} = Pr(Class_i) \cdot IOU_{pred}^{truth}$

网络结构



YOLO网络设计遵循了GoogleNet的思想，但与之有所区别。YOLO使用了**24个级联的卷积（conv）层和2个全连接（fc）层**，其中conv层包括3*3和1*1两种Kernel，最后一个fc层即YOLO网络的输出，长度为S*S*(B*5+C)=7*7*30.

损失函数

$$loss = \sum_{i=0}^{S^2} coordError + iouError + classError$$

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

坐标误差

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

IOU误差

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

分类误差

x,y,w,h,C,p为网络预测值，
$\hat{x}, \hat{y}, \hat{w}, \hat{h}, \hat{C}, \hat{p}$ 为标注值。

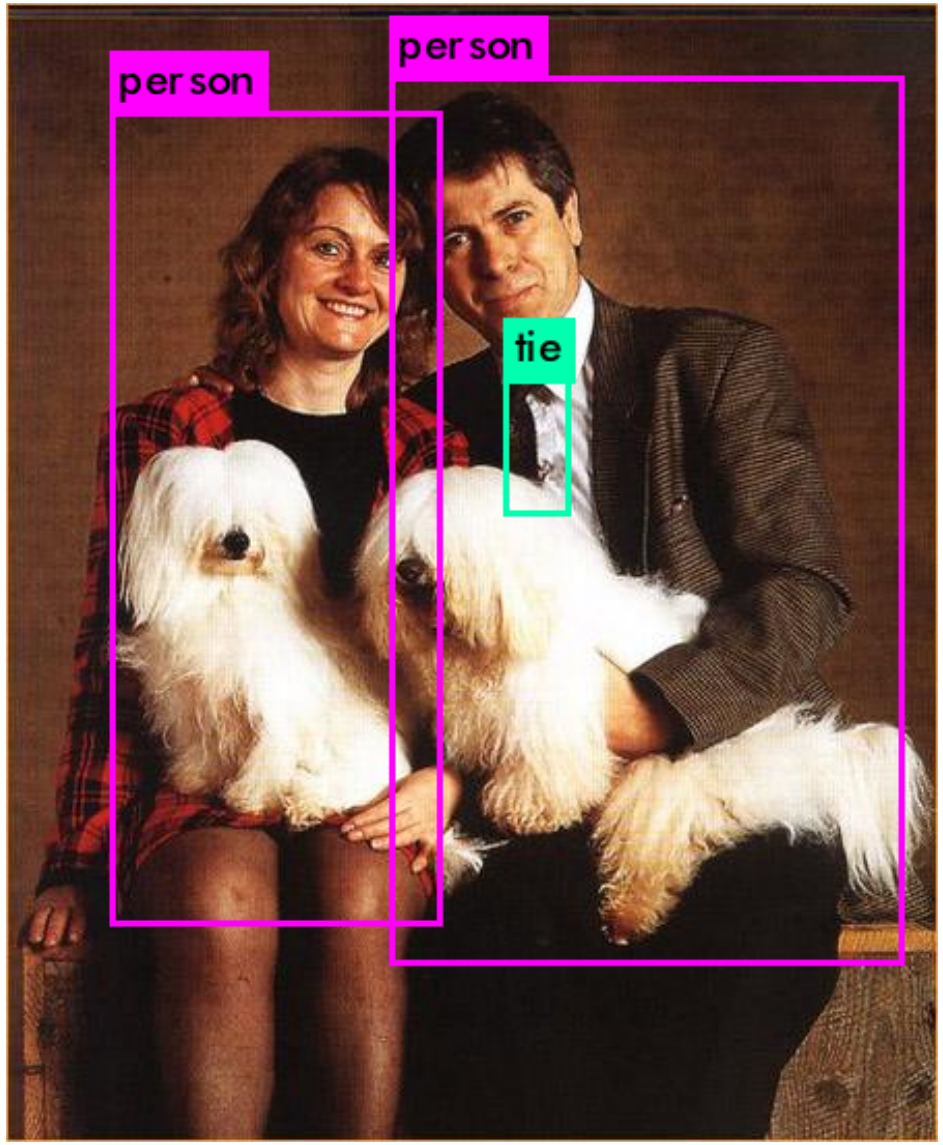
Π_{ij}^{obj} ：落入格子i的第j个bounding box内

Π_{i}^{obj} ：物体落在格子i中

YOLO的限制

1）每个格子只能预测出两个bounding box、一种类别，这导致模型对相邻目标的检测准确率下降。因此，YOLO对成堆目标的检测准确率较低。

2）损失函数对small bounding box、large bounding box的error平等对待，这将影响目标检测的准确率。因为对于small bounding box，small error 的影响更大。



实时检测系统对比

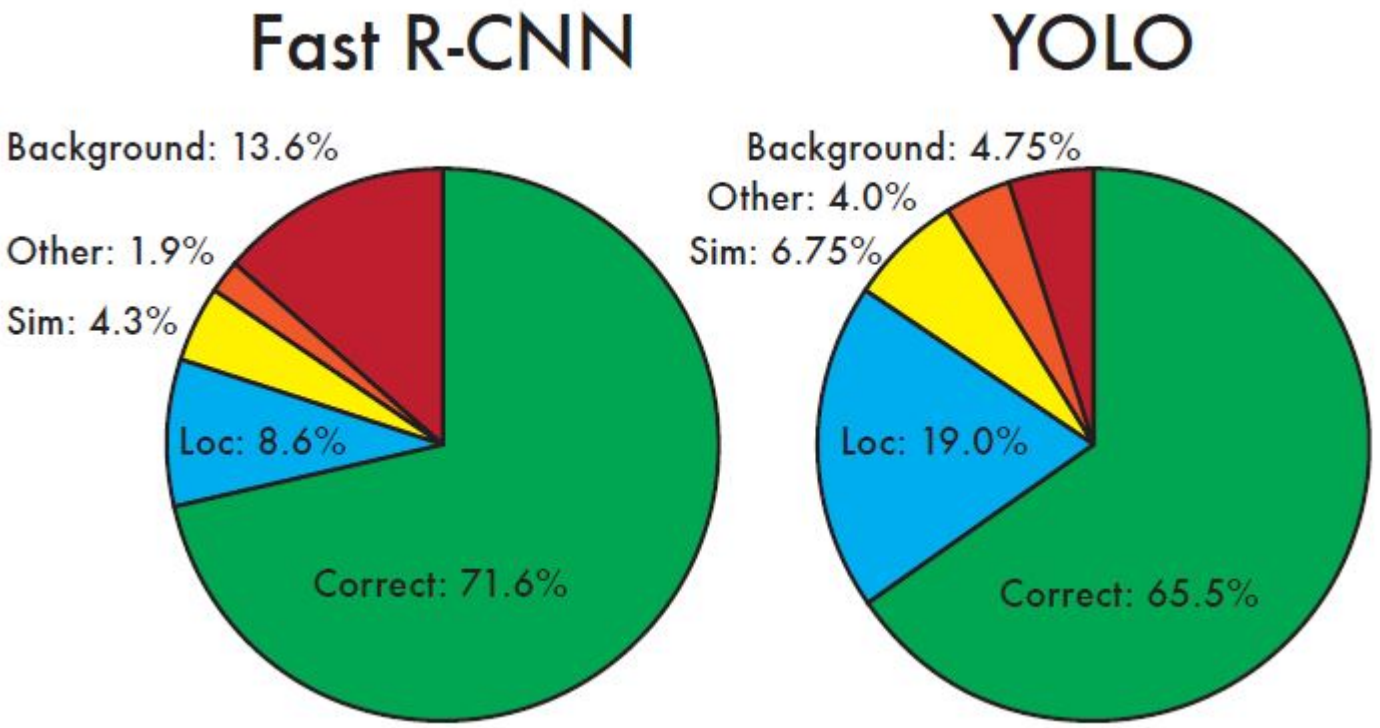
Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

