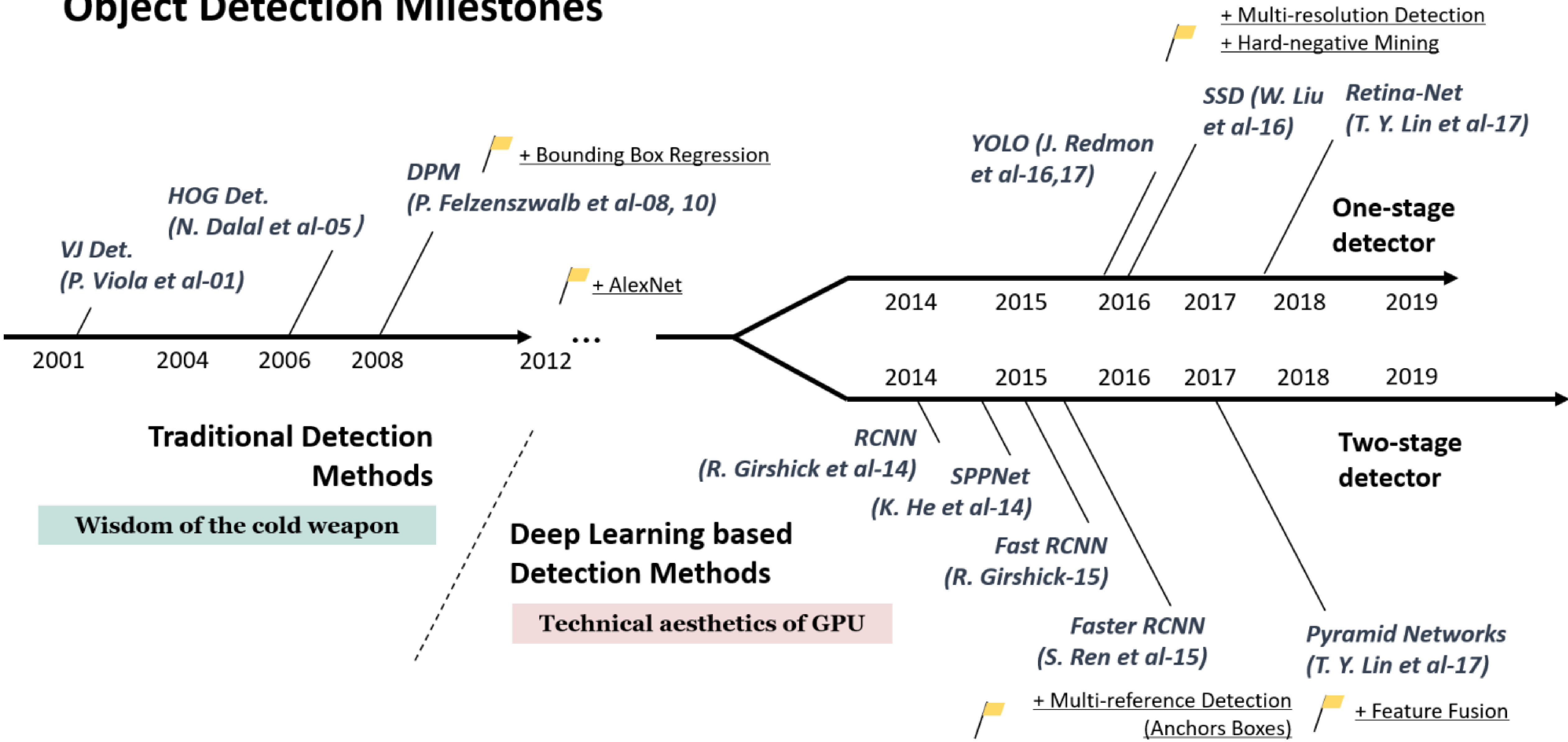
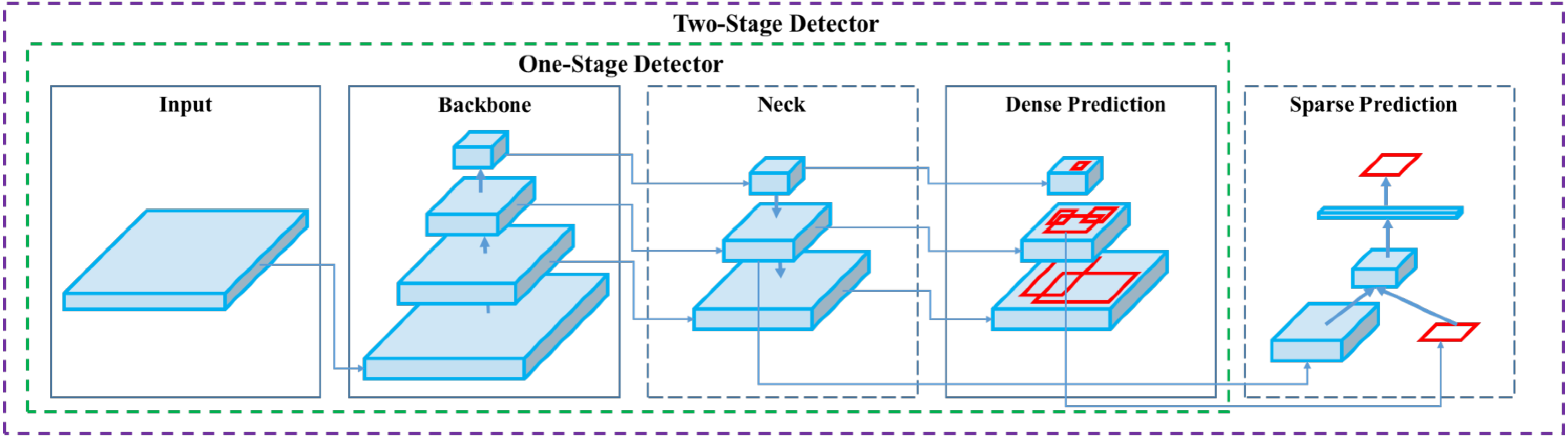
YOLO目标检测系列发展史

Object Detection Milestones





Input: { Image, Patches, Image Pyramid, … }

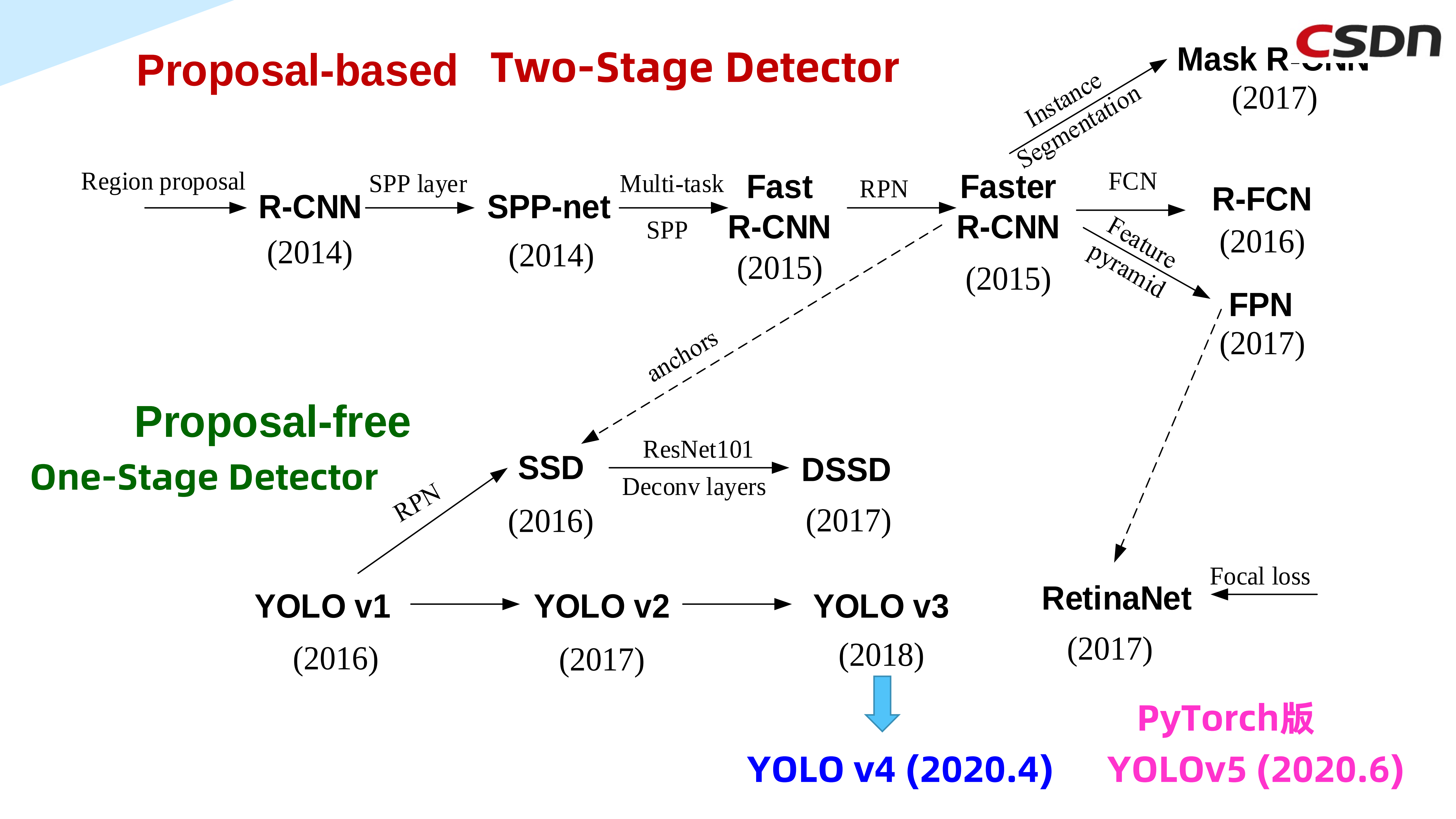
Backbone: { VGG16 [68], ResNet-50 [26], ResNeXt-101 [86], Darknet53 [63], … }

Neck: { FPN [44], PANet [49], Bi-FPN [77], … }

Head:

Dense Prediction: { RPN [64], YOLO [61, 62, 63], SSD [50], RetinaNet [45], FCOS [78], … }

Sparse Prediction: { Faster R-CNN [64], R-FCN [9], … }



Proposal-based
Two-Stage Detector
Mask R-
(2017)
Instance
Segmentation
Region proposal
R-CNN
(2014)
SPP layer
SPP-net
(2014)
Multi-task
SPP
Fast
R-CNN
(2015)
RPN
Faster
R-CNN
(2015)
FCN
R-FCN
(2016)
Feature
pyramid
FPN
(2017)
anchors
Proposal-free
One-Stage Detector
RPN
SSD
(2016)
ResNet101
Deconv layers
DSSD
(2017)
YOLO v1
(2016)
YOLO v2
(2017)
YOLO v3
(2018)
RetinaNet
(2017)
Focal loss
YOLO v4 (2020.4)
PyTorch版
YOLOv5 (2020.6)

Darknet

Darknet is an open source neural network framework written in C and CUDA. It is fast, easy to install, and supports CPU and GPU computation.

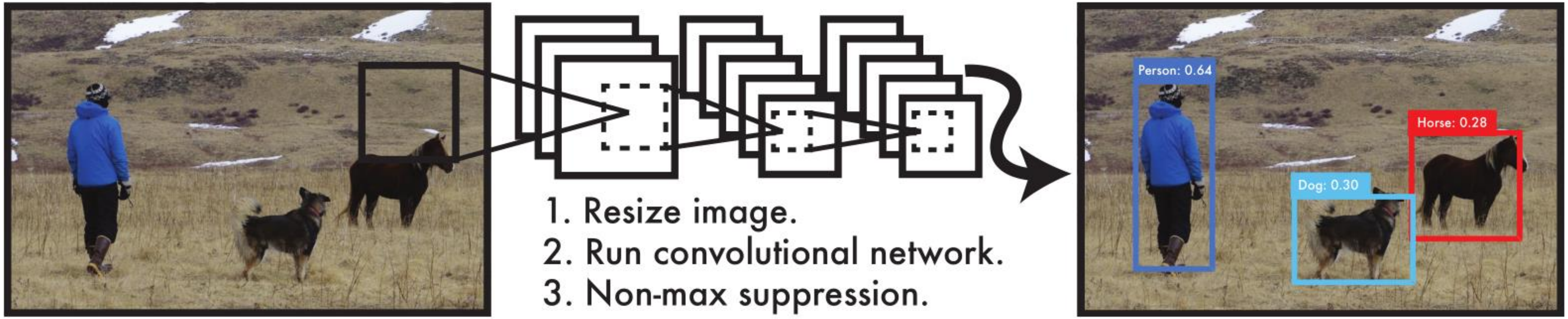
Yolo v4 paper: https://arxiv.org/abs/2004.10934