在 VOC2012下对比mAP

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	**79.8**	87.7	49.6	74.9	52.1	86.0	81.7	83.3	**81.8**	**48.6**	**73.5**	59.4	79.9	65.7
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [28]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEP_ENS_COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [29]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [33]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

YOLO在检测小目标时准确率比R-CNN低大约8~10%，在检测大目标是准确率高于R-CNN。采用Fast-R-CNN+YOLO的方式准确率最高，比Fast-R-CNN的准确率高了2.3%。

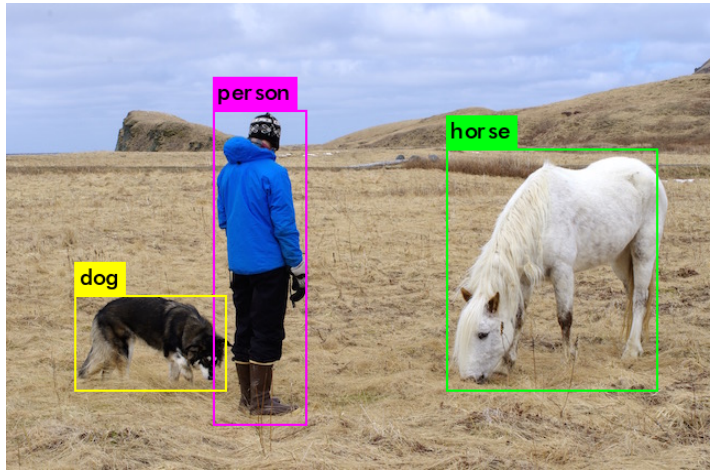
和Fast R-CNN对比

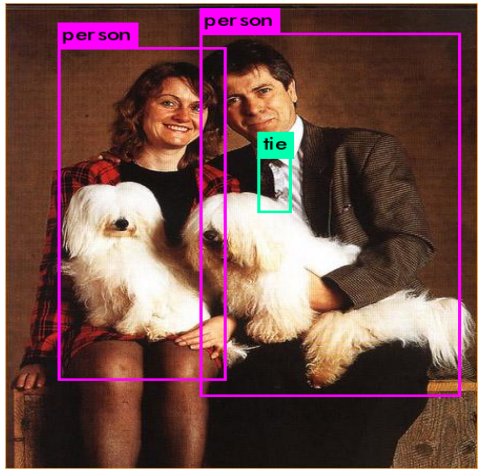


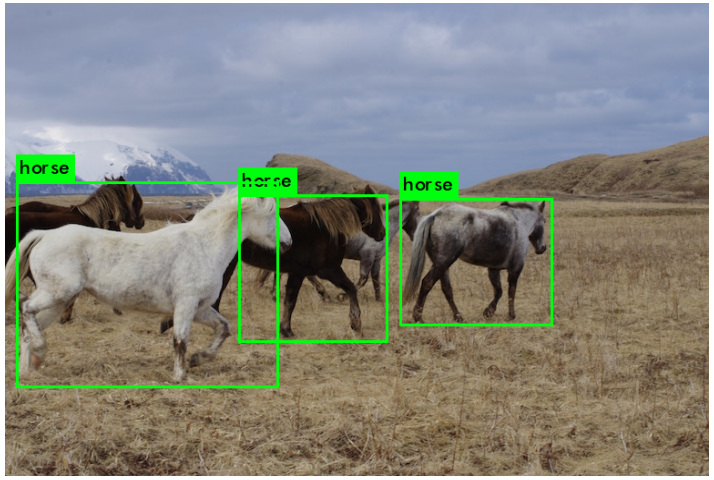
Correct：正确检测和识别的比例，即分类正确且IOU>0.5
Localization:分类正确，但0.1<IOU<0.5
Similar:类别相似，IOU>0.1
Other:分类错误，IOU>0.1
Background: 对于任何目标IOU<0.1

YOLO的error中，目标定位错误占据的比例最大，比Fast-R-CNN高出了10个点

原图







检测图

谢谢大家