Yolo v4 source code: https://github.com/AlexeyAB/darknet

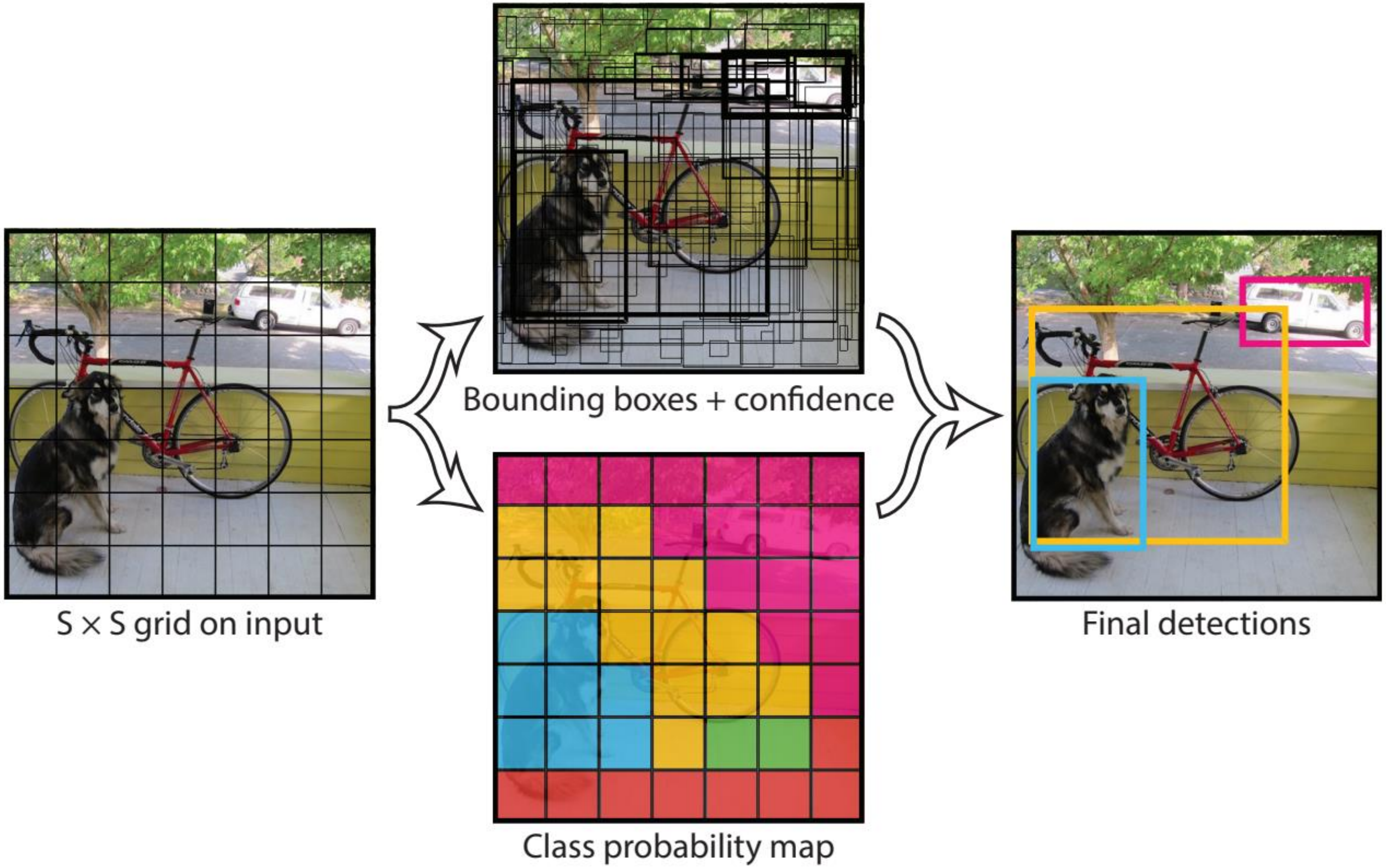
For more information see the Darknet project website.

You Only Look Once: Unified, Real-Time Object Detection

- YOLO将特征图划分为S×S的格子(grid cells)， 每个格子负责对落入其中的目标进行检测，一次性预测所有格子所含目标的边界框、定位置信度、以及所有类别概率向量。



YOLO算法的基本思想



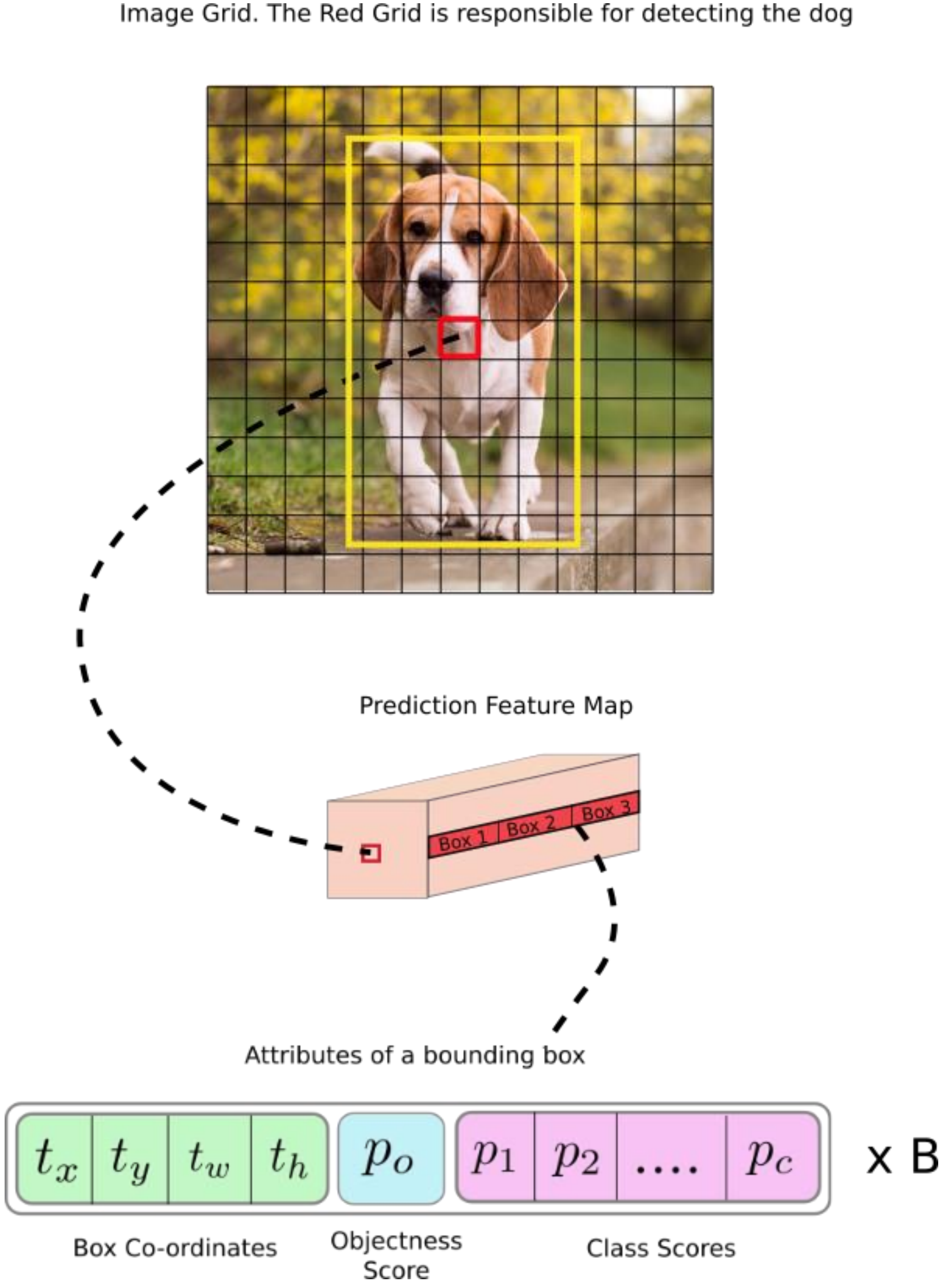
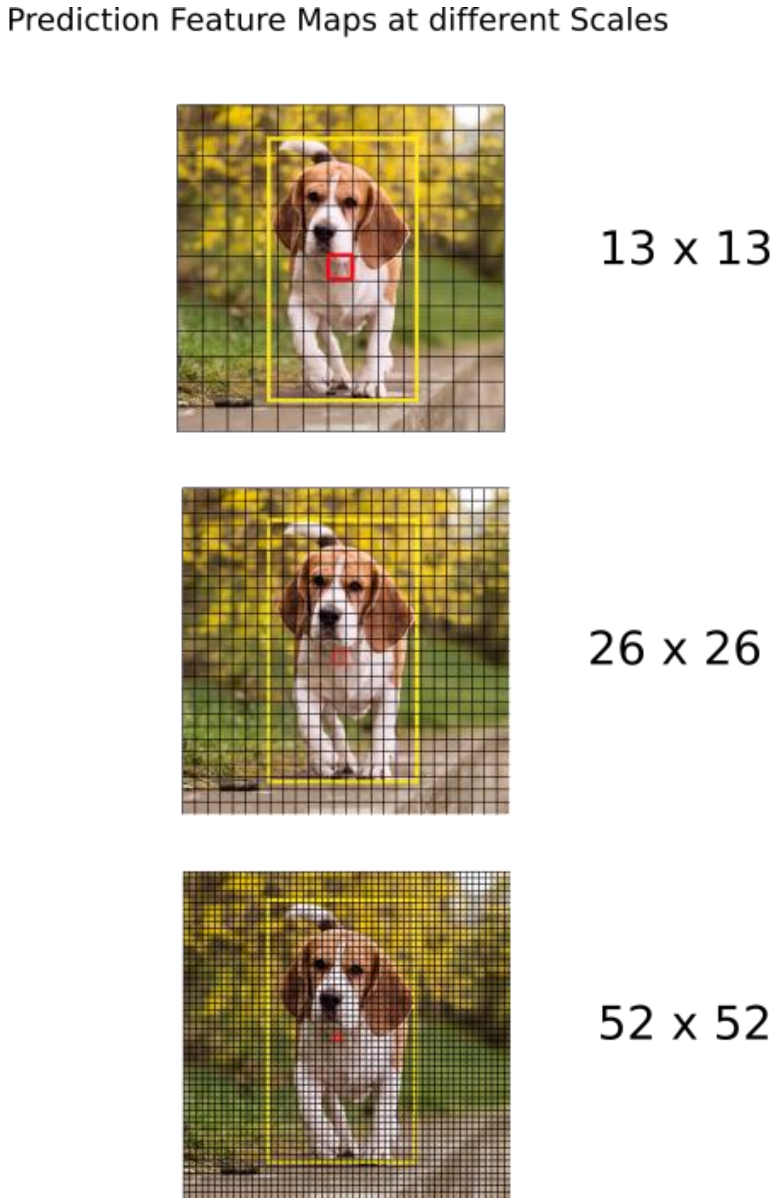
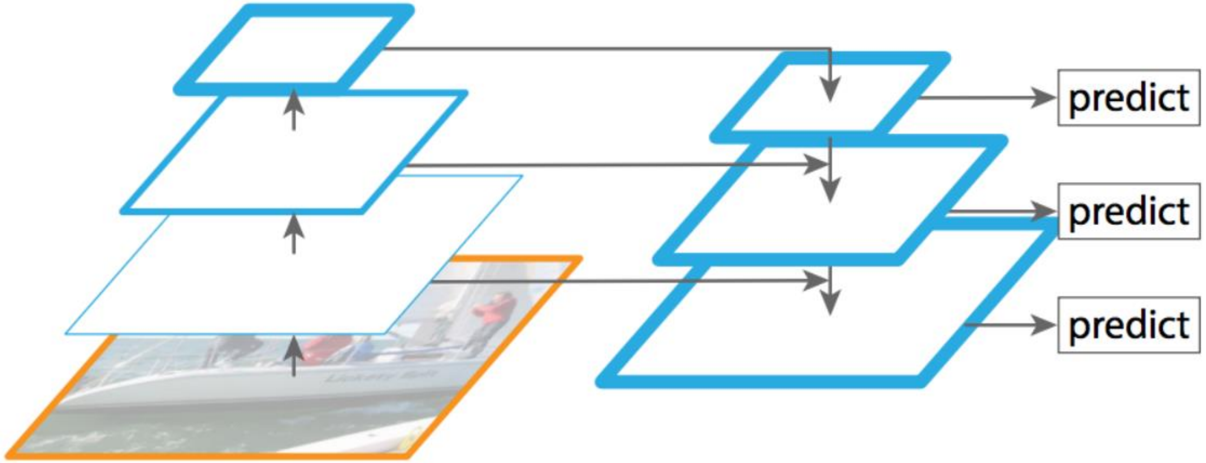


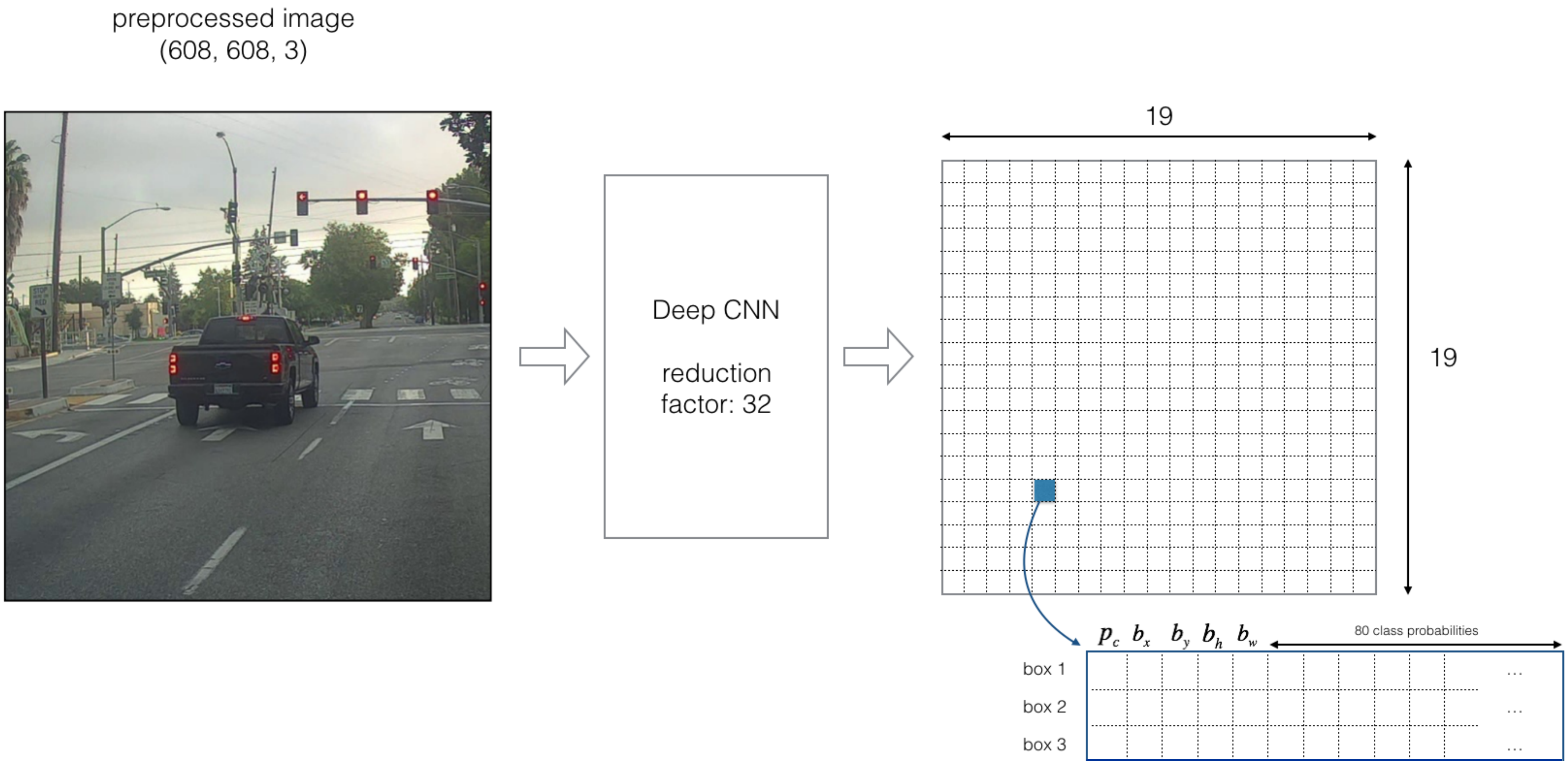
Image Grid. The Red Grid is responsible for detecting the dog
Prediction Feature Map
Box 1
Box 2
Box 3
Attributes of a bounding box
t_x t_y t_w t_h
p_o
p_1 p_2 .... p_c
x B
Box Co-ordinates
Objectness Score
Class Scores

多尺度融合

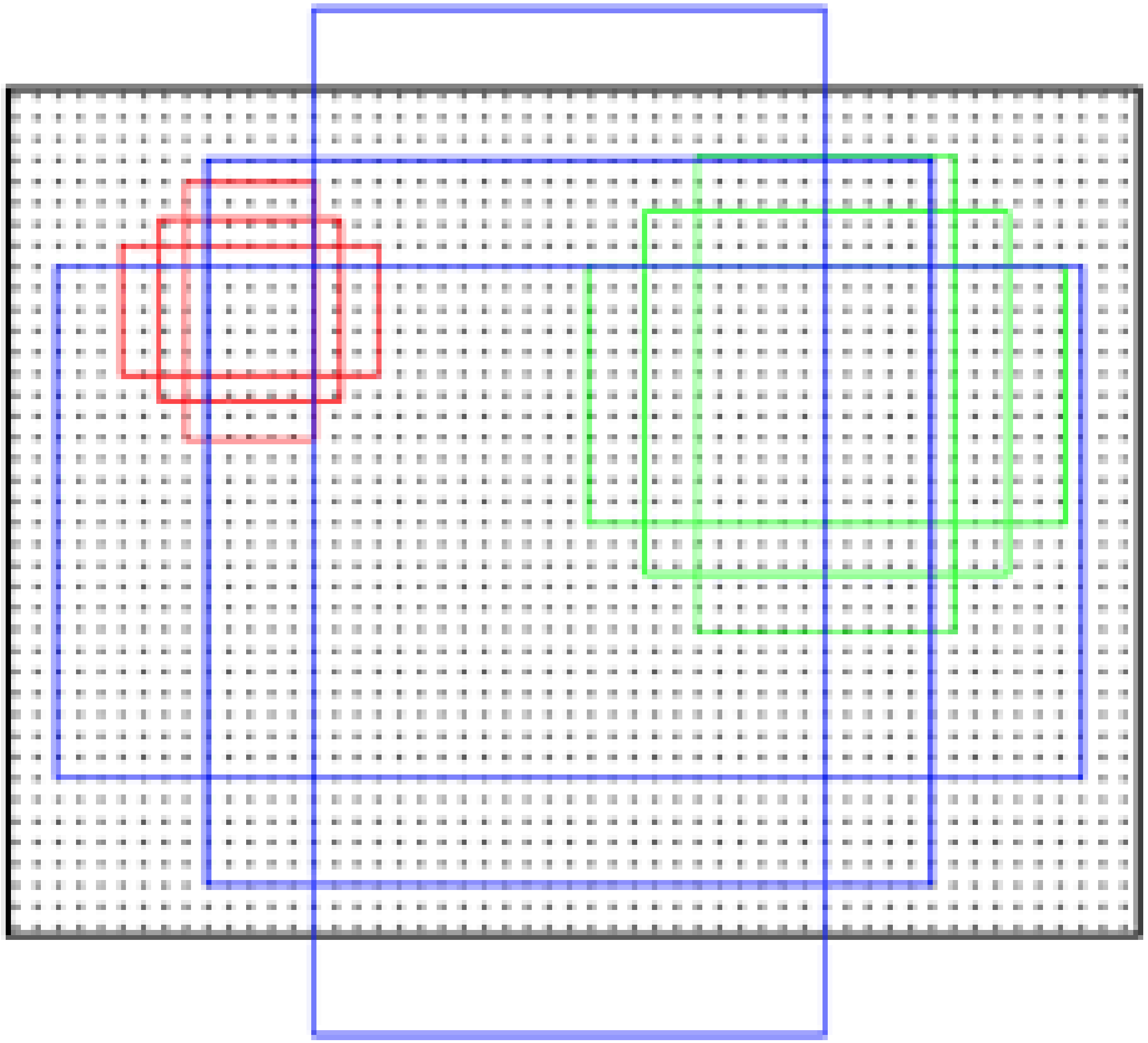


Feature Pyramid Network (FPN)

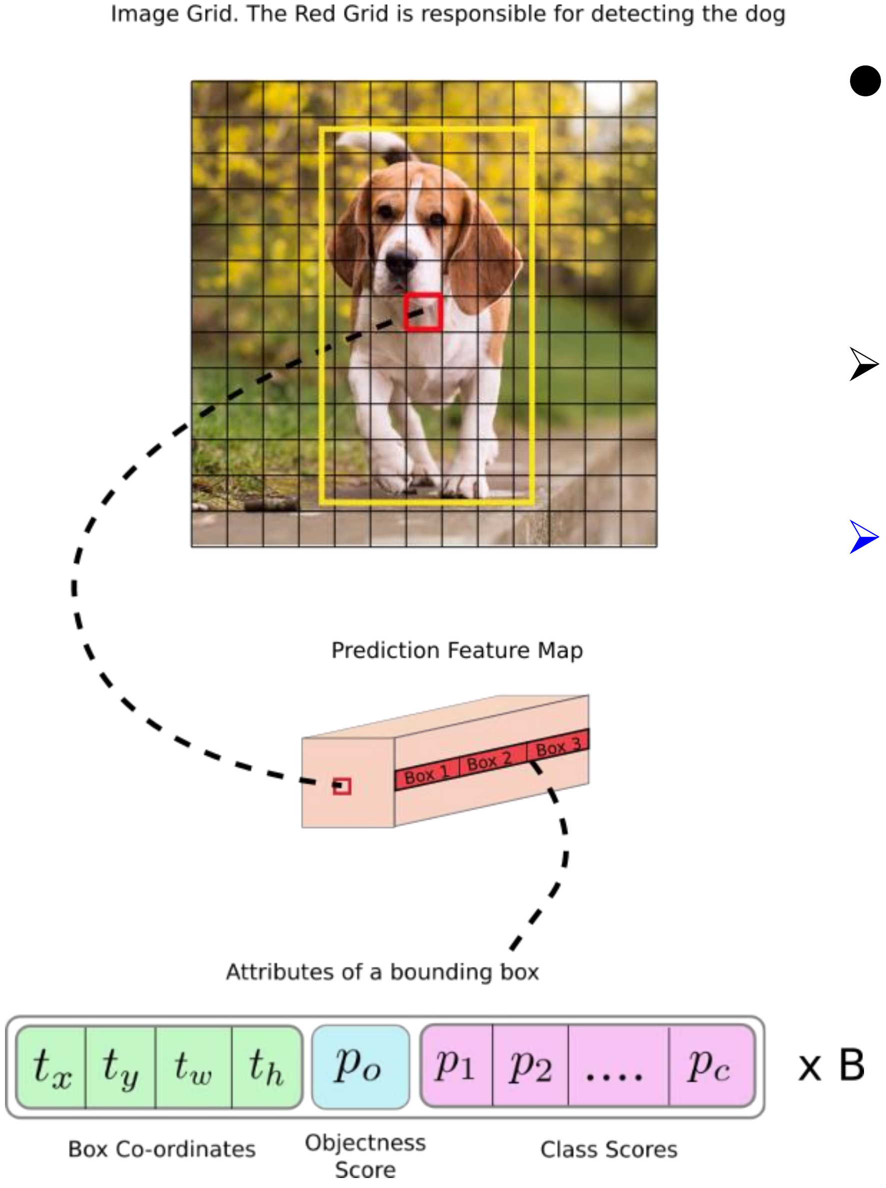



preprocessed image
(608, 608, 3)
Deep CNN
reduction
factor: 32
19
19
p_c b_x b_y b_h b_w
80 class probabilities
box 1
box 2
box 3
...
...
...

Anchor（锚框）机制



YOLO算法的基本思想

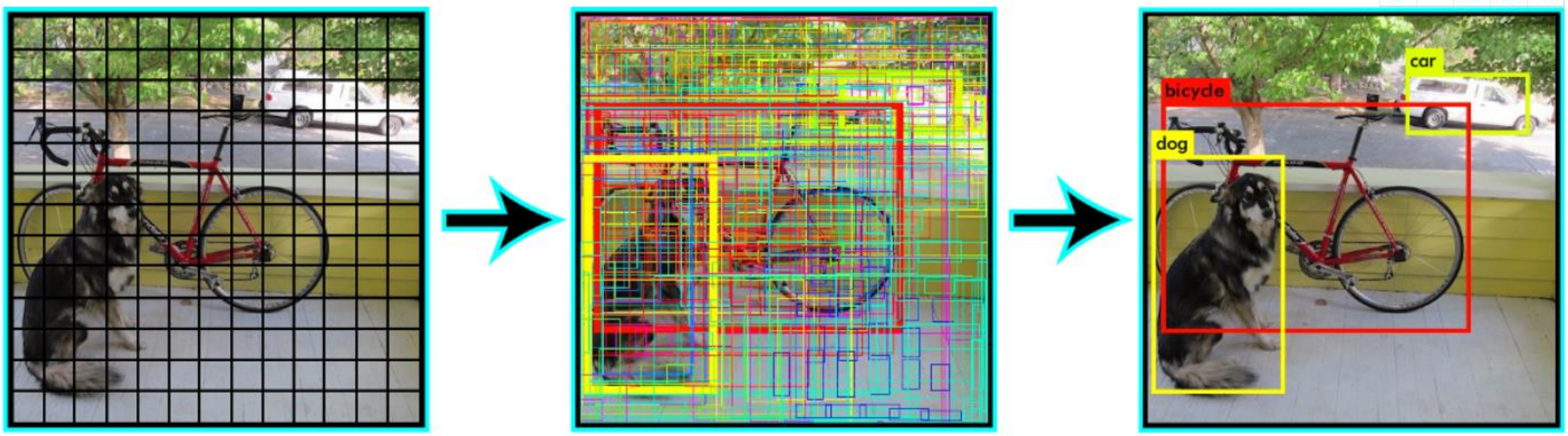


- 首先通过特征提取网络对输入图像提取特征，得到一定大小的特征图，比如13×13（相当于416×416图片大小），然后将输入图像分成13×13个grid cells

- YOLOv3/v4: 如果GT中某个目标的中心坐标落在哪个grid cell中，那么就由该grid cell来预测该目标。每个grid cell都会预测3个不同尺度的边界框。
- YOLOv5: 不同于yolov3/v4，其GT可以跨层预测，即有些bbox在多个预测层都算正样本；匹配数范围可以是3-9个。

- 预测得到的输出特征图有两个维度是提取到的特征的维度，比如13×13，还有一个维度（深度）是 B×（5+C）

- 注：B表示每个grid cell预测的边界框的数量（YOLO v3/v4中是3个）；C表示边界框的类别数（没有背景类，所以对于VOC数据集是20）； 5表示4个坐标信息和一个目标性得分（objectness score）。



每个预测框的类别置信度得分(class confidence score) 计算如下：

$$\text{class confidence score} = \text{box confidence score} \times \text{conditional class probability}$$

它测量分类和定位（目标对象所在的位置）的置信度。

条件类别概率　　框置信度得分　　类别置信度得分

$$\boxed{\Pr(\text{Class}_i|\text{Object})} * \boxed{\Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}}} = \boxed{\Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}}$$

$$\begin{aligned}\text{box confidence score} &\equiv P_r(object) \cdot IoU \\ \text{conditional class probability} &\equiv P_r(class_i|object) \\ \text{class confidence score} &\equiv P_r(class_i) \cdot IoU \\ &= \text{box confidence score} \times \text{conditional class probability}\end{aligned}$$

where

$P_r(object)$ is the probability the box contains an object.

IoU is the IoU (intersection over union) between the predicted box and the ground truth.

$P_r(class_i|object)$ is the probability the object belongs to $class_i$ given an object is presence.

$P_r(class_i)$ is the probability the object belongs to $class_i$

NMS (Non-Maximum Suppression)
非极大抑制

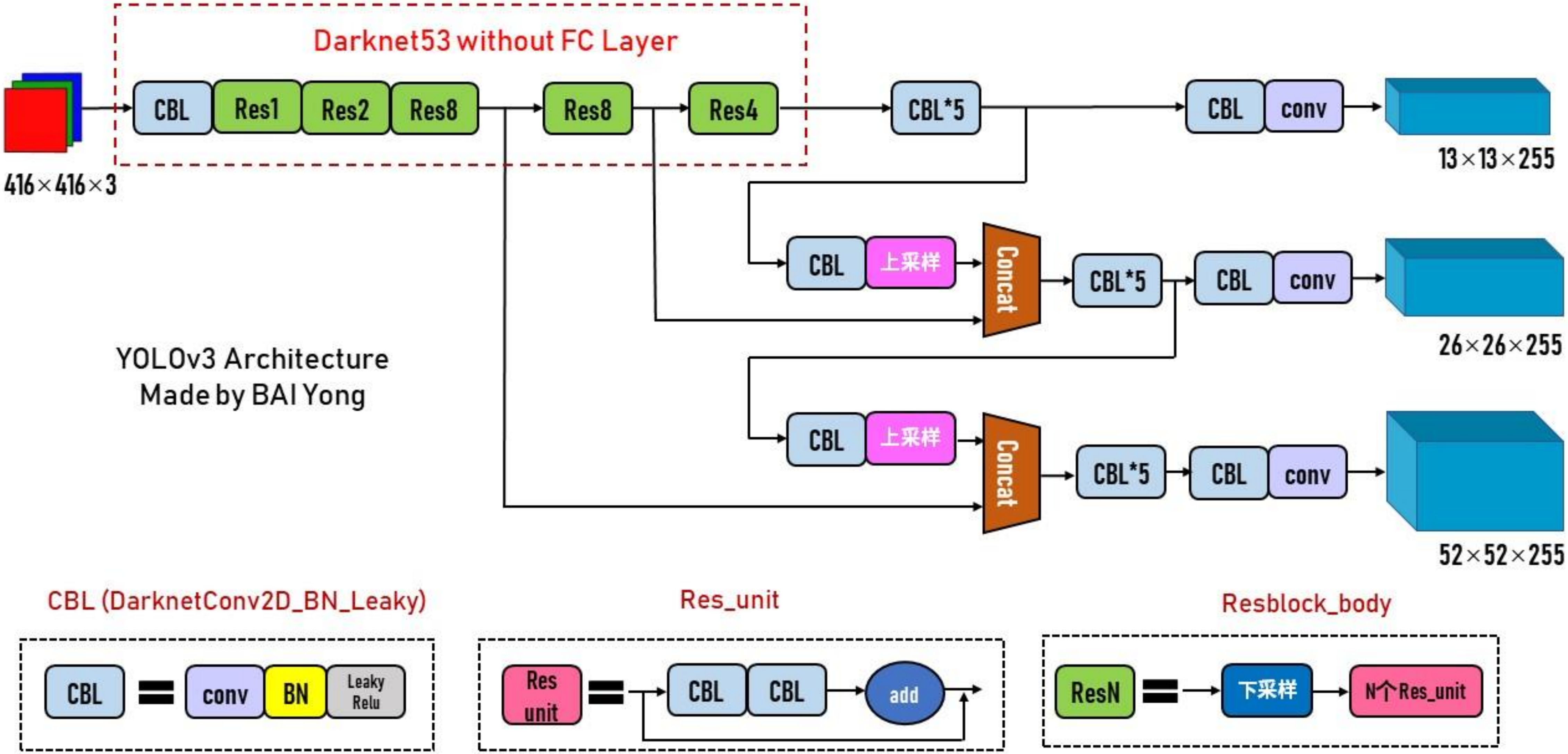
测试时没有GT框，只能比较多个预测框，比较相互之间的IOU，做NMS

Multiple Grids may detect the same object
NMS is used to remove multiple detections

Before non-max suppression

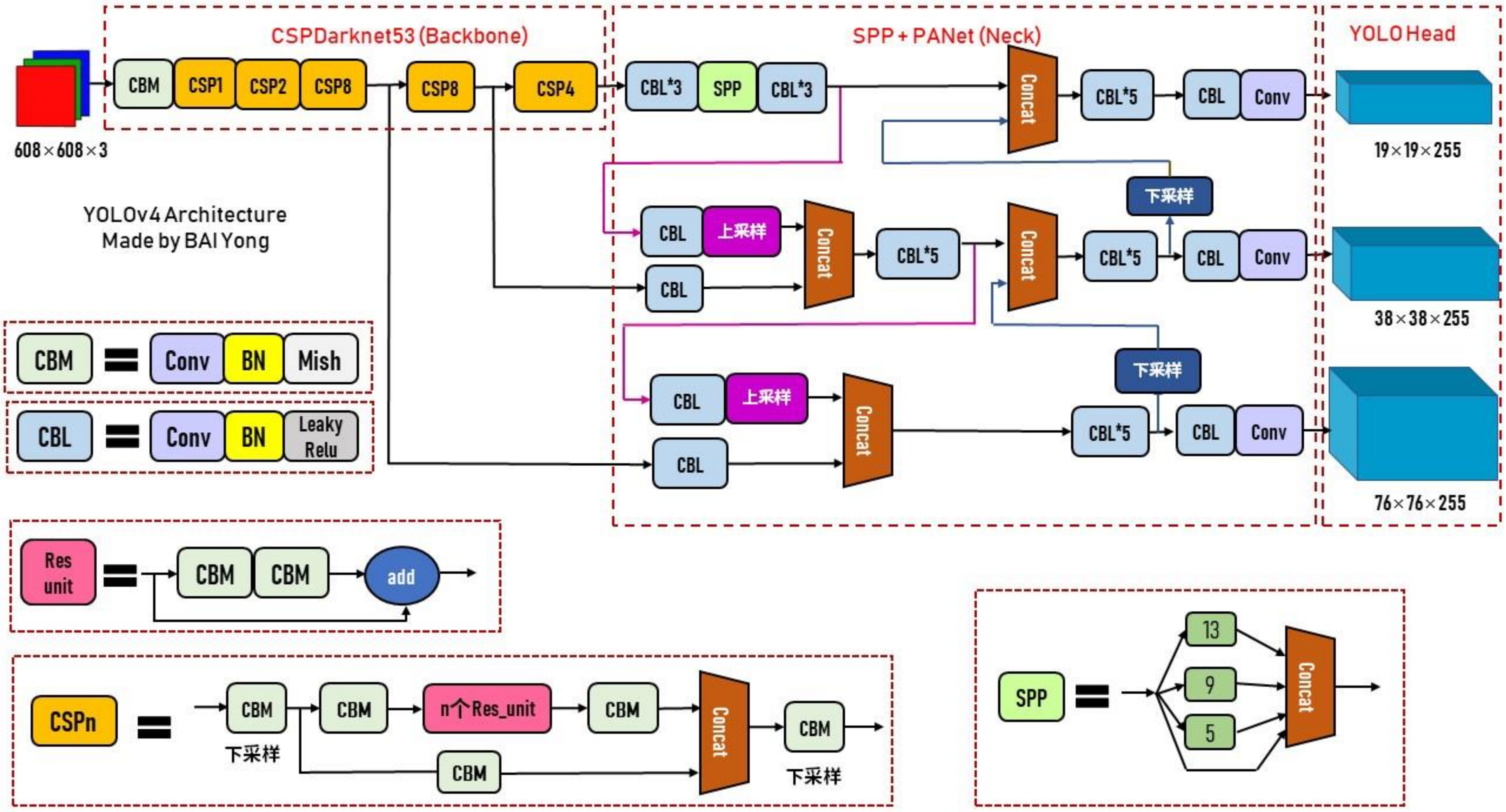
After non-max suppression

YOLOv3 网络架构



Darknet53 without FC Layer
416×416×3
CBL
Res1
Res2
Res8
Res8
Res4
CBL*5
CBL
conv
13×13×255
CBL
上采样
Concat
CBL*5
CBL
conv
26×26×255
CBL
上采样
Concat
CBL*5
CBL
conv
52×52×255
YOLOv3 Architecture
Made by BAI Yong
CBL (DarknetConv2D_BN_Leaky)
CBL
conv
BN
Leaky Relu
Res_unit
Res unit
CBL
CBL
add
Resblock_body
ResN
下采样
N个Res_unit

YOLOv4 网络架构



CSPDarknet53 (Backbone)
SPP + PANet (Neck)
YOLO Head
608×608×3
CBM
CSP1
CSP2
CSP8
CSP8
CSP4
CBL*3
SPP
CBL*3
Concat
CBL*5
CBL
Conv
19×19×255
下采样
CBL
上采样
Concat
CBL*5
Concat
CBL*5
CBL
Conv
CBL
38×38×255
下采样
CBL
上采样
Concat
CBL*5
CBL
Conv
CBL
76×76×255
YOLOv4 Architecture
Made by BAI Yong
CBM = Conv BN Mish
CBL = Conv BN Leaky Relu
Res unit = CBM CBM add
CSPn = CBM 下采样 CBM n个Res_unit CBM CBM Concat CBM 下采样
SPP = 13 9 5 Concat

损失函数(Loss function)

损失函数包括：

- **classification loss,** 分类损失
- **localization loss,** 定位损失（预测边界框与GT之间的误差）
- **confidence loss,** 置信度损失（框的目标性；objectness of the box）

总的损失函数：

classification loss + localization loss + confidence loss